ОГЛАВЛЕНИЕ

УДК 004.822

МОДЕЛЬ ЛИНГВИСТИЧЕСКОГО ГРАФА ЗНАНИЙ «TURKLANG» КАК БАЗА ДЛЯ СОЗДАНИЯ ИНСТРУМЕНТОВ ОБУЧЕНИЯ ТЮРКСКИМ ЯЗЫКАМ

А. Р. Гатиатуллин[1] [0000-0003-3063-8147], Н. А. Прокопьев[2] [0000-0003-0066-7465]

[1]*Академия наук РТ;* [2]*Казанский Федеральный Университет*

[1]ayrat.gatiatullin@gmail.com, [2]nikolai.prokopyev@gmail.com

Аннотация

Описаны элементы модели лингвистического графа знаний «Turklang», разработанного в Институте прикладной семиотики АН РТ и используемого в качестве базы для создания ряда лингвистических ресурсов и инструментов: портал «Тюркская морфема», электронный корпус татарского языка «Туган Тел», лингвистические процессоры.

Для создания образовательной среды необходимы предметно-ориентированные графы знаний, для получения которых не применимы методы создания общих и открытых графов. В работе описаны лингвистические графы знаний, которые отображают, с одной стороны, потенциальные возможности тюркских языков, с другой стороны, примеры реального использования в текстах. Особенность этих графов знаний заключается в том, что они содержат лингвистические единицы разных языковых уровней, а также семантические универсалии, соответствующие значениям этих лингвистических единиц, которые встроены в единую модель лингвистического графа знаний. Структура такого графа знаний позволяет формировать учебные курсы, строить индивидуальную образовательную траекторию, а также создавать задания и средства автоматизированной проверки в рамках контроля знаний при обучении тюркским языкам. Это дает возможность разрабатывать впоследствии, на основе этих графов, программы обучения с учетом структурно-функциональных особенностей тюркских языков, а также способствует реализации индивидуальных целей обучающихся.

Ключевые слова*: граф знаний, база знаний, лингвистический ресурс, лингвистическая единица, малоресурсные языки, тюркские языки, веб-портал, электронное образование, контроль знаний, автоматизированная оценка ответа*

ВВЕДЕНИЕ

В 1991 году Т. Джонс [1] выдвинул идею обучения языку на основе лингвистических баз данных и гипотезу, что обучение языкам будет более эффективным, если обучающийся сам будет выступать в роли исследователя языка, а учитель будет обеспечивать ему контекст и направления познания языка. Таким образом, учащиеся имеют возможность работать с лингвистическими базами данных и проводят исследование в своих учебных целях. Т. Джонс выделяет следующие преимущества данного подхода к обучению:

1. Обучающийся учится видеть языковые структуры, искать аналогии и обобщать полученные данные. Кроме того, использование аутентичных материалов делает речь обучающихся более идиоматичной, приближая ее к речи носителей языка.
2. Преподаватель вместо транслятора информации о языке становится координатором исследований ученика, что позволяет ученику самостоятельно обрабатывать потоки информации «об особенностях употребления языковой формы, а также решать задачи, связанные с ее осознанием».
3. Изменяется роль грамматики в изучении иностранного языка. Утверждается, что грамматика языка не способна отразить все разнообразие его синтаксических структур, поэтому неэффективно изучать грамматику отдельно от области функционирования правил грамматики. Использование в учебном процессе лингвистических ресурсов может сделать этот процесс более естественным.

Мы считаем, что приведенные положения справедливы и для тюркских языков, которые обладают богатой морфологией, и для проверки названной гипотезы необходимо наличие лингвистических ресурсов, содержащих тюркские лингвистические базы знаний. С учетом структурной близости тюркских языков эти ресурсы могут быть универсальными для всех тюркских языков. Свою положительную роль может сыграть и многоязычность этих ресурсов, потому что обучаемый сможет сравнивать языковые примеры в разных тюркских языках, акцентируя внимание на особенностях конкретного изучаемого языка.

В работе [2] авторы используют идеи, выдвинутые Т. Джонсом [1], и выделяют три вида учебных материалов, которые обучающиеся могут использовать для изучения иностранного языка:

1. Электронная лексикография;
2. Корпусные исследования;
3. Проектирование электронных учебников и учебных терминологических баз данных.

К преимуществам изучения иностранного языка с использованием перечисленных учебных материалов, с учетом идей Т. Джонса, авторы [2] добавили ряд дополнительных пунктов:

1. Работа с актуальным материалом;
2. Развитие исследовательских навыков;
3. Усвоение естественного построения речи.

Авторы работы [2] также считают, что ценность лингвистической базы данных при изучении иностранного языка повышается, если она сочетает в себе учебную, справочную, систематизирующую и коммуникативную функции.

Воспользуемся гипотезами, выдвинутыми авторами перечисленных работ, для их апробации на тюркских языках. Для проверки сформулированных утверждений необходимо наличие лингвистических ресурсов, которые, как указано ранее, должны содержать в себе учебную, справочную, систематизирующую и коммуникативную функции. Однако в настоящее время практически все тюркские языки, кроме турецкого, являются малоресурсными языками, так как испытывают недостаток в лингвистических ресурсах различных типов. В Институте прикладной семиотики Академии наук Республики Татарстан разрабатывается ряд лингвистических ресурсов для компьютерной обработки тюркских языков. В роли наиболее значимых из них можно выделить:

1. Лингвистический портал «Тюркская морфема»;
2. Электронный корпус «Туган тел».

Электронный корпус «Туган тел» ранее был реализован в двух версиях на разных технологических платформах, в настоящее время создается третья версия электронного корпуса на базе графовых баз данных. Графовые базы данных позволяют более эффективно представлять в корпусе синтаксическую и семантическую информацию.

В основе новой версии электронного корпуса «Туган тел» и лингвистического портала «Тюркская морфема» лежит единая лингвистическая модель знаний «TurkLang», реализованная в виде лингвистического графа знаний. Эта модель также является разработкой Института прикладной семиотики АН РТ.

ЛИНГВИСТИЧЕСКИЕ ГРАФЫ ЗНАНИЙ

В настоящее время одним из эффективных способов представления лингвистической информации в различных ресурсах являются графы знаний. Имеется целый ряд работ с описанием лингвистических графов знаний, в основном зарубежных, что показывает невысокую развитость отечественных разработок и исследований в данном направлении.

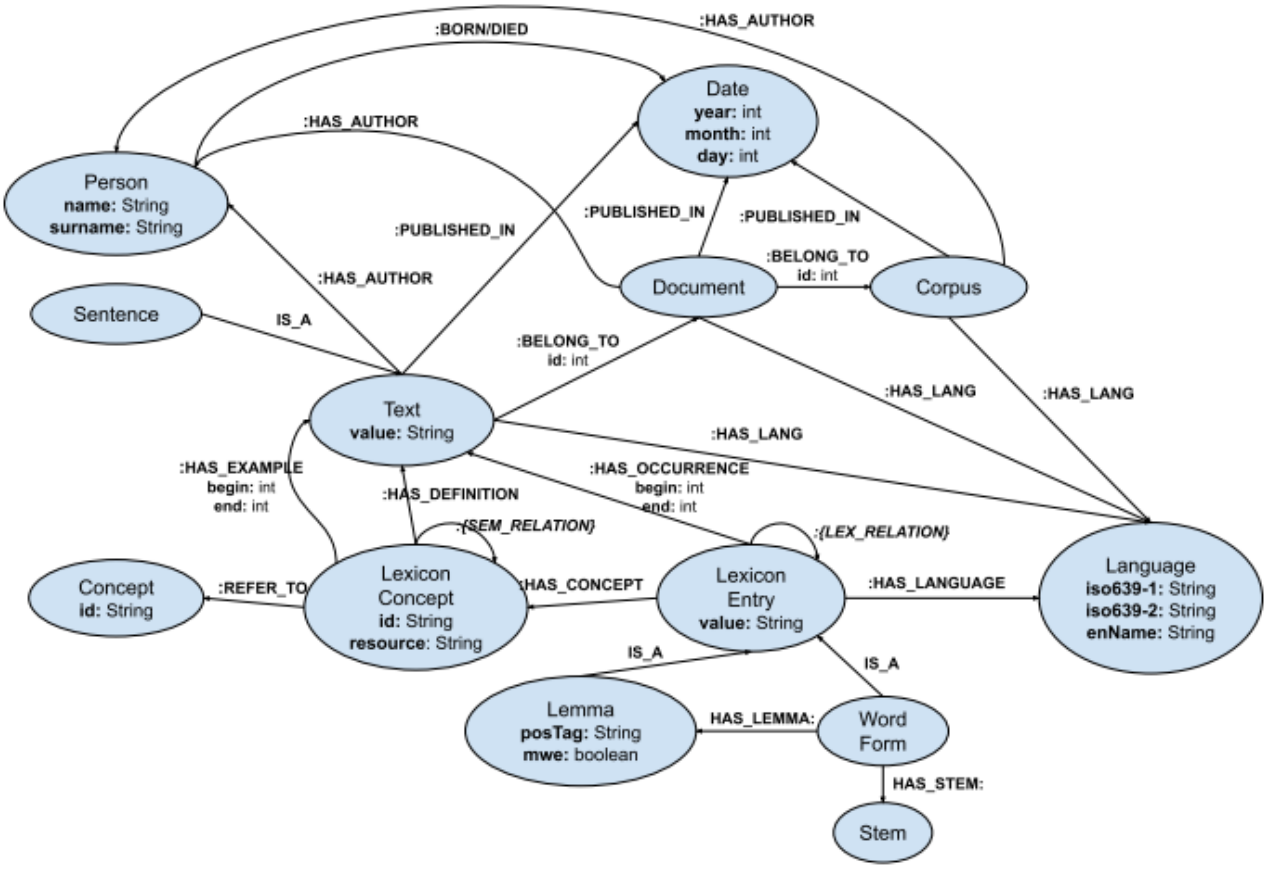


Рис. 1. Модель лингвистического графа знаний

Рассмотрим один из примеров, который, на наш взгляд, наиболее близок к требованиям, предъявляемым к графам знаний для представления тюркского лингвистического ресурса. Таким примером является лингвистический граф, описанный в работе [3] (модель данного лингвистического графа знаний представлена на рисунке 1). По утверждению авторов, этот граф позволяет моделировать:

1. Отношения между концептами и их лексическим представлением;
2. Информацию о статистике слов;
3. Диахроническую информацию как понятий, так и слов.

Лингвистический граф, описанный в названной работе, включает такие вершины, как концепты (Concept), лексические концепты (Lexicon Concept) и лексические входы или лексемы (Lexicon Entry). Лексические концепты связаны между собой таксономическими отношениями типа гипонимии и гиперонимии. Лексические входы связаны как с леммой (Lemma), так и с основой словоформы (Stem). Лемма – это словарная форма. В ряде случаев лемма и основа совпадают, что зависит от типа языка.

Недостатком этого графа знаний для представления полнотекстовой информации является то, что в нем отсутствует возможность описания ситуационно-фреймовой семантики. Данный граф позволяет описывать только лексическую информацию, аналогичную той, что представлена в известном электронном лингвистическом тезаурусе WordNet.

Для описания ситуационных сценариев подходящим ресурсом являются графы знаний фреймового типа. Самыми известными и наиболее заполненными из них являются такие лингвистические базы знаний, как FrameNet и VerbNet. Поэтому необходимо включение в многоуровневый лингвистический граф знаний элементов ресурсов такого типа.

Также при том, что данный граф предназначен для представления словарной информации, в нем также отсутствует возможность грамматической (морфологической и синтаксической) структуры лингвистических единиц. Учитывая богатую морфологическую структуру тюркских языков, можно утверждать, что это является необходимым условием для представления данных о тюркских текстах.

АРХИТЕКТУРА ЛИНГВИСТИЧЕСКОГО ГРАФА ЗНАНИЙ «TURKLANG»

На основе проделанного анализа лингвистических графов знаний нами была создана модель лингвистического графа знаний «TurkLang» для описания тюркских языков. Главное отличие этого графа знаний основано на структурно-функциональных особенностях тюркских языков. В тюркских языках существует четкое деление на структурные компоненты слова, именуемые морфемами. Та-

кое деление позволяет представить морфологическую структуру тюркской словоформы в виде графа, вершинами которого являются морфемы, а ребрами – порядок следования в словоформе.

Модель лингвистического графа знаний для описания тюркских языков «TurkLang» представляет единый граф знаний, который подразделяется на несколько подграфов. Подобное разделение связано с содержимым этих подграфов и с тем, что представляют собой вершины и ребра этих подграфов (схема разделения на подграфы представлена на рис. 2).

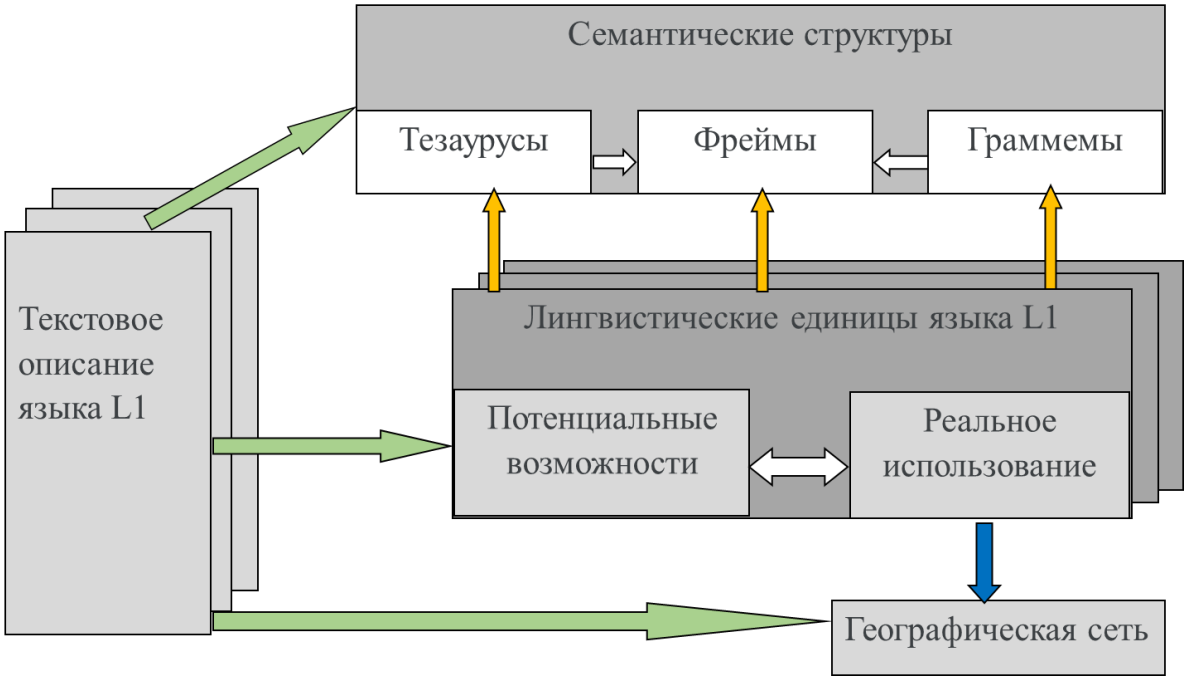


Рис. 2. Архитектура подграфов графа знаний портала

Подграф «Семантические структуры» сам является комбинацией нескольких подграфов, содержащих различные семантические универсалии (семантические единицы, универсальные для всех тюркских языков). «Тезаурусы» – это подграф, вершинами которого являются концепты, связанные между собой отношениями гипонимии и гиперонимии. «Фреймы» – подграф с семантическими сценариями ситуаций, представленными в виде фреймов. Вершинами подграфа «Граммемы» являются грамматические категории и вершины для их классификации.

Подграфы типа «Лингвистические единицы» содержат вершины, соответствующие лингвистическим единицам разных языковых уровней: морфемы, словоформы, аналитические формы, предложения; каждый подграф соответствует

одному тюркскому языку. Ребра отражают структурные связи между этими единицами. На рисунке данный подграф разделен на две составляющие: «Потенциальные возможности» и «Реальное использование», реализующиеся в двух лингвистических ресурсах. «Потенциальные возможности» лингвистических единиц тюркских языков описаны в лингвистическом портале «Тюркская морфема», а «Реальное использование» в речи и тексте представлено в электронном корпусе «Туган тел», который содержит тексты тюркских языков.

Структура модели лингвистического графа знаний «TurkLang» для описания тюркских языков (вершины и связи) и то, из чего состоят подграфы рисунка 2, раскрыта на рис. 3. В центре этого графа изображен элемент «Морфема» (morpheme), который является основной лингвистической единицей графа знаний портала.

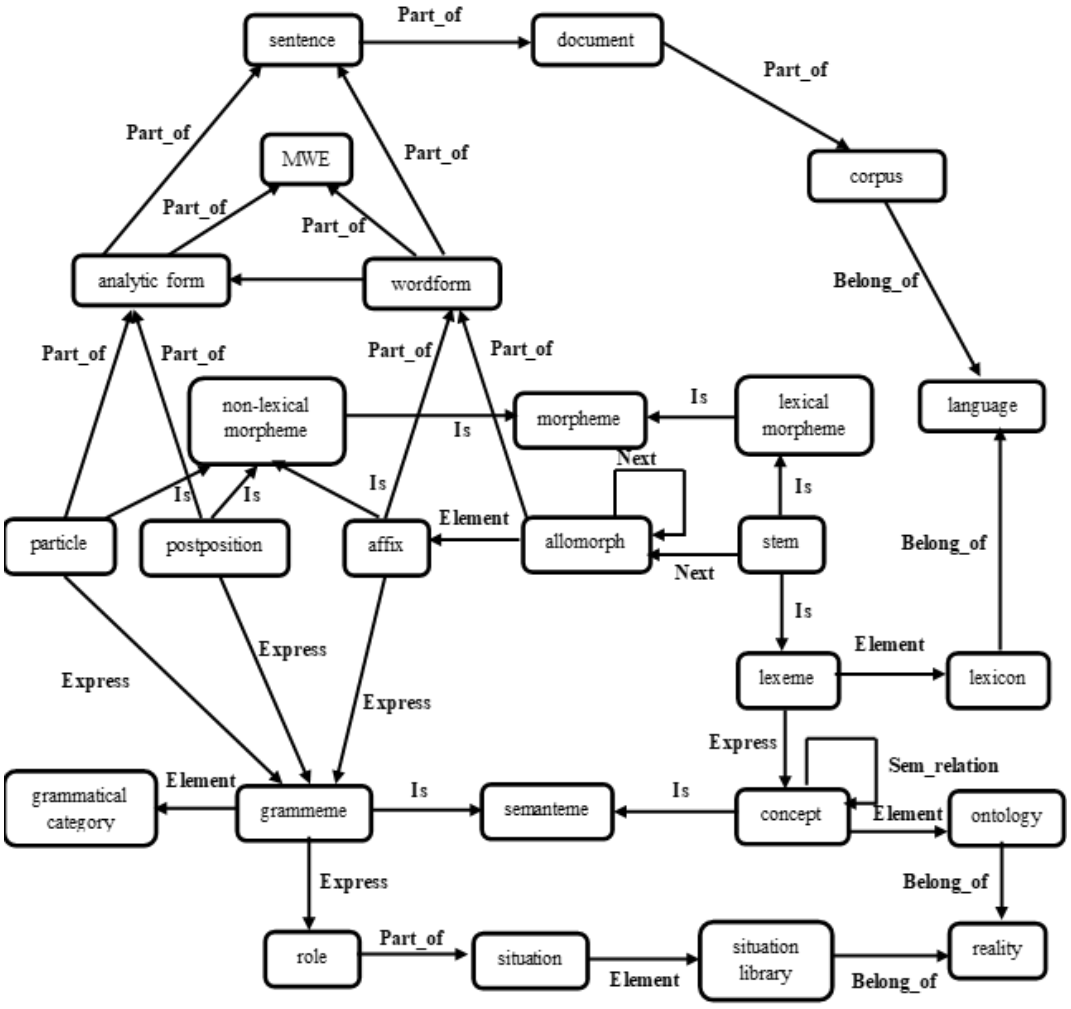


Рис. 3. Структура модели лингвистического графа знаний TurkLang

Вершины, обозначающие «Корпус» (corpus) и «Документ» (document), относятся к разделу графа с описанием реального использования языка. Элементы «Предложение» (sentence), «Морфема» (morpheme), «Корень» (stem), «Аффикс» (affix), «Частица» (particle), «Послелог» (postposition) и «Многословное выражение» (MWE) присутствуют как в разделах реального, так и потенциального описаний языка (см. рис. 2).

Элементы «Грамматическая категория» (grammatical category), «Граммема» (grammeme) и «Семантема» (semanteme) относятся к семантическим структурам грамматики. Элементы «Лексема» (lexeme), «Концепт» (concept) и «Онтология» (ontology) относятся к семантическим структурам тезауруса. Элементы «Ситуация» (situation) и «Роль» (role) относятся к семантическим структурам фреймов. Связь семантических структур с лингвистическими единицами, а также морфотактическая связь лингвистических единиц между собой выражают потенциальные возможности графа знаний по генерации новых текстовых описаний (разработке анализаторов и синтезаторов текстов на различных уровнях: морфологическом, синтаксическом и семантическом).

ИСПОЛЬЗОВАНИЕ ГРАФА ЗНАНИЙ «TURKLANG» ДЛЯ РАЗРАБОТКИ ИНСТРУМЕНТОВ ОБУЧЕНИЯ ТЮРКСКИМ ЯЗЫКАМ

В работе [1] содержится утверждение, что ученик, изучающий языки, должен учиться видеть языковые структуры, искать аналогии и обобщать полученные данные. Кроме того, использование аутентичных материалов делает речь обучающихся более идиоматичной, приближая ее к речи носителей языка. Мы считаем, что все эти возможности в полной мере реализованы в предложенной нами модели, на основе которой построены лингвистический портал «Тюркская морфема» и электронный корпус «Туган тел».

Рассмотрим примеры того, как из базы знаний, построенной на основе нашей модели, можно извлекать информацию о сравнительных особенностях тюркских языков и строить учебные задания для их изучения. Особенно эффективно это может работать для обучения студентов, уже знающих один тюркский язык, другому тюркскому языку на примерах рассмотрения разницы в ситуацион-

ных фреймах двух соответствующих друг другу предложений в языковой паре. Далее представлены примеры учебных заданий для турецко-татарской языковой пары.

1. В зависимости от ролевой схемы глагола выбирается вариант перевода.

o insanı vuruyor *'он убивает человека'* →
PN(o) N(insan)+**ACC(-yI) V(vur)**+PRES(-Iyor) →
PN(ул) N(кеше)+**ACC(-нЫ) V(үтер)**+PRES(-Й) →
ул кешене үтерә
o insana vuruyor *'он ударяет человека'* →
PN(o) N(insan)+**DIR(-yA**) **V(vur)**+PRES(-Iyor) →
PN(ул) N(кеше)+**DIR(-ГА) V(сук)**+PRES(-Й) →
ул кешегә суга

2. Разные ролевые схемы в разных языках.

o bunu Ayşeye sordu *'он спросил это у Айшы'* →
PN(o) N(bu)+ACC(-yI) N(Ayşe)+**DIR(-yA)** V(sor)+PST_DEF(-du) →
PN(ул) N(бу)+ACC(-нЫ) N(Әйшә)+**ABL(-ДАн)** V(сора)+PST_DEF(-ДЫ) →
ул моны Әйшәдән сорады
o işe başlıyor *'он начинает работу'* →
PN(o) N(iş)+**DIR(-yA)** V(başla)+PRES(-Iyor) →
PN(ул) N(эш)+**ACC(-нЫ)** V(башла)+PRES(-Й) →
ул эшне башлый

Такого рода задания могут быть сгенерированы с использованием примеров реальных предложений на том или ином языке из электронного корпуса, семантической разметки на основе ситуационных фреймов портала «Тюркская морфема», морфогенератора предложений портала для перевода и морфоанализатора портала для получения схемы разбора. Также ученику доступна информационно-справочная система портала, с помощью которой возможно более подробное изучение морфем, грамматики и семантики задания.

Ролевые схемы на основе ситуационных фреймов портала «Тюркская морфема» могут быть использованы не только при генерации заданий, но и для автоматизированной проверки ответа ученика. Для этого предлагается следующая реализация прагматически-ориентированного алгоритма автоматического анализа

ответа обучаемого с использованием фреймов, тезауруса и грамматики портала (в частности, морфотактики). Схема этого алгоритма представлена на рис. 4.

Рис. 4. Схема алгоритма анализа ответа

Рассмотрим алгоритм:

1. На первом этапе ответ на вопрос в виде текста на тюркском языке сначала проходит этап морфоанализа с помощью анализатора портала «Тюркская морфема».

2. Далее полученный морфологический разбор поступает на вход лексическому процессору, в котором производятся лексический анализ с использованием модели ответа (свой для каждого вопроса) и трансформация текста ответа в форму цепочки концептов из тезауруса портала (канонизированный ответ). Модель ответа своя для каждого вопроса, она определяет ожидаемый лексикон и семантику ответа, имея вид пар «Концепт – Множество корневых морфем, соответствующих ответу». Такая модель ответа может быть сгенерирована полностью в автоматическом режиме с использованием лингвистического ресурса портала «Тюркская морфема» и ситуационного фрейма (какого – указано далее).

3. На третьем этапе канонизированный ответ поступает на вход семантическому интерпретатору, который производит проверку соответствия цепочки концептов индивидуальной концептуальной грамматике ответа на основе ситуационного фрейма, ожидаемого в ответе для данного задания. В большинстве случаев это тот же ситуационный фрейм, с помощью которого это задание и было сгенерировано. На выходе получается числовой вектор, называемый вектором ситуации. Этот вектор должен позволять оценить правильность, точность и полноту ответа, содержать данные о соответствии ответа ожидаемому ситуационному фрейму, ожидаемому лексикону, о длине ответа, модальность и т. д.

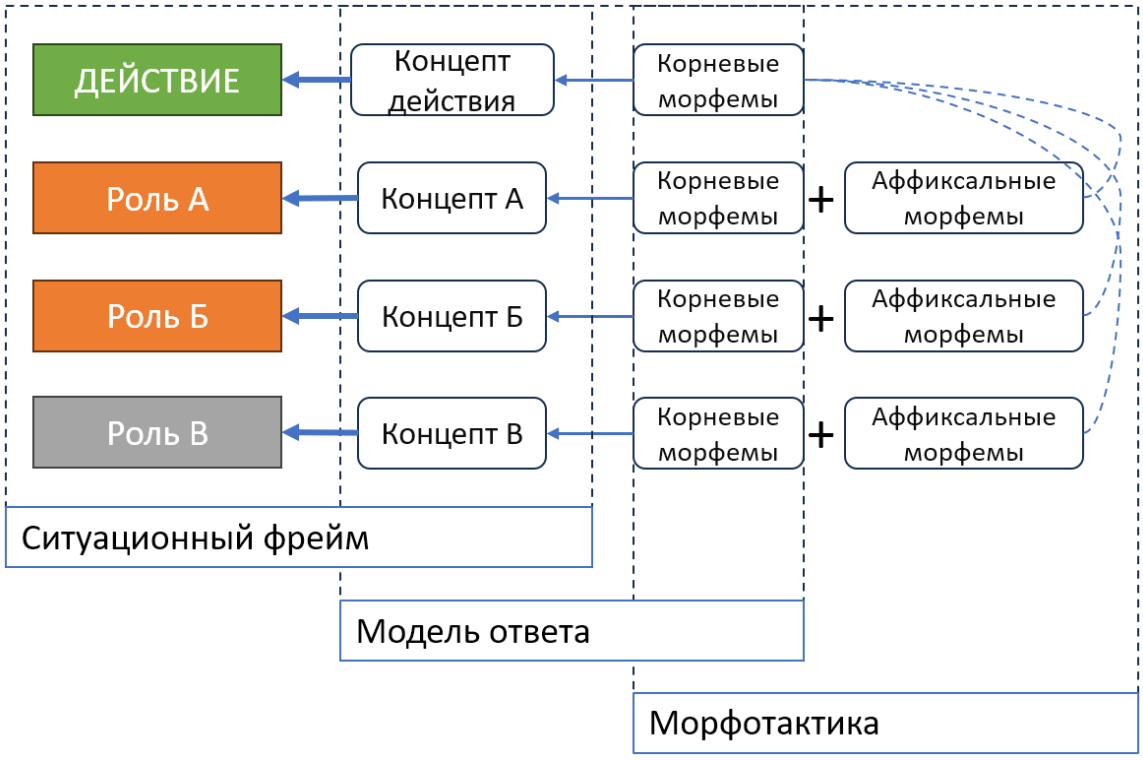


Рис. 5. Схема данных, извлекаемых из портала для анализа ответа

На рис. 5 представлена схема данных, извлекаемых из портала в процессе анализа. Ситуационный фрейм задается действием, определяющим ситуацию и роли объектов, участвующих в данной ситуации. Действию соответствует некоторый набор концептов действия, а ролям объектов – концепты объектов, которые могут выполнять данные роли. К перечисленным концептам могут также относиться атрибутивные единицы – концепты атрибутов действий и концепты атрибутов объектов – но они не обязательны к использованию в фрейме, значит, и в

ответе. Каждому концепту в некотором тюркском языке соответствует множество корневых морфем, и данные пары определяют модель ответа. С корневыми морфемами и между собой правилами морфотактики связаны аффиксальные морфемы. Кроме того, некоторые аффиксальные морфемы обязательны к использованию при реализации ситуационного фрейма в языке. Все эти элементы в полной мере определяют грамматику и семантику всевозможных вариаций правильных ответов (с учетом строго структурированного синтаксиса тюркских языков), что позволяет произвести анализ и предварительную оценку в виде вектора ситуации в автоматическом режиме.

Ранее реализация и оценка аналогичного алгоритма были представлены авторами в статье [4], однако в нем использовались иные языковые универсалии, недостаточно учитывающие особенности тюркских языков и не имеющие ресурсов для автоматической генерации. Представленный здесь алгоритм в полной мере использует ресурс портала «Тюркская морфема» для автоматического анализа ответа. При этом для генерации заданий может быть использован ресурс электронного корпуса «Туган тел» и иных электронных корпусов тюркских языков.

ЗАКЛЮЧЕНИЕ

Представленная модель лингвистического графа «Turklang» находит свою реализацию в разработанных ранее и разрабатываемых на данный момент лингвистических ресурсах и инструментах обработки естественного языка, таких как портал «Тюркская морфема» и электронный корпус тюркских языков «Туган тел». Данный лингвистический граф позволяет наиболее полно представить лингвистическую информацию для тюркских языков, с учетом их структурно-функциональных особенностей, на всех языковых уровнях: грамматика (морфология и синтаксис) и семантика (тезаурус и ситуационные фреймы) как в их потенциальных возможностях, так и в реальном использовании в текстах и речи. За счет этого возможны реализация ресурсов для электронного образования, учебных курсов с использованием информационно-справочной системы, а также разработка автоматических генераторов учебных заданий, внедренных в данные ресурсы и курсы, что является дальнейшей задачей, стоящей перед авторами статьи.

СПИСОК ЛИТЕРАТУРЫ

1. *Johns T.* Should you be persuaded. Two samples of data-driven learning materials // T. Johns & P. King (Eds.). Classroom Concordancing. ELR Journal. 1991. № 4. P. 1–16.

2. *Левенкова А.Ю., Трифонова И.С.* Базы данных в лингвистике и языковом образовании: современное состояние и возможности их использования при обучении иностранному языку // Известия ВГПУ. 2023. №2 (175). С. 90–101.

3. *Basile P., Cassotti P., Ferilli S., McGillivray B.* A New Time-sensitive Model of Linguistic Knowledge for Graph Databases // Proc. of the 1st Workshop on Artificial Intelligence for Cultural Heritage co-located with the 21st International Conference of the Italian Association for Artificial Intelligence (AixIA 2022), CEUR Workshop Proceedings. 2022. V. 3286. P. 69–80.

4. *Suleymanov D., Prokopyev N.* Development of Prototype of Natural Language Answer Processor for e-Learning // Kuznetsov S.O., Panov A.I., Yakovlev K.S. (Eds.). Artificial Intelligence. RCAI 2020. Lecture Notes in Computer Science. 2020. V. 12412. P. 448–459.

LINGUISTIC KNOWLEDGE GRAPH “TURKLANG” FOR CREATION OF TOOLS FOR TEACHING TURKIC LANGUAGES

A. R. Gatiatullin[1] [0000-0003-3063-8147], N. A. Prokopyev[2] [0000-0003-0066-7465]

[1]*Tatarstan Academy of Sciences;* [2]*Kazan Federal University*

[1]ayrat.gatiatullin@gmail.com, [2]nikolai.prokopyev@gmail.com

Abstract

This article presents elements of the linguistic knowledge graph “Turklang”, developed at the Institute of Applied Semiotics of the Academy of Sciences of Tatarstan and used as a basis for creating a number of linguistic resources and tools: the portal “Turkic Morpheme”, the electronic corpus of the Tatar language “Tugan Tel”, morpho-analyzer. Creating an educational environment requires subject-oriented knowledge graphs, for which methods of general and open graphs are not suitable. This paper

describes linguistic knowledge graphs, which reflect, on the one hand, potential capabilities of Turkic languages, and on the other hand, examples of actual use in texts. Peculiarity of these knowledge graphs is that they contain linguistic units of different linguistic levels, and concepts corresponding to meanings of these linguistic units, which are built into the thesaurus of concepts. Structure of this knowledge graph allows to formulate the content of a training course, build an individual educational trajectory, as well as create tests and tools of automated answer grading as part of knowledge control when teaching Turkic languages. This makes it possible to subsequently develop, based on these graphs, training programs taking into account the structural and functional features of the Turkic languages, and also contributes to the implementation of individual goals of students.

Keywords*: knowledge graph, knowledge base, linguistic resource, linguistic unit, low-resource languages, Turkic languages, web portal, e-learning, knowledge control, automated answer grading*

REFERENCES

1. *Johns T.* Should you be persuaded. Two samples of data-driven learning materials // T. Johns & P. King (Eds). Classroom Concordancing. ELR Journal. 1991. № 4. P. 1–16.

2. *Levenkova A.Yu., Trifonova I.S.* Bazy dannykh v lingvistike i yazykovom obrazovanii: sovremennoe sostoyanie i vozmozhnosti ikh ispol'zovaniya pri obuchenii inostrannomu yazyku // Izvestiya VGPU. 2023. № 2 (175). P. 90–101.

3. *Basile P., Cassotti P., Ferilli S., McGillivray B.* A New Time-sensitive Model of Linguistic Knowledge for Graph Databases // Proceedings of the 1st Workshop on Artificial Intelligence for Cultural Heritage co-located with the 21st International Conference of the Italian Association for Artificial Intelligence (AixIA 2022), CEUR Workshop Proceedings. 2022. V. 3286. P. 69–80.

4. *Suleymanov D., Prokopyev N.* Development of Prototype of Natural Language Answer Processor for e-Learning // Kuznetsov S.O., Panov A.I., Yakovlev K.S. (Eds). Artificial Intelligence. RCAI 2020. Lecture Notes in Computer Science. 2020. V. 12412. P. 448–459.

СВЕДЕНИЯ ОБ АВТОРАХ

ГАТИАТУЛЛИН Айрат Рафизович – 1972 г. рождения. Окончил Казанский государственный университет в 1994 г., к. т. н. (2002). Ведущий научный сотрудник Института прикладной семиотики Академии наук Республики Татарстан. В списке научных трудов более 60 работ.

Ayrat Rafizovich GATIATULLIN – born in 1972. Graduated from Kazan State University in 1994, candidate in technical sciences (2002). Leading researcher at the Institute of Applied Semiotics of the Academy of Sciences of the Republic of Tatarstan. List of scientific works includes more than 60 publications.

email: ayrat.gatiatullin@gmail.com
ORCID: 0000-0003-3063-8147;
Author ID (РИНЦ): 161758;
Author ID (Scopus): 56500678000.

ПРОКОПЬЕВ Николай Аркадиевич – 1992 г. рождения. Окончил Институт вычислительной математики и информационных технологий Казанского Федерального университета в 2015 году. Научный сотрудник Института прикладной семиотики Академии наук РТ. В списке научных трудов более 40 работ.

Nikolai Arkadievich PROKOPYEV – born in 1992. Graduated from the Institute of Computational Mathematics and Information Technologies of the Kazan Federal University in 2015. Researcher at the Institute of Applied Semiotics of the Academy of Sciences of the Republic of Tatarstan. List of scientific works includes more than 40 publications.

email: nikolai.prokopyev@gmail.com
ORCID: 0000-0003-0066-7465;
Author ID (РИНЦ): 999214;
Author ID (Scopus): 57190803409;
Researcher ID (WoS): S-3829-2016.

Материал поступил в редакцию 12 мая 2024 года

УДК 519.688

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ОБУЧАЮЩЕГО БЛОКЧЕЙН-СИМУЛЯТОРА

О. М. Меховников[1] [0009-0008-5247-7341], А. С. Тощев[2] [0000-0003-4424-6822]

[1, 2] *Институт информационных технологий и интеллектуальных систем, Казанский (Приволжский) федеральный университет, ул. Кремлевская, 35, г. Казань, Республика Татарстан 420008*

[1]oleg_mekhovnikov@mail.ru, [2]atoschev@kpfu.ru

Аннотация

Представлен блокчейн-симулятор, предназначенный для обучения студентов и начинающих блокчейн-разработчиков. Симулятор создан с целью предоставить пользователям интуитивно понятное и доступное средство для изучения основных концепций и механизмов функционирования блокчейна. Рассмотрены основные аспекты проектирования и архитектуры симулятора, а также представлена демонстрация работы приложения. Разработанный симулятор способствует привлечению новых специалистов в сферу блокчейн-разработки.

***Ключевые слова**: блокчейн, блокчейн-симулятор, введение в блокчейн*

ВВЕДЕНИЕ

Актуальность работы продиктована ростом популярности технологии блокчейн и, как следствие, потребностью в подготовке специалистов в этой области. В ходе исследования было выявлено, что существующие симуляторы, а именно, Bitcoin Simulator [1], BlockSim: Faria [2], SimBlock [3], BlockSim: Alharby [4] и VIBES [5], мало пригодны для целей обучения, так как работают по принципу «черного ящика» и не дают возможности просматривать данные отдельно взятого блока, транзакции и те данные, которые временно хранятся на узле до включения в блокчейн (например, транзакции из пула узла). Таким образом, возникает потребность в создании нового симулятора, который будет обеспечивать

прозрачность потока данных и наглядно демонстрировать принципы работы блокчейн-сети.

АРХИТЕКТУРА БЛОКЧЕЙН-СИМУЛЯТОРА

Архитектура симулятора представляет собой многослойную структуру модулей (рис. 1). Рассмотрим каждый из представленных слоев.

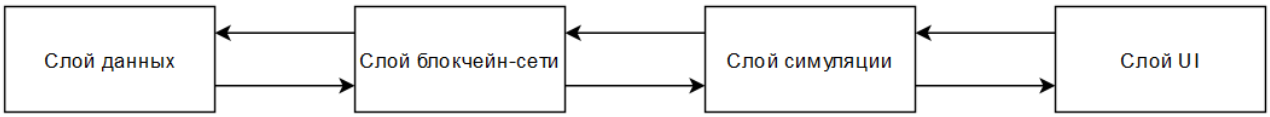


Рис. 1. Поток данных в многослойной архитектуре

Слой данных инкапсулирует модель блокчейна. В целях достижения наибольшей прозрачности работы системы данный слой был несколько упрощен в сравнении с Bitcoin [6]. В частности, были вырезаны такие механизмы, как деревья Меркла [7], упрощенная проверка платежей [6] и UTXO [6]. В этих же целях были упрощены структуры данных блокчейна. Для сравнения: диаграмма классов блокчейна, построенная по данным реального блока Bitcoin [8], представлена на рис. 2.

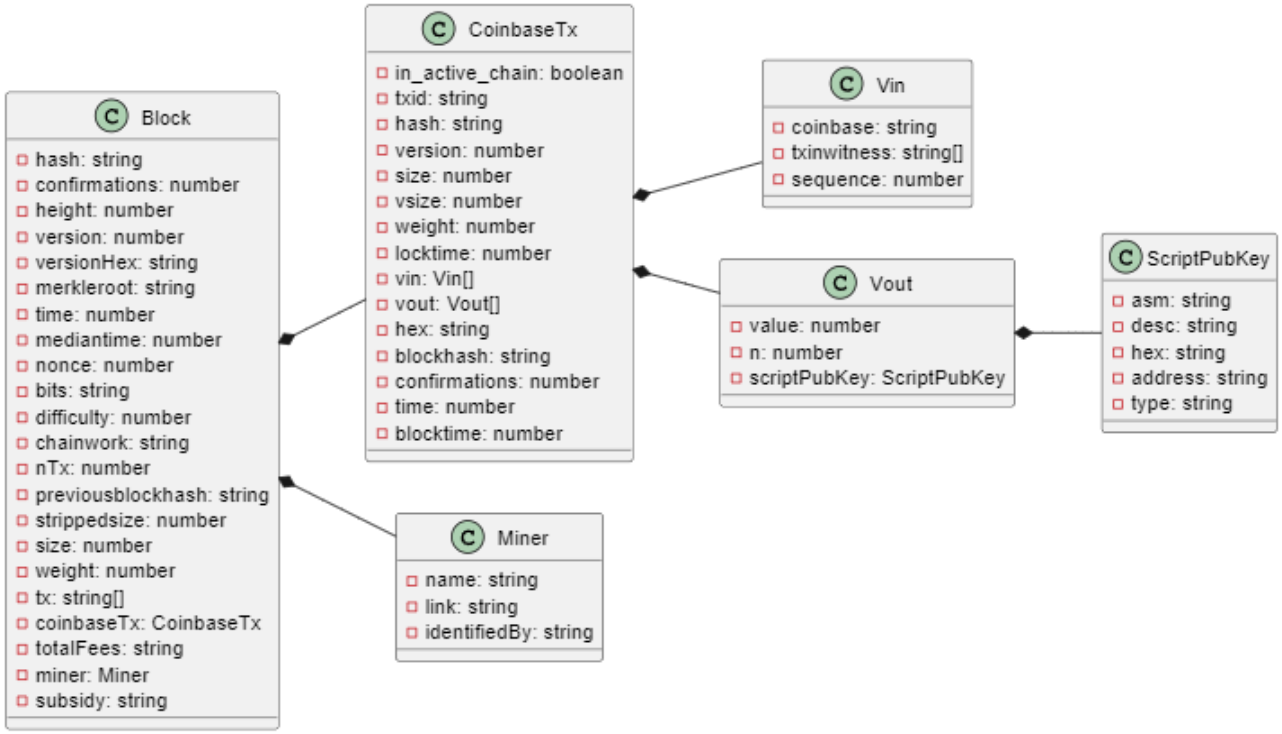


Рис. 2. Диаграмма классов блокчейна Bitcoin

Диаграмма классов блокчейна разработанного симулятора представлена на рис. 3.

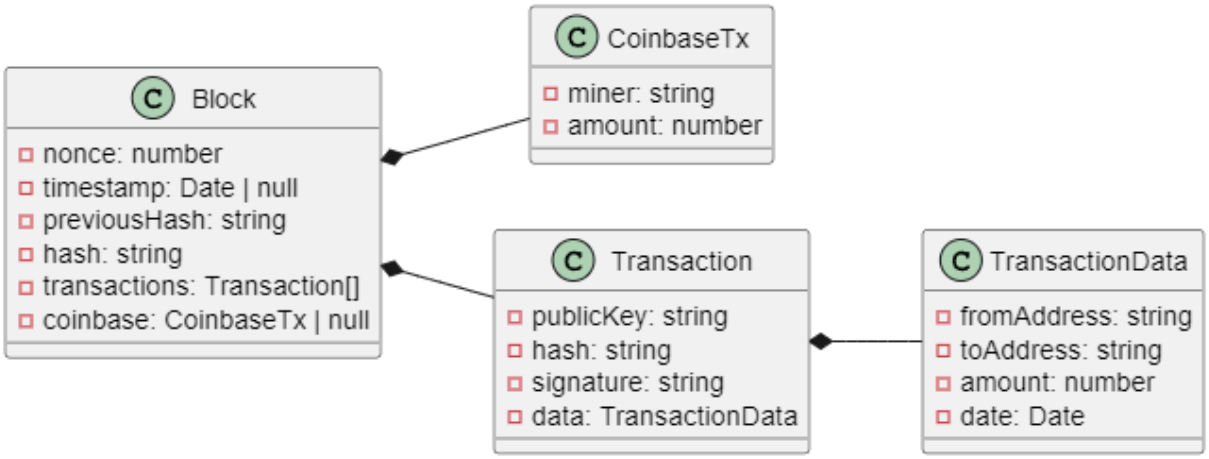


Рис. 3. Диаграмма классов блокчейна

Слой блокчейн-сети инкапсулирует логику работы сети, аналогичной Bitcoin с незначительными изменениями:

- Новые транзакции рассылаются всем узлам.
- Каждый узел включает полученные транзакции в блок.
- Узлы выбирают случайный nonce и находят хэш блока.
- Как только узлу удается найти подходящий хэш, он отправляет блок в сеть.
- Узлы принимают блоки только в том случае, если все транзакции в них корректны и не повторяются.
- Узлы выражают согласие с новыми данными, начиная работу над следующим блоком и используя хэш предыдущего блока в качестве новых исходных данных.

Слой симуляции отвечает за моделирование процесса обмена транзакциями между участниками сети. Симуляция реализована по принципу дискретно-событийного моделирования: генератор случайных чисел возвращает число, определяющее следующее событие, которое будет обработано системой (рис. 4).

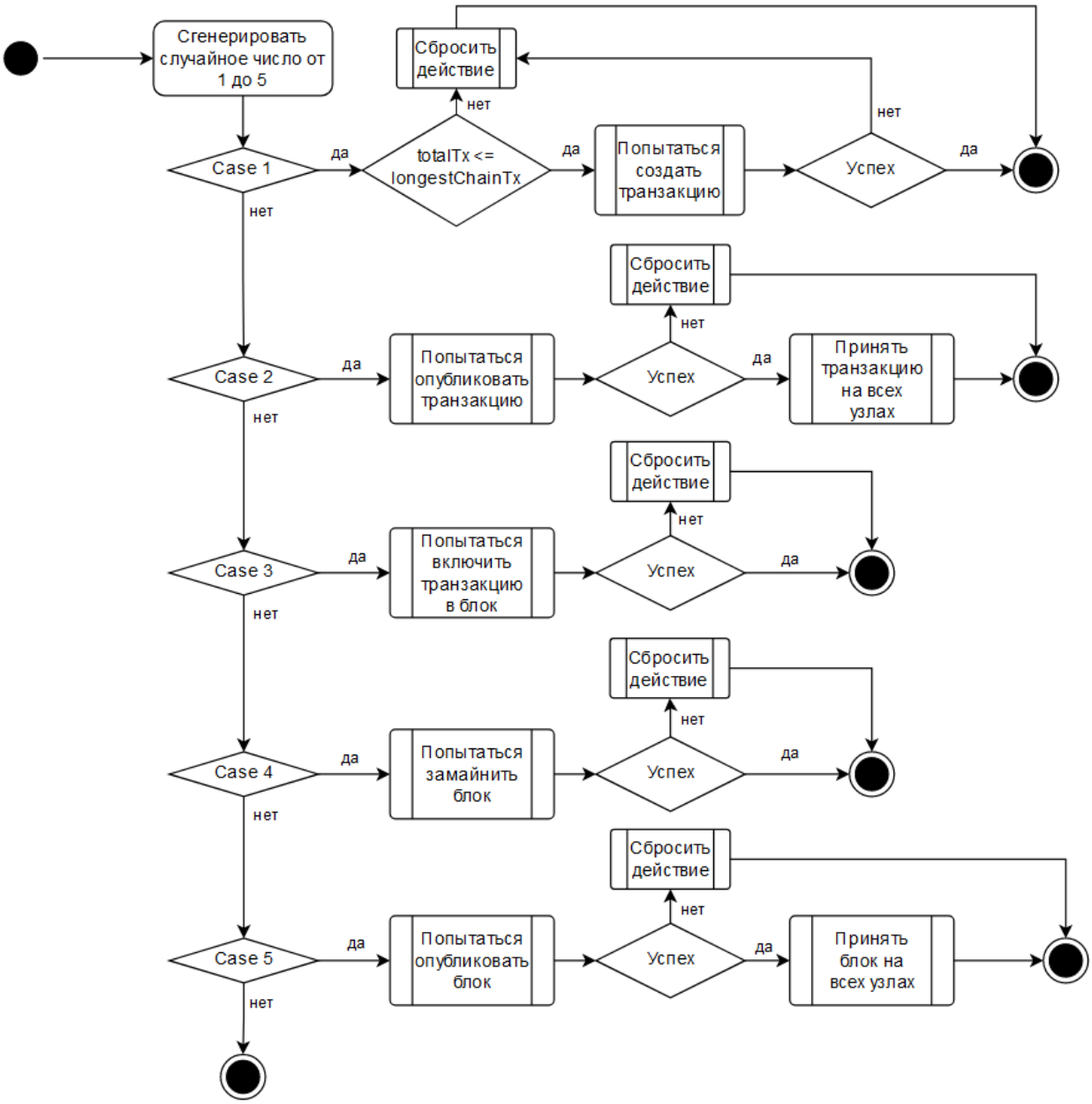


Рис. 4. Блок-схема алгоритма симуляции

totalTx – общее число совершенных транзакций.

longestChainTx – число транзакций в самом длинном блокчейне.

Слой UI включает в себя такие компоненты, как сетевая диаграмма, лог операций, список узлов, миниатюра блокчейна, модальные окна блока и узла и т. д.

ДЕМОНСТРАЦИЯ РАБОТЫ БЛОКЧЕЙН-СИМУЛЯТОРА

Программа принимает на вход файл в формате JSON, описывающий массив узлов. В текущем примере блокчейн каждого узла представлен одним блоком, который содержит единственную coinbase-транзакцию, которая начисляет 10000 BTC на адрес, связанный с данным узлом (листинг 1).

```
{
        "id": 1,
        "name": "Узел 1",
        "publicKey":
"04ebe3074d1fe1c97a1385a2644856085815d592b4f7db1eb71e3faff6f64694c2d97815c8cf5c5
b6979083e487231fbb3bd6968f62d09c81056ea2fc9ec73ffd2",
        "privateKey":
"3503c1dbc9712c3f7cb22b167479c371770ab60ece3b4f7c22994be33187d24b",
        "blockchain": {
            "chain": [
                {
                    "nonce": 2083236893,
                    "previousHash":
"0000000000000000000000000000000000000000000000000000000000000000",
                    "hash":
"3f5b126e860591a11402e6899ea5a5e88343df767aa5800168a10f6b4df31ec1",
                    "transactions": [],
                    "coinbase": {
                        "miner":
"04ebe3074d1fe1c97a1385a2644856085815d592b4f7db1eb71e3faff6f64694c2d97815c8cf5c5
b6979083e487231fbb3bd6968f62d09c81056ea2fc9ec73ffd2",
                        "amount": 10000
                    }
                }
            ]
        },
        "transactionPool": [],
        "newBlock": null,
        "newTransaction": null
    }
```

Листинг 1. Входные данные для Узла 1.

В текущей реализации параметры блокчейна устанавливаются через программный код. Для данного примера была задана следующая конфигурация:

- Количество транзакций в блоке = 3.
- Вознаграждение за майнинг = 10 BTC.

Рассмотрим процесс работы симулятора. После завершения загрузки приложения и чтения входных данных из файла главный экран программы выглядит следующим образом (рис. 5.)

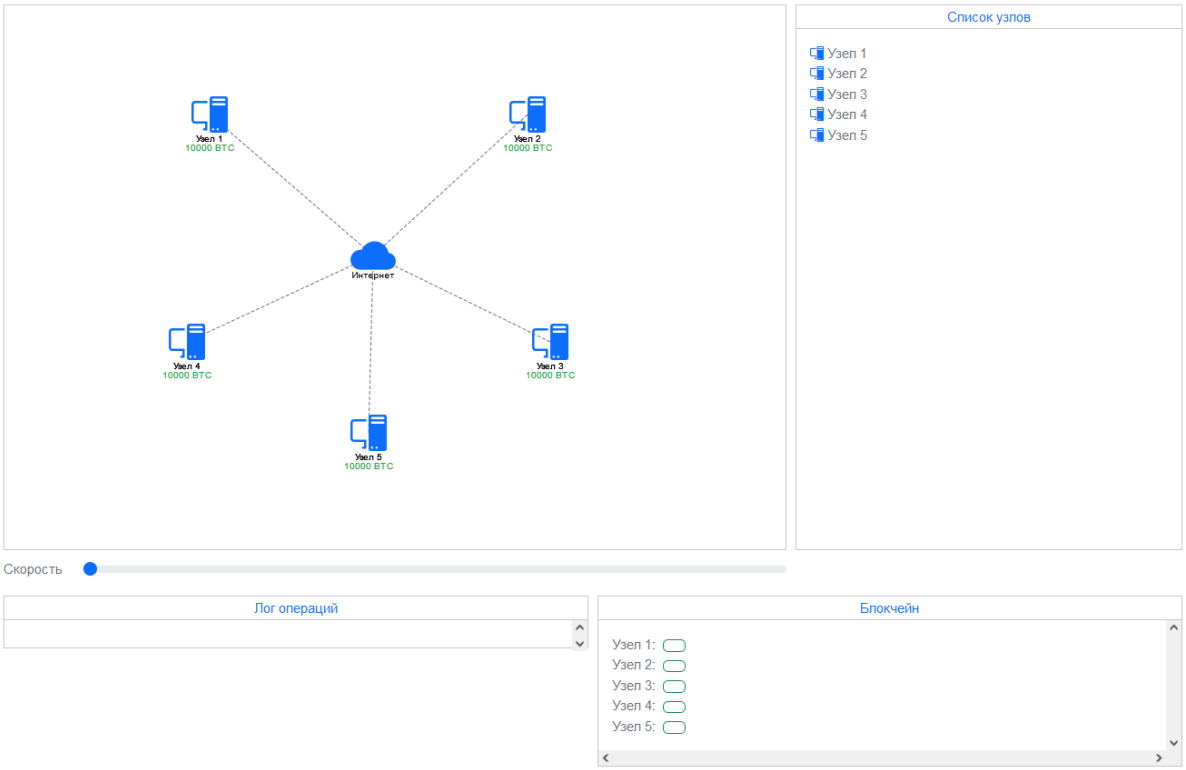


Рис. 5. Главный экран программы после загрузки входных данных

Далее узлы начинают обмениваться криптовалютными транзакциями и генерировать блоки. Сгенерированные блоки отображаются в компоненте Блокчейн (рис. 6).

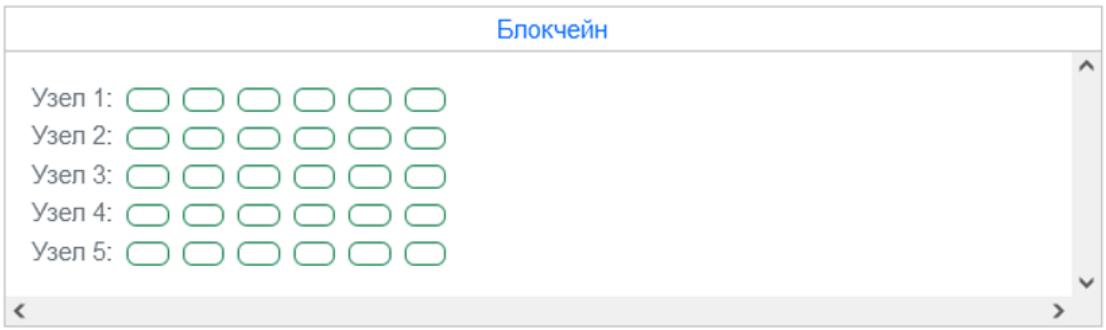


Рис. 6. Блокчейн после генерации 5 блоков

Данный компонент отображает схему копий блокчейна, хранящихся на каждом узле. Симулятор предоставляет возможность просмотра данных каждого блока, включая его транзакции, при выборе соответствующего блока в блокчейне (рис. 7).

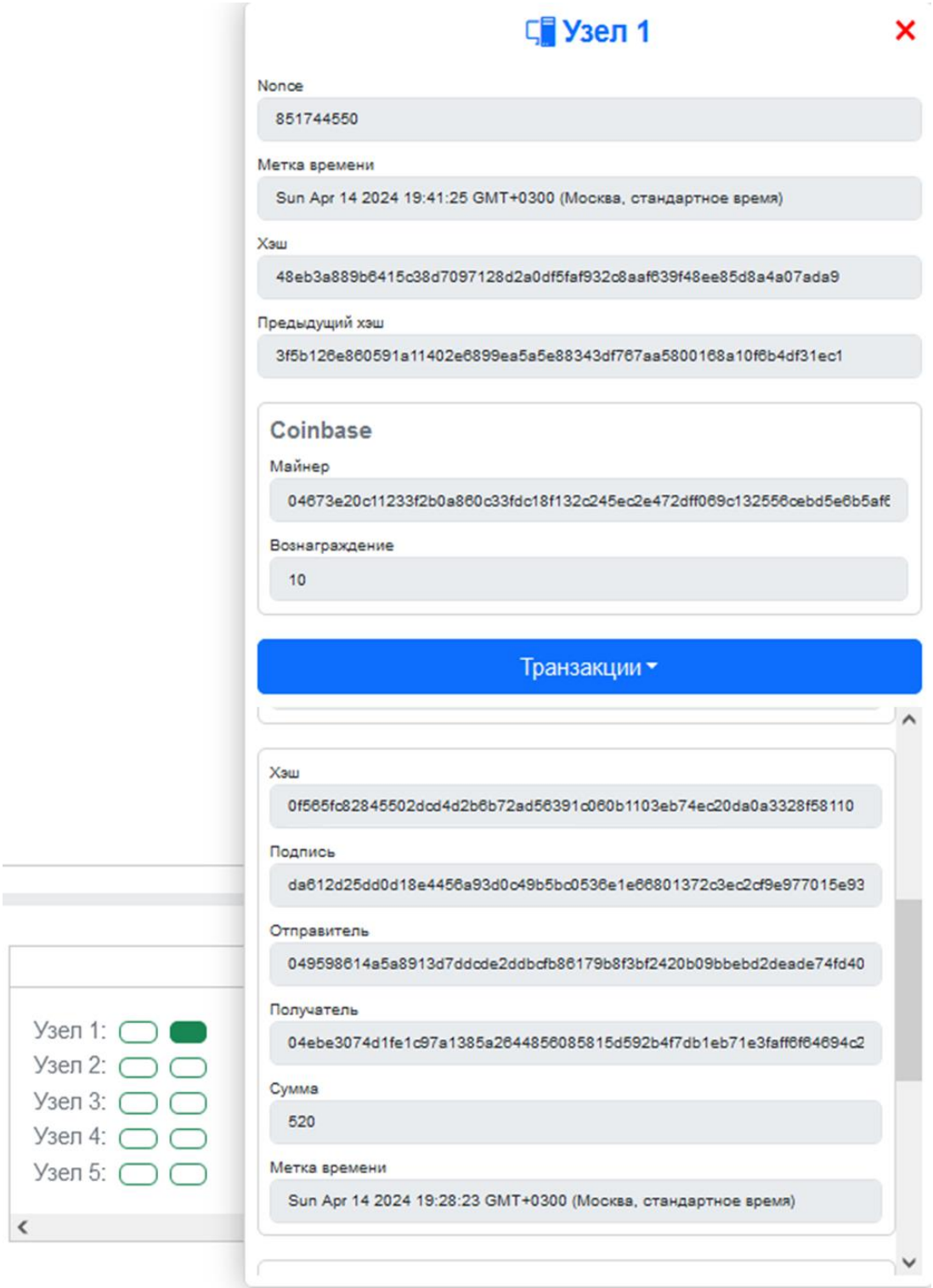


Рис. 7. Просмотр блока

В ходе симуляции через равные промежутки времени происходит одно из событий: *Транзакция создана, Транзакция распространяется по сети, Транзакция попадает в пул узла, Транзакция попадает в блок, Новый блок сгенерирован, Новый блок распространяется по сети, Новый блок добавляется к блокчейну узла, Новый блок отброшен узлом.* Для пользовательского вывода событий предусмотрены два компонента: Лог операций и Сетевая диаграмма.

В лог операций отображается информация о 1000 последних событий, произошедших в системе (рис. 8).

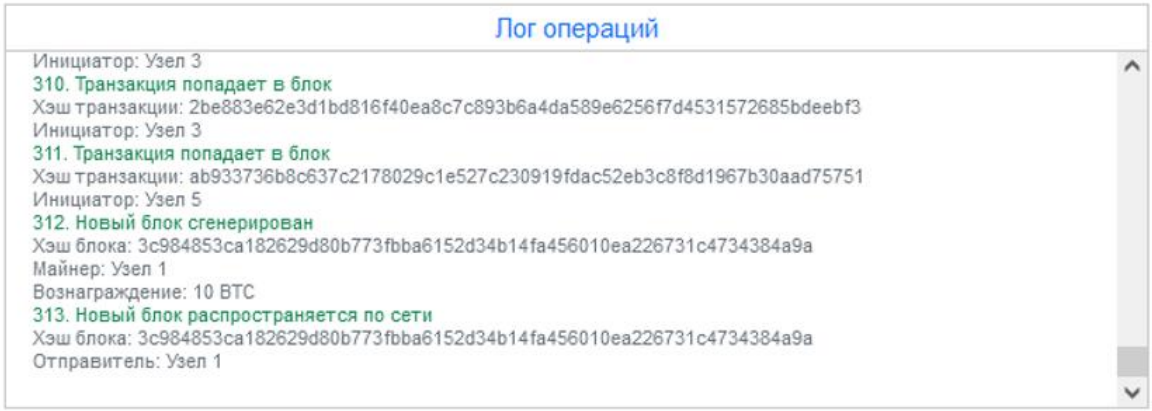


Рис. 8. Лог операций после генерации 5 блоков

Сетевая диаграмма служит для анимированного отображения событий, которые происходят в системе в данный момент (рис. 9).

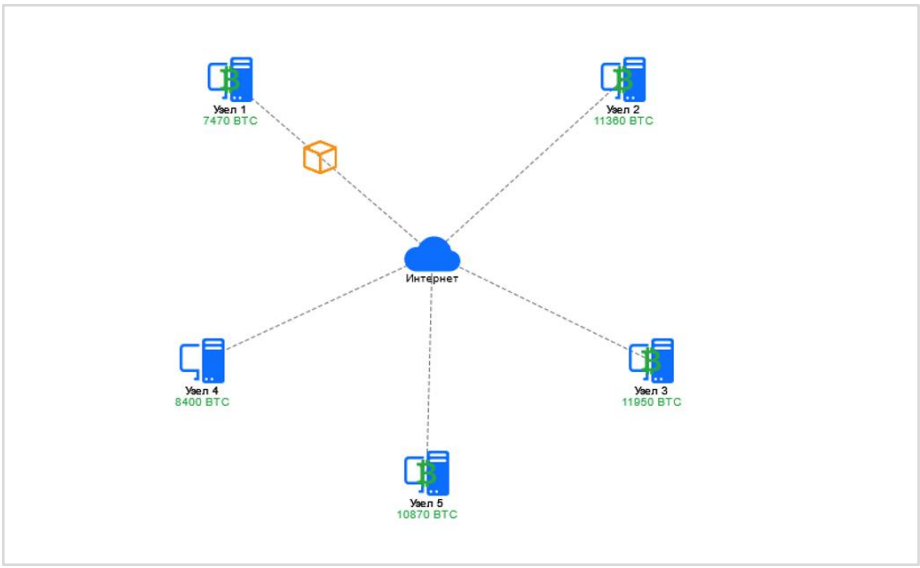


Рис. 9. Сетевая диаграмма после генерации 5 блоков

К примеру, на данном снимке на сетевой диаграмме отображены два события: *Новый блок распространяется по сети, Транзакция попадает в пул узла.*

При наступлении событий также в реальном времени обновляются следующие данные в соответствующих компонентах пользовательского интерфейса: *Блоки блокчейна, Транзакции блокчейна, Текущая неопубликованная транзакция узла, Пул транзакций узла, Текущий неопубликованный генерирующийся блок узла.*

Рассмотрим результаты симуляции: в ходе работы приложения было сгенерировано 15 транзакций, которые были включены в 5 новых блоков. Анализ выходных данных показал, что получившийся на выходе снимок состояния блокчейн-сети является полностью валидным.

ЗАКЛЮЧЕНИЕ

В результате исследования спроектирован и разработан обучающий блокчейн-симулятор, который включает в себя основные компоненты и механизмы реального блокчейна. Симулятор позволяет пользователям ознакомиться с такими ключевыми аспектами технологии блокчейн, как создание транзакций и формирование блоков.

Эксперимент показал, что разработанный симулятор способен не только корректно моделировать блокчейн-сеть, но и в удобном виде предоставлять информацию о событиях, происходящих в системе.

Существенным преимуществом симулятора является то, что его может запустить в браузере любой желающий. Симулятор может быть использован в качестве учебного материала при обучении технологии блокчейн. Например, его можно демонстрировать студентам высших учебных заведений при освоении соответствующих образовательных программ.

СПИСОК ЛИТЕРАТУРЫ

1. *Gervais A., Karame G.O., Wüst K., Glykantzis V., Ritzdorf H., Capkun S.* On the security and performance of proof of work blockchains // Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016. P. 3–16.

2. *Faria C., Correia M.* BlockSim: blockchain simulator // 2019 IEEE International Conference on Blockchain (Blockchain). IEEE, 2019. P. 439–446.

3. *Aoki Y., Otsuki K., Kaneko T., Banno R., Shudo K.* Simblock: A blockchain network simulator // IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2019. P. 325–329.

4. *Alharby M., Van Moorsel A.* Blocksim: a simulation framework for blockchain systems // ACM SIGMETRICS Performance Evaluation Review. 2019. Vol. 46, No. 3. P. 135–138.

5. *Stoykov L., Zhang K., Jacobsen H.-A.* Vibes: fast blockchain simulations for large-scale peer-to-peer networks // Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference: Posters and Demos. 2017. P. 19–20.

6. *Nakamoto S.* Bitcoin: A peer-to-peer electronic cash system. URL: http://www.bitcoin.org/bitcoin.pdf.

7. *Mercle R.C.* Protocols for public key cryptosystems // Proc. 1980 Symposium on Security and Privacy, IEEE Computer Society. 1980. P. 122–133.

8. Block #838,614 | bitcoinexplorer.org.
URL: https://bitcoinexplorer.org/block-height/838614.

DESIGN AND DEVELOPMENT OF A TRAINING BLOCKCHAIN SIMULATOR

Oleg Mekhovnikov[1] [0009-0008-5247-7341], Alexander Toschev[2] [0000-0003-4424-6822]

[1, 2] *Institute of Information Technologies and Intelligent Systems, KFU University, st. Kremlin, 35, Kazan, Republic of Tatarstan 420008*

[1]oleg_mekhovnikov@mail.ru, [2]atoschev@kpfu.ru

Abstract

This article presents an educational blockchain simulator intended for training students and beginning blockchain developers. The simulator was created to provide users with an intuitive and accessible tool for learning the basic concepts and mechanisms of blockchain functioning. The article discusses the main aspects of the design and architecture of the simulator, and also provides a demonstration of the applica-

tion. In addition, the possibilities for further development of the simulator and its potential as a tool for teaching and research in the field of blockchain technologies are discussed. The resulting simulator contributes to the field of education and science, helping to increase the level of competence of specialists and the development of innovative solutions in the field of blockchain.

Keywords*: blockchain, blockchain simulator, introduction to blockchain*

REFERENCES

1. *Gervais A., Karame G.O., Wüst K., Glykantzis V., Ritzdorf H., Capkun S.* On the security and performance of proof of work blockchains // Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016. P. 3–16.

2. *Faria C., Correia M.* BlockSim: blockchain simulator // 2019 IEEE International Conference on Blockchain (Blockchain). IEEE, 2019. P. 439–446.

3. *Aoki Y., Otsuki K., Kaneko T., Banno R., Shudo K.* Simblock: A blockchain network simulator // IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2019. P. 325–329.

4. *Alharby M., Van Moorsel A.* Blocksim: a simulation framework for blockchain systems // ACM SIGMETRICS Performance Evaluation Review. 2019. Vol. 46, No. 3. P. 135–138.

5. *Stoykov L., Zhang K., Jacobsen H.-A.* Vibes: fast blockchain simulations for large-scale peer-to-peer networks // Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference: Posters and Demos. 2017. P. 19–20.

6. *Nakamoto S.* Bitcoin: A peer-to-peer electronic cash system.
URL: http://www.bitcoin.org/bitcoin.pdf

7. *Mercle R.C.* Protocols for public key cryptosystems // Proc. 1980 Symposium on Security and Privacy, IEEE Computer Society. 1980. P. 122–133.

8. Block #838,614 | bitcoinexplorer.org.
URL: https://bitcoinexplorer.org/block-height/838614

СВЕДЕНИЯ ОБ АВТОРАХ

МЕХОВНИКОВ Олег Максимович – разработчик ПО, студент магистратуры, исследователь. Казанский федеральный университет, Казань, Россия.

Oleg Maksimovich MEKHOVNIKOV – software developer, master's student, researcher. Kazan Federal University, Kazan, Russia.

email: oleg_mekhovnikov@mail.ru
ORCID: 0009-0008-5247-7341

ТОЩЕВ Александр Сергеевич – доцент, к. н. , КФУ, Институт информационных технологий и интеллектуальных систем, Кафедра программной инженерии, г. Казань.

Alexander Sergeevich TOSCHEV – Associate Professor, Ph.D., KFU, Institute of Information Technologies and Intelligent Systems, Department of Software Engineering, Kazan.

email: atoschev@kpfu.ru
ORCID: 0000-0003-4424-6822

Материал поступил в редакцию 13 мая 2024 года

УДК 004.02, 378.146

АВТОМАТИЗИРОВАННОЕ ОЦЕНИВАНИЕ КОРОТКИХ ОТВЕТОВ ОБУЧАЮЩИХСЯ С ИСПОЛЬЗОВАНИЕМ ЯЗЫКОВЫХ МОДЕЛЕЙ

Ч. Б. Миннегалиева[1] [0000-0003-4648-1623], И. И. Кашапов[2] [0009-0004-6341-8380], О. Д. Морозова[3] [0009-0009-5670-561X]

[1–3]*Институт вычислительной математики и информационных технологий, Казанский федеральный университет, Казань*

[1]mchulpan@gmail.com, [2]k.i.i.08@yandex.ru, [3]olka30.olka30@mail.ru

Аннотация

Методы проверки ответов обучающихся с использованием языковых моделей в настоящее время исследуются разными специалистами. Результаты автоматизированного оценивания зависят от предметной области и особенностей учебной дисциплины. В работе проанализированы ответы студентов, полученные в ходе изучения курса «Компьютерная графика и дизайн». При помощи языковых моделей определены векторы документов. Предложен метод оценивания ответов через нахождение косинусного сходства полученных векторов и уточнение оценок проверкой ключевых слов. Результаты могут использоваться при предварительной проверке ответов студентов и являются базой для дальнейших исследований.

Ключевые слова: языковая модель, контроль знаний, обработка текста, ключевое слово ответа, автоматизированная оценка ответов обучающихся, косинусное сходство, векторное представление документа, BERT, word2vec, открытый вопрос

ВВЕДЕНИЕ

Проверка и оценка знаний, умений, навыков студентов являются обязательной составляющей обучения, процессом обратной связи [1]. От результатов и условий проведения контроля знаний зависят качество образовательного процесса и степень удовлетворенности обучающихся. Составление заданий для проверочных работ, тестирований и экзаменов всегда являлось достаточно трудоём-

кой работой для учителей и преподавателей [2]. Вопросы объективного оценивания заданий, выполненных обучающимися, и процедура проведения контрольных мероприятий также продолжают обсуждаться участниками образовательного процесса. С развитием онлайн-образования контроль знаний остаётся одной из наиболее острых и дискуссионных проблем организации обучения [3], так как в данном случае проверка знаний, умений и навыков в большинстве случаев проводится дистанционно. Увеличивается доля самостоятельной работы обучающихся, и, как правило, возрастает количество студентов, обучающихся у одного преподавателя. В настоящее время специалистами изучаются различные подходы к автоматизации составления заданий и оцениванию выполненных работ. Разработан инструмент анализа проектной работы, позволяющий преподавателю выставлять оценки, анализируя собранные данные о работе студентов [4]. Программный комплекс (система контроля знаний), использующий конструктивно-выборочный метод, позволяет на практике применять новый тип заданий для автоматизированного контроля знаний и навыков по программированию [5]. Предлагаются новые подходы, основанные на формализации знаний. Например, система, базой которой является онтология OntoMath^{Edu}, предназначена для решения ряда задач, в том числе для автоматического тестирования знаний [6].

Во время тестирования знаний применяются различные формы заданий. На данном этапе развития технологий платформы онлайн-обучения предоставляют возможность автоматической проверки ответов на такие вопросы, которые требуют выбора одного или нескольких верных из множества имеющихся, сопоставления, ввода числового ответа, заполнения пропусков и другие. При выполнении заданий открытого типа обучающиеся самостоятельно формулируют ответ. Включение в тесты таких заданий повышает объективность контроля знаний, так как в этом случае не предъявляются варианты верных и неверных ответов и снижается вероятность угадывания или подбора верного варианта.

В настоящей работе рассмотрены подходы к автоматизированному оцениванию коротких ответов. Такие ответы формулируются на естественном языке, их длина может изменяться от одного предложения до одного абзаца.

Специалисты начали исследовать проблему автоматизированного оценива-

ния коротких ответов более 50 лет назад [7]. Сегодня изучается возможность получения такой оценки при помощи языковых моделей, которые в последние годы используются для решения разных задач, в том числе в образовании [8, 9]. Рассмотрены подходы к автоматизации решения текстовых математических задач с использованием моделей [10]. Предложена методика оценки и анализа формулировок результатов обучения. В качестве исходных данных выступают формулировки результатов обучения, между которыми вычисляется семантическое расстояние. Трансформация исходного текста в вектор происходит с помощью языковой модели [11]. Представлено исследование задачи автоматической классификации коротких связных текстов (эссе) на английском языке [12]. Авторы этой работы подробно рассмотрели оценочные метрики для задач автоматического оценивания коротких ответов, учитывая специфику наборов данных. Так как база ответов обучающихся, как правило, имеет меньший объём, чем, например, база отзывов или данных с веб-страниц, интерпретация результатов должна идти с учётом этого факта [13]. По этой причине исследователи уделяют большое внимание подготовке набора данных для изучения проблемы автоматизированного оценивания коротких ответов, полученных на разных языках [14]. В работе [15] авторы оценили при помощи предварительно обученных моделей 840 ответов на 21 вопрос. Улучшить точность оценки можно при расширении базы ответов. Также значение имеет сбалансированность набора, например, сопоставимое количество верных и неверных ответов. В этой работе получены результаты применения семи моделей, но отмечено, что они требуют перепроверки с другими наборами данных. После такой проверки планируется перейти к следующей цели — расширить проект, включив в него работу над ответами в виде эссе [16].

Нами были собраны 1319 коротких ответов студентов на 41 вопрос, полученных в ходе изучения дисциплины по выбору «Компьютерная графика и дизайн». Ответам была дана экспертная оценка. Большинство ответов было оценено экспертом как верные (1) или неверные (0). В некоторых случаях, например, если ответ предполагал перечисление, экспертная оценка могла выражаться дробным числом от 0 до 1. В настоящей работе представлены результаты оценивания ответов при помощи языковых моделей и уточнения полученных оценок с использованием проверки ключевых слов.

ОЦЕНИВАНИЕ ОТВЕТОВ С ИСПОЛЬЗОВАНИЕМ ЯЗЫКОВЫХ МОДЕЛЕЙ

Языковые модели word2vec и BERT применяются для решения различных задач, связанных с обработкой текста. Например, в [17] с использованием этих моделей проведен сентимент-анализ данных социальных сетей. В [18] определены наиболее эффективные модели семейства BERT для выявления деструктивного контента в социальных медиа. В настоящее время исследователи изучают не только процесс обучения и полученные результаты, но и объяснимость и прозрачность алгоритмов моделей, что в перспективе должно привести к их более оптимальному использованию (см., например, [19]).

Для оценивания ответов ранее мы применили модель word2vec, обученную на текстах по компьютерной графике. Результаты показали, что при помощи этой модели можно выявить ответы, явно неверные и близкие к верным по формулировке. В случаях, когда значение косинусной меры было близко к 0.5, предлагался дополнительный этап проверки [20].

Ответы студентов, полученные во время опросов, зачетов и экзаменов, дополнительно были проверены при помощи готовой языковой модели на основе BERT [21]. Если значение косинусной меры для векторов корректного ответа и ответа студента не превосходило 0.5, мы считали, что модель верно оценила ответ студента. Например, вариантом корректного (шаблонного) ответа на вопрос «Что называется воксельной моделью?» является ответ «Трехмерная модель, состоящая из вокселей (элементов объема, кубиков)». Значения косинусной меры между векторами некоторых ответов студентов и шаблонного ответа приведены в таблице 1. Ответы здесь даны в том виде, в котором получены от студентов, орфография и пунктуация сохранены.

Таблица 1. Результаты оценивания ответов

Ответы студентов	**Экспертная оценка**	**Косинусное сходство**
«Воксельная модель – состоящая из вокселей (кубов)»	1	0.716
«адаптация цветовой модели»	0	0.237
«Воксельная модель – трехмерный растр, состоит	1	0.452

из вокселей (как пиксели в 2D, только в 3D). Из них строится изображение»		

Как видно из таблицы 1, в первом и втором случаях модель оценила ответы верно, в третьем случае правильный ответ студента был оценен как неверный. Всего в 76.5 % случаев модель на основе трансформеров верно оценила ответы студентов. Но точность по отдельным вопросам варьируется от 46 до 97 процентов.

Была осуществлена проверка, как повторение теории влияет на результаты теста. Для этого студентам непосредственно перед тестированием была предоставлена возможность прочитать необходимый теоретический материал, содержащий ответы на предлагаемые вопросы. Количество правильных ответов ожидаемо увеличилось. Также точность результата проверки ответов, которые получены после прочтения, выше на 11.1%, что связано в первую очередь с тем, что, формулируя ответы, студенты использовали чаще те слова и предложения, которые только что повторили. Данные по 10 вопросам представлены на рис. 1 (точность оценки в %).

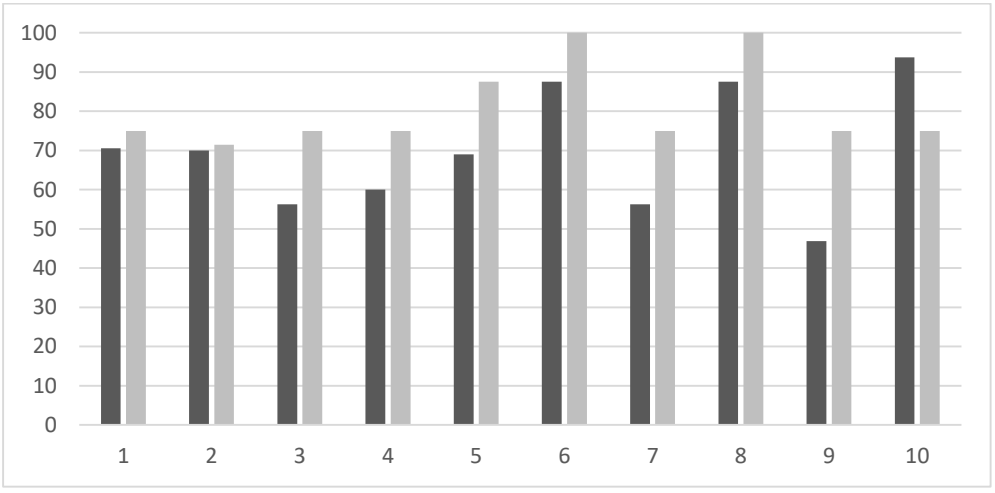


Рис. 1. Доля верно оцененных ответов (левые столбцы – ответы получены без повторения учебного материала, правые столбцы – ответы получены после повторения учебного материала)

В тех случаях, когда опрос проходит непосредственно после изучения темы, использование в формулировках корректного (шаблонного) ответа выражений из теоретического материала (лекций, рекомендованной литературы) позволит более точно оценить ответы. Если во время тестирования проверяются остаточные знания, при достаточно заполненной базе ответов возможно оценивание на основе сравнения верных ответов, полученных от студентов. Например, обучающиеся получили задание «Графический редактор – это ... (дописать определение, примеры приводить не нужно)». Как правило, это задание не вызывает затруднений, с графическим редактором обучающиеся знакомы со школы. Верные ответы студентов были сформулированы по-разному, например, «Приложение для редактирования изображений», «инструмент (приложение), которое позволяет редактировать и создавать изображения», «Приложение, с помощью которого можно создавать и изменять изображения» и т. д. Было найдено косинусное сходство между ответами студентов и корректным (шаблонным) ответом. После этого были выбраны 10 верных ответов, полученных от обучающихся. Затем сформирован вектор, являющийся средним для векторов верных ответов. Далее было вычислено косинусное сходство между средним вектором и векторами 10 случайных ответов студентов. В таблице 2 приведены оценки, выставленные экспертом, значения косинусной меры между векторами и абсолютные значения разности между оценками, выставленными экспертом и значениями косинусной меры.

Таблица 2. Результаты оценивания ответов различными способами

Номера ответов студентов	**Экспертная оценка**	**Значение косинусной меры**		**Абсолютное значение разности между экспертной оценкой и значением косинусной меры**	
		1-й способ	**2-й способ**	**1-й способ**	**2-й способ**
1	0.5	0.396	0.534	0.104	0.034
2	1	0.397	0.581	0.603	0.419
3	0	0.176	0.346	0.176	0.346
4	1	0.394	0.569	0.606	0.431
5	0.5	0.342	0.535	0.158	0.035

6	0	0.372	0.452	0.372	0.452
7	1	0.627	0.749	0.373	0.251
8	1	0.574	0.763	0.426	0.237
9	1	0.339	0.614	0.661	0.386
10	0	0.612	0.755	0.612	0.755

При 1-м способе рассматривались вектор ответа студента и вектор корректного (шаблонного) ответа. При 2-м способе – вектор ответа студента и вектор, являющийся средним для векторов верных ответов студентов. В большинстве случаев результаты оценивания стали лучше. Абсолютное значение разности между экспертной оценкой и значениями косинусной меры между векторами увеличилось для ответов, которые содержат слова верных ответов, но сформулированы ошибочно. Это ответ 3: «инструмент для работы с мультимедиями», ответ 6: «Приложение /облако, где можно редактировать или вносить какие-либо правки», ответ 10: «Графический редактор – приложение для редактирования» (орфография и пунктуация сохранены). Уточнить оценки ответов возможно проверкой ключевых слов.

ПРОВЕРКА КЛЮЧЕВЫХ СЛОВ

Под ключевыми словами корректного (шаблонного) ответа на вопрос будем понимать значимые, важные слова ответа. Специалистами проверка ключевых слов предлагается как одна из возможностей совершенствования процесса оценивания коротких ответов [22].

Для проверки ключевых слов сначала была проведена предобработка ответов, включая токенизацию, удаление стоп-слов и знаков препинания, а также лемматизацию. Чтобы избежать дублирования исходной информации в ответах студентов, были удалены слова, которые содержались в формулировке самого вопроса. Такие слова не несут дополнительного смысла и могут исказить результаты анализа. Например, если ответ на вопрос «Как получается цвет в модели RGB?» начинается с повторения слов вопроса: «Цвет в модели RGB получается …», данные слова были удалены. Этап предобработки помог стандартизировать и очистить текстовые данные, делая их готовыми для дальнейшего анализа и обеспечивая более точные и непротиворечивые результаты.

На следующем шаге были определены ключевые слова, для каждого из них назначен вес, который отражает значимость ключевого слова в ответе. Совпадения ключевых слов между корректным ответом и ответом студента оцениваются с учетом их весов. На приведенный выше вопрос про модель RGB было получено 64 ответа. Вариант корректного ответа: «смешиванием основных компонент цветовой модели – Red, Green, Blue (красный, зеленый, синий)». В 51 случае (79.7 %) оценки, выставленные экспертом и определенные через косинусное сходство с использованием языковой модели, совпали. В 7 случаях из оставшихся неверно оцененных 13 ответов проверка ключевых слов помогла уточнить оценки. Ключевыми были определены слова: «красный» (или «Red»), «зеленый» («Green»), «синий» («Blue»). В итоге в 6 случаях из 64 оценка, определенная через косинусное сходство, не совпала с экспертной. Некоторые из неверно оцененных ответов приведены в таблице 3 (орфография и пунктуация сохранены).

Таблица 3. Результаты оценивания ответов

Ответы студентов	**Экспертная оценка**	**Косинусное сходство**	**Ключевые слова в ответе**
«Цвет получается путем смешивания 3 основных цветов в одном пикселе»	1	0.267	отсутствуют
«Путем смешивания 3 цветов RGB в разных пропорциях»	1	0.241	приведены сокращенно
«Путем слияния в разных количествах параметров R G и B, варьирующих в значении от 0 до 255, где все по 0 – черный цвет, все по 255 – белый»	1	0.123	приведены сокращенно
«RGB – Red Green Blue?»	0	0.535	присутствуют

Вопрос про модель RGB не вызывал затруднений у студентов. В первых трех случаях из приведенных в таблице 3 эксперт не снизил оценку за то, что цвета не

были названы полностью. Отметим, что экспертные оценки не снижались за грамматические, орфографические, речевые, стилистические и другие ошибки, если они не искажали смысл предложения, так как проверялись знания по компьютерной графике. В четвертом ответе обучающийся не написал про смешение (сложение) компонентов, поэтому эксперт оценил ответ как неверный. Косинусное сходство близко к 0.5, ключевые слова присутствуют. После вычисления косинусного сходства и проверки ключевых слов в 90.7 % случаев ответы студентов на данный вопрос были оценены верно.

Таким образом, векторизация ответов при помощи языковых моделей предоставляет возможность предварительного оценивания ответов обучающихся. В работе были проанализированы ответы, которые определяются, в основном, как верные или неверные. Разработка критериев позволит получить более объективные оценки эксперта и повысит точность результатов применения моделей, но увеличит время работы преподавателей. Полученные результаты могут помочь при анализе развернутых ответов на вопросы, требующих пояснения нескольких понятий.

ЗАКЛЮЧЕНИЕ

Представлены результаты анализа ответов студентов, сформулированных в свободной форме. Векторы предложений получены при помощи языковой модели на основе BERT, далее определялось косинусное сходство ответа обучаемого и корректного (шаблонного) ответа. В 76.5 % всех случаев языковая модель позволила оценить верность ответа обучающегося. Точность полученной оценки зависит от формулировки корректного ответа, времени и места тестирования в графике учебного процесса. При достаточном заполнении базы ответов возможно вместо шаблонного ответа взять вектор, являющийся средним вектором для векторов верных ответов обучающихся. Описаны подходы к уточнению оценки при помощи проверки ключевых слов. Результаты могут использоваться при предварительной проверке ответов студентов, в составе цифровых образовательных ресурсов при выполнении тестов на самопроверку с последующим обсуждением оценок.

Благодарности

Работа выполнена за счет средств Программы стратегического академического лидерства Казанского (Приволжского) федерального университета («ПРИОРИТЕТ-2030»)

СПИСОК ЛИТЕРАТУРЫ

1. *Шепелюк О.Л.* Методы контроля знаний студентов в условиях реализации образовательных стандартов // Глобальный научный потенциал. 2021. № 8 (125). С. 112–114.

2. *Бухман Л.М.* Проблемы тестового контроля знаний и их решение // Известия Самарского научного центра Российской академии наук. 2010. Т. 12, № 5. С. 21–24.

3. *Тихонова Н.В.* Организация контроля знаний студентов в условиях удаленного обучения // Казанский лингвистический журнал. 2021. Т. 4, № 1. С. 111–125.

4. *Атнагулов А.А., Абрамский М.М.* О подходе к автоматизации оценки знаний в области разработки программного обеспечения на основе анализа данных проектной работы // Электронные библиотеки. 2023. Т. 26, № 5. С. 589–599.

5. *Жуков И.А.* Система контроля знаний и практических навыков по программированию // Информатика и образование. 2023. Т. 38, № 2. С. 66–74.

6. *Nevzorova O.A., Falileeva M.V., Kirillovich A.V., Lipachev E.K., Shakirova L.R., Dyupina A.E.* OntoMathEdu Educational Ontology: Problems of Ontological Engineering // Pattern Recognition and Image Analysis. 2023. V. 33 (3). P. 460–466.

7. *Burrows S., Gurevych I. and Stein B.* The Eras and Trends of Automatic Short Answer Grading // International Journal of Artificial Intelligence in Education. 2015. V. 25. P. 60–117.

8. *Yan L., Sha L., Zhao L., Li Y., Martinez-Maldonado R., Chen G., Li X., Jin Y., Gašević D.* Practical and ethical challenges of large language models in education: A systematic scoping review // British Journal of Educational Technology. 2023. V. 55, Is. 1. P. 90–112.

9. *Zirar A.* Exploring the impact of language models, such as ChatGPT, on student learning and assessment // Review of Education.2023. V. 11, e3433.

10. *Sundaram S.S., Gurajada S., Padmanabhan D., Sam Abraham S., Fisichella M.*

Does a language model “understand” high school math? A survey of deep learning based word problem solvers // WIREs Data Mining and Knowledge Discovery. 2024. 1534.

11. *Гиниятуллин В.М., Ермолаев Е.В., Салихова М.А., Хлыбов А.В., Чурилов Д.А., Чурилова Е.А.* Исследование структуры и содержания компетенций с помощью языковой модели ELMO // Современные наукоемкие технологии. 2021. № 8. С. 58–65.

12. *Лагутина Н.С., Лагутина К.В., Бредерман А.М., Касаткина Н.Н.* Классификация текстов по уровням CEFR с использованием методов машинного обучения и языковой модели BERT // Моделирование и анализ информационных систем. 2023. Т. 30, № 3. С. 202–213.

13. *Ahmed A., Joorabchi A., Hayes M.* On Deep Learning Approaches to Automated Assessment: Strategies for Short Answer Grading //In Proceedings of the 14th International Conference on Computer Supported Education (CSEDU 2022). 2022. V. 2. P. 85–94.

14. *Agarwal D., Gupta S., Baghel N.* ScAA: A Dataset for Automated Short Answer Grading of Children’s Free-text Answers in Hindi and Marathi // Proceedings of the 17th International Conference on Natural Language Processing, Patna, India, December 18 – 21, 2020. P. 430–436.

15. *Wilianto D., Girsang A.S.* Automatic Short Answer Grading on High School’s E-Learning Using Semantic Similarity Methods // TEM Journal. 2023. V. 12, Is. 1. P. 297–302.

16. *Divya A., Haridas V. and Narayanan J.* Automation of Short Answer Grading Techniques: Comparative Study using Deep Learning Techniques // 2023 Fifth International Conference on Electrical, Computer and Communication Technologies (ICECCT), Erode, India. 2023. P. 1–7.

17. *Ярушкина Н.Г., Мошкин В.С., Константинов А.А.* Применение языковых моделей word2vec и bert в задаче сентимент-анализа текстовых сообщений социальных сетей // Автоматизация процессов управления. 2020. № 3 (61). С. 60–69.

18. *Минаев В.А., Симонов А.В.* Сравнение моделей-трансформеров BERT при выявлении деструктивного контента в социальных медиа // Информация и безопасность. 2022. Т. 25, № 3. С. 341–348.

19. *Surov I.* Opening the Black Box: Finding Osgood's Semantic Factors in Word2vec Space // Informatics and Automation. 2022. V. 21, No. 5. P. 916–936.

20. *Миннегалиева Ч.Б., Сабитова Г.А., Гаялиев А.М.* Метод предварительной оценки ответов обучающихся на основе векторной модели документов // Электронные библиотеки. 2023. Т. 26, № 3. С. 324–339.

21. *Reimers N., Gurevych I.* Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks // In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) Hong Kong, China Association for Computational Linguistics. 2019. P. 3982–3992.

22. *Кожевников В.А., Сабинин О.Ю.* Система автоматической проверки ответов на открытые вопросы на русском языке // Научно-технические возможности СПбГПУ. 2018. Т. 11, № 3. С. 57–72.

AUTOMATED STUDENTS' SHORT ANSWERS GRADING USING LANGUAGE MODELS

Chulpan Minnegalieva[1] [0000-0003-4648-1623], Ilnur Kashapov[2] [0009-0004-6341-8380], Olga Morozova[3] [0009-0009-5670-561X]

[1–3]*Institute of Computer Mathematics and Information Technologies, Kazan Federal University*

[1]mchulpan@gmail.com, [2]k.i.i.08@yandex.ru, [3]olka30.olka30@mail.ru

Abstract

Methods for assessing student answers using language models are currently being studied by various specialists. The results of automated assessment depend on the subject area and characteristics of the academic discipline. This paper analyzes the students' answers received during the course «Computer Graphics and Design». It is proposed to determine the cosine similarity of document vectors obtained using language models and refine the estimates by checking keywords. The results obtained can be used for preliminary assessment of students' answers and are the basis for further research.

***Keywords**: language model, knowledge control, natural text processing, student answer keyword, automated short answer grading, cosine similarity, document vector representation, BERT, word2vec, open-ended question*

REFERENCES

1. *Shepelyuk O.L.* Metody kontrolya znanij studentov v usloviyah realizacii obrazovatel'nyh standartov // Global'nyj nauchnyj potencial. 2021. № 8 (125). S. 112–114.

2. *Buhman L.M.* Problemy testovogo kontrolya znanij i ih reshenie // Izvestiya Samarskogo nauchnogo centra Rossijskoj akademii nauk. 2010. T. 12, № 5. S. 21–24.

3. *Tihonova N.V.* Organizaciya kontrolya znanij studentov v usloviyah udalennogo obucheniya // Kazanskij lingvisticheskij zhurnal. 2021. T. 4, № 1. S. 111–125.

4. *Atnagulov A.A., Abramskij M.M.* Student teams project work analysis tool development // Russian Digital Libraries Journal. 2023. T. 26, No 5. P. 589–599.

5. *Zhukov I.A.* Sistema kontrolya znanij i prakticheskih navykov po programmirovaniyu // Informatika i obrazovanie. 2023. T. 38, № 2. S. 66–74.

6. *Nevzorova O.A., Falileeva M.V., Kirillovich A.V., Lipachev E.K., Shakirova L.R., Dyupina A.E.* OntoMathEdu Educational Ontology: Problems of Ontological Engineering // Pattern Recognition and Image Analysis. 2023. V. 33 (3). P. 460–466.

7. *Burrows S., Gurevych I. and Stein B.* The Eras and Trends of Automatic Short Answer Grading // International Journal of Artificial Intelligence in Education. 2015. V. 25. P. 60–117.

8. *Yan L., Sha L., Zhao L., Li Y., Martinez-Maldonado R., Chen G., Li X., Jin Y., Gašević D.* Practical and ethical challenges of large language models in education: A systematic scoping review // British Journal of Educational Technology. 2023. V. 55, Is. 1. P. 90–112.

9. *Zirar A.* Exploring the impact of language models, such as ChatGPT, on student learning and assessment // Review of Education.2023. V. 11, e3433.

10. *Sundaram S.S., Gurajada S., Padmanabhan D., Sam Abraham S., Fisichella M.* Does a language model "understand" high school math? A survey of deep learning based word problem solvers // WIREs Data Mining and Knowledge Discovery. 2024. 1534.

11. *Giniyatullin V.M., Ermolaev E.V., Salihova M.A. Hlybov A.V., Churilov D.A., Churilova E.A*. Issledovanie struktury i soderzhaniya kompetencij s pomoshch'yu yazykovoj modeli ELMO // Sovremennye naukoemkie tekhnologii. 2021. № 8. S. 58–65.

12. *Lagutina N.S., Lagutina K.V., Brederman A.M., Kasatkina N.N.* Klassifikaciya tekstov po urovnyam CEFR s ispol'zovaniem metodov mashinnogo obucheniya i yazykovoj modeli BERT // Modelirovanie i analiz informacionnyh sistem. 2023. T. 30, № 3. S. 202–213.

13. *Ahmed A., Joorabchi A., Hayes M.* On Deep Learning Approaches to Automated Assessment: Strategies for Short Answer Grading //In Proceedings of the 14th International Conference on Computer Supported Education (CSEDU 2022). 2022. V. 2. P. 85–94.

14. *Agarwal D., Gupta S., Baghel N.* ScAA: A Dataset for Automated Short Answer Grading of Children's Free-text Answers in Hindi and Marathi // Proceedings of the 17th International Conference on Natural Language Processing, Patna, India, December 18 – 21, 2020. P.430–436.

15. *Wilianto D., Girsang A.S.* Automatic Short Answer Grading on High School's E-Learning Using Semantic Similarity Methods // TEM Journal. 2023. V. 12, Is. 1. P. 297–302.

16. *Divya A., Haridas V. and Narayanan J.* Automation of Short Answer Grading Techniques: Comparative Study using Deep Learning Techniques // 2023 Fifth International Conference on Electrical, Computer and Communication Technologies (ICECCT), Erode, India. 2023. P. 1–7.

17. *Yarushkina N.G., Moshkin V.S., Konstantinov A.A.* Primenenie yazykovyh modelej word2vec i bert v zadache sentiment-analiza tekstovyh soobshchenij social'nyh setej // Avtomatizaciya processov upravleniya. 2020. № 3 (61). S. 60–69.

18. *Minaev V.A., Simonov A.V.* Sravnenie modelej-transformerov BERT pri vyyavlenii destruktivnogo kontenta v social'nyh media // Informaciya i bezopasnost'. 2022. T. 25, № 3. S. 341–348.

19. *Surov I.* Opening the Black Box: Finding Osgood's Semantic Factors in Word2vec Space // Informatics and Automation. 2022. V. 21, No. 5. P. 916–936.

20. *Minnegalieva Ch.B., Sabitova G.A., Gayaliev A.M.* Method of pre-assessment

of students' answers based on the vector model of documents// Russian Digital Libraries Journal. 2023. T. 26, No 3. P. 324–339.

21. *Reimers N., Gurevych I.* Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks // In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) Hong Kong, China Association for Computational Linguistics. 2019. P. 3982–3992.

22. *Kozhevnikov V.A., Sabinin O.Yu.* Sistema avtomaticheskoj proverki otvetov na otkrytye voprosy na russkom yazyke // Nauchno-tekhnicheskie vozmozhnosti SPbGPU. 2018. T. 11. № 3. S. 57–72.

СВЕДЕНИЯ ОБ АВТОРАХ

МИННЕГАЛИЕВА Чулпан Бакиевна – к. п. н., доцент кафедры информационных систем Института вычислительной математики и информационных технологий, Казанский федеральный университет, г. Казань.

Chulpan MINNEGALIEVA – Associate Professor, Institute of Computer Mathematics and Information Technologies, Kazan Federal University.

email: mchulpan@gmail.com

ORCID: 0000-0003-4648-1623.

КАШАПОВ Ильнур Илхамович – магистр 1 г.о., Институт вычислительной математики и информационных технологий, Казанский федеральный университет, г. Казань.

Ilnur KASHAPOV – master's student, Institute of Computer Mathematics and Information Technologies, Kazan Federal University.

email: k.i.i.08@yandex.ru

ORCID: 0009-0004-6341-8380

МОРОЗОВА Ольга Дмитриевна – магистр 1 г.о., Институт вычислительной математики и информационных технологий, Казанский федеральный университет, г. Казань.

Olga MOROZOVA – master's student, Institute of Computer Mathematics and Information Technologies, Kazan Federal University.

email: olka30.olka30@mail.ru

ORCID: 0009-0009-5670-561X

Материал поступил в редакцию 2 мая 2024 года

УДК 004.89

АНАЛИЗ МОДЕЛЕЙ МАШИННОГО ОБУЧЕНИЯ НА ОСНОВЕ МЕТОДОВ ОБЪЯСНИМОГО ИСКУССТВЕННОГО ИНТЕЛЛЕКТА В ОБРАЗОВАТЕЛЬНОЙ АНАЛИТИКЕ

Д. А. Минуллин[1] [0000-0001-7713-5251], Ф. М. Гафаров[2] [0000-0003-4704-154X]

[1, 2]*Институт вычислительной математики и информационных технологий, Казанский (Приволжский) федеральный университет*

[1]minullin.dima@mail.ru, [2]fgafarov@yandex.ru

Аннотация

Проблема прогнозирования досрочного отчисления студентов российских вузов является актуальной, поэтому требуется разработка новых инновационных подходов для её решения. Для решения данной проблемы возможна разработка предиктивных систем на основе использования данных о студентах, имеющихся в информационных системах вузов. В настоящей работе исследованы модели машинного обучения для прогнозирования досрочного отчисления студентов, обученные на основе данных о характеристиках и успеваемости студентов. Основная научная новизна работы заключается в использовании методов объяснимого ИИ для интерпретации и объяснения функционирования обученных моделей машинного обучения. Методы объяснимого искусственного интеллекта позволяют понять, какие из входных признаков (характеристик студента) оказывают наибольшее влияние на результаты прогнозов обученных моделей, а также могут помочь понять, почему модели принимают те или иные решения. Полученные результаты расширяют понимание влияния различных факторов на досрочное отчисление студентов.

Ключевые слова*: образовательная аналитика, интеллектуальный анализ данных, машинное обучение, объяснимый искусственный интеллект*

ВВЕДЕНИЕ

Одной из ключевых проблем современного образования является досрочное отчисление из высших учебных заведений студентов по неуспеваемости. Дан-

ная проблема может привести не только к личностным трудностям самих студентов, но и повлечь за собой социальные проблемы, связанные с утратой образовательных ресурсов общества [1]. Досрочное отчисление студентов из-за проблем с успеваемостью сильно влияет на всю систему образования, т. к. приводит не только к уменьшению количества студентов в учебных группах, но и может повлечь за собой личные образовательные последствия, а также проблемы для самого университета, такие, например, как снижение рейтинга и увеличение затрат [2, 3]. Таким образом, необходимы стратегии и меры, направленные на решение данной проблемы и улучшение качества образования, с целью снижения количества отчислений студентов. Исследования названной проблемы основаны не только на данных об успеваемости, но и на широком спектре факторов, которые могут отказать влияние на решение студентов о продолжении обучения. В этих исследованиях анализируются такие аспекты, как мотивация, уровень интереса к обучению, успешная интеграция в университетскую среду, адаптация к новым условиям, удовлетворенность образовательным процессом. Цель исследователей – определить влияние этих факторов на решение студентов продолжить обучение. На основе полученных данных разрабатываются различные рекомендации для университетов по снижению количества отчисленных студентов и предлагаются меры по повышению качества образования [4–7].

В связи со сказанным разработка инновационных подходов к прогнозированию академической успешности студентов и выявлению студентов, которые могут быть отчислены, становится особенно актуальной. Одним из перспективных подходов здесь является принятие решений на основе данных. Использование методов машинного обучения для предсказания вероятности преждевременного отчисления студентов представляет собой современную стратегию [8]. Эти методы могут дать возможность оперативно выявлять студентов, находящихся в группе риска, и предотвращать их отчисление. Точные прогнозы, основанные на данных, позволят учебным заведениям предпринять шаги по оказанию помощи и адаптации, ориентированные на улучшение успеваемости студентов и повышение их удовлетворенности образовательным процессом.

Модели машинного обучения способны эффективно распознавать сложные закономерности и делать прогнозы на основе данных. В последнее время растет

интерес к интерпретации таких моделей как со стороны научного сообщества [9–11], так и со стороны пользователей различных предиктивных систем. Сфера интерпретируемого машинного обучения, также известная как «объяснимый ИИ» (Explainable AI), активно развивается, и в этой области появляется все больше исследований [12–17]. Такие подходы уже успешно применяется в различных сферах, включая медицину, энергетику и науку [18–21].

Для анализа моделей машинного обучения имеются различные инструменты и фреймворки, помогающие раскрыть внутреннюю работу моделей, делая их результаты более понятными (например, Captum [22, 23] и SHAPexplainer [24]). Использование этих инструментов позволяет визуализировать результаты работы алгоритмов интерпретации моделей, чтобы выяснить, как различные значения входных данных влияют на прогнозы моделей [25–28].

В настоящей работе показано использование методов Integrated Gradients, DeepLIFT, GradientSHAP и SHAP для анализа результатов применения моделей, обученных на решение задачи прогнозирования досрочного отчисления студентов. Для обучения моделей машинного обучения были использованы данные по выпускникам и досрочно выбывшим студентам Казанского (Приволжского) федерального университета в период с 2012 по 2019 годы. Для определения потенциальной группы риска среди обучающихся нами была использована модель прогнозирования бинарной классификации; выявлены факторы, которые больше всего способствуют досрочному отчислению студентов.

ДАННЫЕ И МЕТОДЫ

Был использован набор данных, представленный в статье [29]. Этот набор состоит из записей о выпускниках с полными данными за весь четырёхлетний цикл обучения, и досрочно выбывших студентах Казанского (Приволжского) федерального университета. Для обучения моделей был собран максимально полный и подробный набор данных, который отражает процесс обучения студентов в вузе, чтобы определить факторы, способствовавшие досрочному выбытию студента. Данные студента характеризуются набором параметров двух типов: числовые и категориальные. К числовым параметрам относятся баллы ЕГЭ, средний балл ЕГЭ, общий средний балл успеваемости за все полные семестры, средний балл за первый и второй семестры. К категориальным параметрам относятся пол

студента, форма обучения (очная\заочная), оплата обучения (бюджет\контракт), тип предыдущего образования, год поступления и специализация студента. Значения всех категориальных параметров были закодированы методом one-hot encoding в соответствии с [29].

Для построения прогностической системы прогнозирования досрочного выбытия студентов были обучены и проанализированы модели двух типов (таблица 1). Первый тип моделей основан на искусственной нейронной сети (ИНС), состоящей из трёх полносвязных слоёв, второй тип – на алгоритме градиентного бустинга (XGBoost). Модели обучались на решениях задачи бинарной классификации, на выходе они давали прогноз того, что студент успешно завершит обучение (выход – 0) или же он будет досрочно отчислен (выход – 1).

Таблица 1. Описание используемых подходов и методов.

Подход\ Метод	**Описание**	**Преимущества**	**Недостатки**
Искусственные нейронные сети	Модели, имитирующие работу человеческого мозга, обучаются на основе выборки данных для предсказания результатов	Могут обрабатывать очень большое количество данных, выявлять сложные нелинейные зависимости, адаптивны и способны к самообучению.	Сложны в интерпретации, требуют значительных вычислительных мощностей и большого количества обучающих данных, подвержены переобучению.
XGBoost	Алгоритм градиентного бустинга, использующий в качестве базовых моделей деревья решений.	Относительно быстрый, эффективный, расширяемый, поддерживает разные типы данных, автоматиче-	Может переобучиться при некорректно подобранных параметрах, требует тщательной

		ски обрабатывает пропущенные значения, регулирует сложность модели.	настройки гиперпараметров.
Integrated Gradients	Метод интерпретации моделей машинного обучения, основанный на аппроксимации градиентов входных данных вдоль базовой линии.	Позволяет увидеть, какие признаки влияют на прогноз модели, позволяет интерпретировать сложные модели, может быть применён к моделям любого размера.	Подвержен проблеме «sparse gradients», вычислительно затратен для больших моделей
SHAP	Метод интерпретации предсказаний моделей машинного обучения, основанный на теории игр, позволяет объяснить влияние каждого признака на прогноз модели на основе их индивидуального вклада.	Объективен, может оценить важность признаков как для всей модели, так и для отдельных предсказаний, позволяет интерпретировать сложные модели.	Требует значительных вычислительных ресурсов для сложных моделей и требовать больше времени для вычислений, может быть сложен в понимании.
DeepLIFT	Метод интерпретации нейронных сетей, сравнивает активацию нейронов и присваивает им оценки.	Позволяет интерпретировать сложные нейронные сети, в частности глубокие сети, пытается решить проблему «sparse gradients», может быть применен к любым архитектурам нейронных сетей.	В некоторых случаях может быть неэффективным для моделей с большим количеством параметров

Gradient-SHAP	Комбинирует методы SHAP и Integrated Gradients для интерпретации моделей, предлагает объяснения на основе важности признаков.	Объединяет преимущества обоих методов, использует градиенты для ускорения SHAP, более точен, позволяет интерпретировать сложные модели.	Требует больше вычислительных ресурсов, чем SHAP или Integrated Gradients, может быть сложным в реализации.

Основным направлением настоящего исследования является применение методов интерпретации моделей машинного обучения, таких как Integrated Gradients, DeepLIFT, GradientSHAP и SHAP. Эти методы позволяют заглянуть внутрь «черного ящика» на основе графической визуализации влияния значений входных параметров на результаты прогноза обученных моделей. Методы Integrated Gradients, DeepLIFT и GradientSHAP относятся к типу градиентных. Эти методы присваивают важность каждому входному признаку, анализируя, как его изменения влияют на выход модели, используя градиентное разложение для количественной оценки этих эффектов. Для моделей XGBoost был применён метод атрибуции SHAP. Различные методы могут по-разному оценивать значимость признаков, и поэтому для получения максимально достоверного результата необходимо одновременно комбинировать разные методы. Эти методы позволяют выявить признаки, оказывающие наиболее существенное влияние на прогнозы модели, а также вводить числовые характеристики для количественной оценки их важности, что позволяет провести сравнение важности признаков.

Программный модуль анализа и интерпретация данных на основе интегрированных градиентов, DeepLIFT и GradientSHAP был разработан с использованием библиотеки Captum, а метод SHAP был реализован на основе одноимённой библиотеки (SHAP) для языка программирования Python. В библиотеку Captum встроены различные методы интерпретации, которые помогают объяснить модели глубокого обучения, разработанные с помощью фреймворка PyTorch. Для реализации модуля на основе SHAP был использован модуль KernelExplainer библиотеки Python SHAP, который работает универсально для любой модели прогнозирова-

ния. Этот модуль вычисляет прогностическую ценность признаков на основе значений SHAP. На вход модуля интерпретации подаются обученные модели, а также данные из тестового набора данных. На выходе модуля получаются атрибуции для каждого входного фактора, показывающие, насколько высока прогностическая сила каждого фактора.

РЕЗУЛЬТАТЫ

Был проведен сравнительный анализ интерпретации моделей нейронных сетей, обученных на полном наборе данных (40883 записей) и на данных в различных разрезах: по полу, специализации, предыдущему образованию, типу обучения. Средняя точность нейросетевой модели, обученной на полном наборе данных, составила 87%, средняя точность моделей, обученных на различных разрезах данных, – 84%. Для проверки точности моделей использовались данные из тестового набора данных, который составлял 30% от исходного.

Методы объяснимого ИИ позволяют получить числовые значения, показывающие степень влияния каждого входного признака на прогноз модели. Такие оценки способствуют пониманию, насколько похожи или отличаются значения атрибуций входных факторов, полученных на основе различных методов интерпретации (Integrated Gradients, DeepLIFT и GradientSHAP, SHAP). Величины весов, полученные в ходе применения методов интерпретации, могут иметь два направления – положительное и отрицательное. Эти направления несут информацию о влиянии конкретных входных параметров модели на выходное значение модели. Важность входных признаков, имеющих отрицательные и положительные числа, заключается в следующем: отрицательное значение показывает негативное влияние на вероятность досрочного отчисления студента, а положительное – увеличивает вероятность досрочного отчисления студента. Важность признака показывает, насколько он важен для классификации (рис. 1).

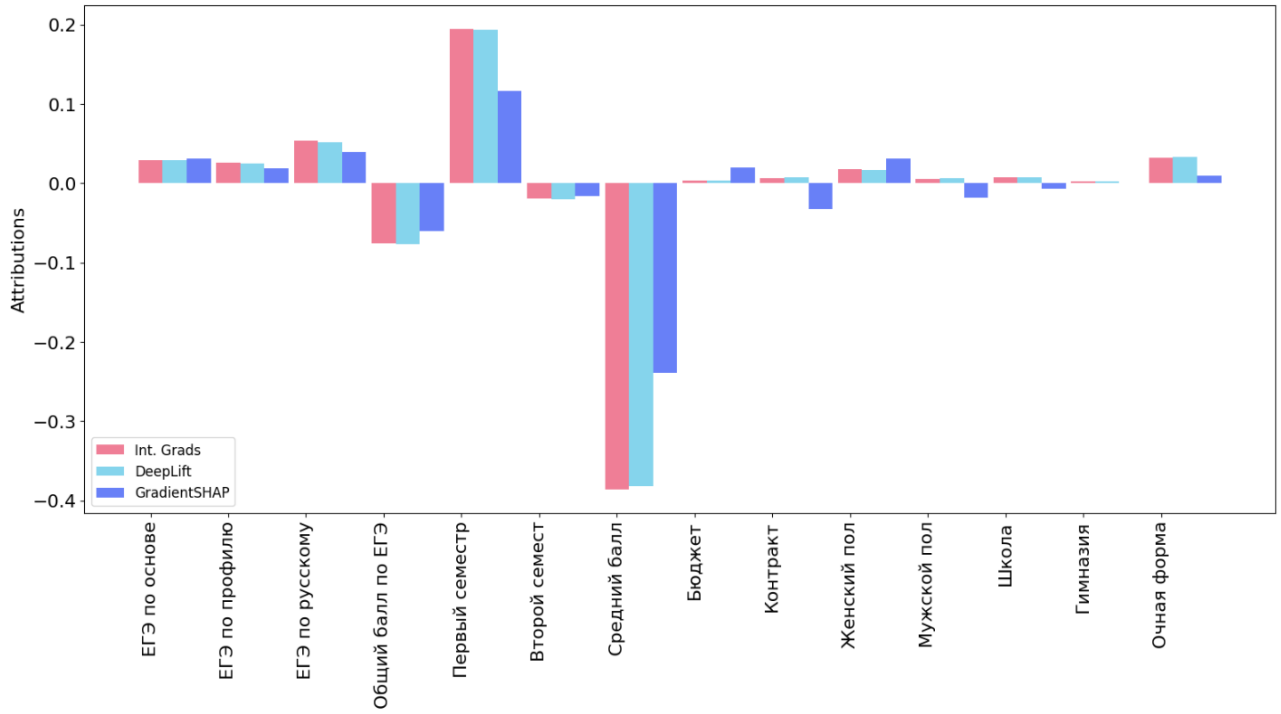


Рис. 1. Параметры атрибуции факторов для на основе методов Integrated Gradients, DeepLIFT и GradientSHAP

Сравнивая значения, полученные в ходе интерпретации нейронных сетей, обученных отдельно для студентов мужского и женского пола (Таблица 2), можно заметить, что в случае женского пола большее влияние на прогноз модели оказывают баллы ЕГЭ и оценки за второй семестр, в то время как у студентов мужского пола влияние баллов ЕГЭ ниже, а оценки за второй семестр имеют даже негативное влияние. Также, в отличие от случая женского пола, у студентов мужского пола начинает приобретать значимость признак – очная форма обучения. Остальные параметров имеют приблизительно одинаковы значения для обоих моделей.

Таблица 2. Значения интегрированных градиентов для нейронных сетей, обученных отдельно для студентов мужского и женского пола.

	Мужской	Женский
ЕГЭ по основе	0,02420	0,06589
ЕГЭ по профилю	0,03204	0,05651
ЕГЭ по русскому	0,03634	0,08342

Первый семестр	0,18182	0,13563
Второй семестр	–0,0099	0,00785
Очная форма	0,03290	0,00346

Также имеются явные различия в значениях интегрированных градиентов для нейронных сетей, обученных независимо для каждого из исследуемых типов обучения (таблица 3). Эти отличия наблюдались между всеми видами обучения, исключения составили такие параметры, как оценки за первый и второй семестры, которые всегда оказывали положительное влияние на прогнозы моделей, и факторы, значения интегрированных градиентов которых близки к нулю. Анализ моделей, обученных на данных студентов очной формы обучения, демонстрирует негативное влияние баллов ЕГЭ, положительное влияние признака пола студента, причём значение «женский пол» оказывает большее влияние на прогнозы модели, чем значение «мужской пол». Также значительное положительное влияние на результаты прогноза модели оказывает принадлежность к бюджетной форме обучения. Для студентов, обучающихся на очно-заочной форме, характерно отрицательное влияние баллов ЕГЭ по профильной дисциплине и русскому языку, и положительное – по основной учебной дисциплине. В отличие от очной формы обучения, для заочной формы обучения присуще отрицательное влияние фактора «женский пол» на прогнозы модели. Для студентов заочной формы обучения характерно положительное влияние баллов ЕГЭ и пола студента, в то время как принадлежность к бюджетной форме обучения имеет отрицательное влияние на прогнозы модели.

Таблица 3. Значения интегрированных градиентов для нейронных сетей, обученных отдельно для разных типов обучения.

	Очная	**Очно-заочная**	**Заочная**
ЕГЭ по основе	–0,05182	0,15287	0,02765
ЕГЭ по профилю	–0,06643	–0,1274	0,01641
ЕГЭ по русскому	–0,04512	–0,0948	0,00523

Женский пол	0,02742	–0,00125	0,02123
Мужской пол	0,00562	0,00762	0,00673
Бюджет	0,02456	0,002542	–0,00512
Контракт	0,00812	0,00856	0.03144

Основное различие между значениями интерпретаций форм обучения (бюджет/контракт) (таблица 4) проявляется в отношении влияния оценок за второй семестр: у бюджетной формы обучения наблюдается отрицательное влияние на прогноз модели, при контрактной – положительное. Важно отметить, что для студентов контрактной формы обучения более значимое влияние на прогнозы модели оказывают баллы ЕГЭ по профильной дисциплине, чем для студентов бюджетной формой обучения. Также можно выделить такие факторы, как «пол студента», «обучение на очной основе» и «женский пол», которые оказывает существенное влияние на прогнозы модели. Влияние же остальных признаков несущественно.

Таблица 4. Значения интегрированных градиентов для нейронных сетей, обученных отдельно для бюджетных и контрактных студентов.

	Бюджет	**Контракт**
ЕГЭ по профилю	0,017248	0,049231
Второй семестр	–0,00909	0,010561
Женский пол	0,012281	0,018576
Мужской пол	0,002932	0,001745
Очное обучение	0,01106	0,024567

Интерпретация моделей, обученных на данных, принадлежащих студентам, разделенным на основании вида их предыдущего образования (таблица 5), показала, что основными факторами, положительно влияющими на прогнозы моделей, являются следующие параметры: баллы по ЕГЭ, оценки за первый и второй семестры, пол студента, очное обучение, бюджетная и контрактная формы обуче-

ния. Во всех случаях, за исключением принадлежности предыдущего образования к «Лицею», «Вузу» и «Колледжу», можно наблюдать положительное влияние фактора «оценки за второй семестр» на прогнозы модели. Также для характеристики «Вуза» характерно негативное влияние на прогноз модели показателей ЕГЭ по основной и по профильным дисциплинам.

Таблица 5. Значения интегрированных градиентов для нейронных сетей, обученных отдельно по типу предыдущего образования.

	Школа	**Гимназия**	**Лицей**	**Вуз**	**Колледж**	**Техникум**
ЕГЭ по основе	0,07532	0,06231	0,05121	–0,01246	0,02541	0,02963
ЕГЭ по профилю	0,08031	0,06912	0,04592	–0,07125	0,00341	0,09863
Второй семестр	0,01834	0,02657	–0,06122	–1,14231	–0,0185	0,06187

Далее было проведено исследование возможностей метода SHAP для интерпретации моделей, обученных на основе градиентного бустнга (XGBoost). На рис. 2 показана диаграмма “waterfall”, которая позволяет визуализировать индивидуальные SHAP-значения, вычисленные для отдельного объекта из набора данных. Все параметры, включённые в набор данных, расположены на оси «*y*» и ранжированы сверху вниз на основе их значений-SHAP для данного конкретного прогноза. Фактическое значение объекта также отображается на оси «*y*». Влияние каждого параметра на прогноз представлено синей или красной стрелками и соответствующим значением-SHAP, которое отражает влияние этого признака на прогноз модели. Каждый прогноз начинает свою точку отсчёта с некоторого «базового значения» и влияние каждого признака смещает прогноз.

Рис. 2. Диаграмма значений SHAP для студента, успешно окончившего обучение (а) и для досрочно отчисленного студента (б).

В библиотеке SHAP имеется также другой тип представления локальных объяснений в виде графика силы – «force plot». Графики силы для двух студентов представлены на рис. 3.

Рис. 3. График силы (force plot) значений SHAP для студента, успешно окончившего обучение (а) и для досрочно отчисленного студента (б).

Для сравнения на рисунках 2 и 3 представлены интерпретации моделей (значения SHAP) для двух возможных классов данных: для студента, окончившего обучение, и досрочно отчисленного студента. Для студента, успешно окончившего обучение, основными показателями, влияющими на продолжение обучения, являются: общая средняя успеваемость, оценки за первый семестр и баллы за ЕГЭ по русскому языку. Оценки за второй семестр и то, что его предыдущим образованием являлся колледж, в свою очередь, оказали наибольшее влияние в сторону досрочного выбытия студента и «двигали» прогноз модели в сторону возможного отчисления (см. рис. 2а и 3а). Для студента (см. рис. 2б и 3б), который

был досрочно отчислен основными признаками, повлиявшим на выбытие, оказались его средний балл (несмотря на его достаточно высокое среднее значение), а также год поступления (2018). Различия SHAP-значений указывают на то, что каждый из признаков по-особенному влияет на результат прогноза модели, что укрепляет доверие к процессу принятия управленческих решений на основе их прогнозов.

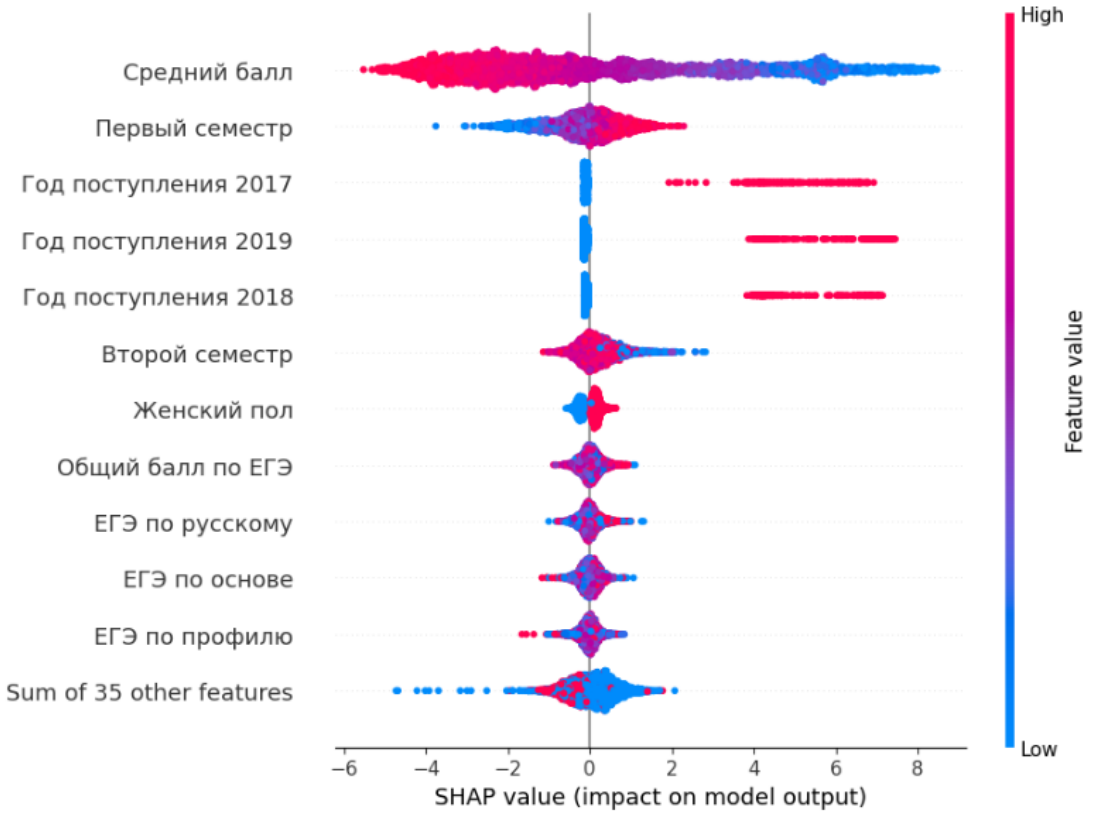


Рис. 4. Сводный график (beeswarm plot) SHAP для модели XGBoost

Метод SHAP предоставляет возможность не только построить графическую визуализацию влияния различных признаков на прогнозы модели для отдельных объектов данных, но и получить представление важности каждого фактора на основе полного набора данных (глобальные объяснения). На рис. 4 показано поведение модели на глобальном уровне на основе графика, который называется «beeswarm plot». На таком типе графиков синие точки означают объекты с малым значением признака, красные – с большим значением признака. Данный график ранжирует параметры по степени их влияния на прогнозы модели сверху вниз от

наибольшего к наименьшему. Графики такого типа помогают понять влияние каждого фактора на прогноз модели в зависимости от расположения точек, их разброса и плотности. Широкий разброс или высокая плотность точек указывают на более значительную изменчивость или более существенное влияние на прогнозы модели. Например, анализируя сводный график на рис. 3, можно прийти к выводу, что высокие оценки за первый семестр и в то же время низкие оценки за второй семестр и низкий средний бал за последующие семестры, скорее всего, могут привести к досрочному отчислению студента из вуза.

Чтобы определить, какие признаки, в целом, являются наиболее важными для прогнозов, выдаваемых моделью, можно использовать функцию shap.plots.bar инструмента SHAP, отражающую результаты усреднения значений Шепли по всем наблюдениям. На рис. 5 представлены гистограммы важности факторов на основе их средних значений Шепли для двух моделей. Первая модель была обучена на данных студентов женского пола, а вторая модель – на данных студентов мужского пола. Графики предоставляют возможность наглядно сравнить, какие факторы оказывают наибольшее значения на прогнозы обученных моделей. Для модели XGBoost со значительным перевесом наибольшую значимость на прогнозы модели оказывает общий средний балл за все семестры обучения, следующий по важности фактор — это средний балл за первый семестр.

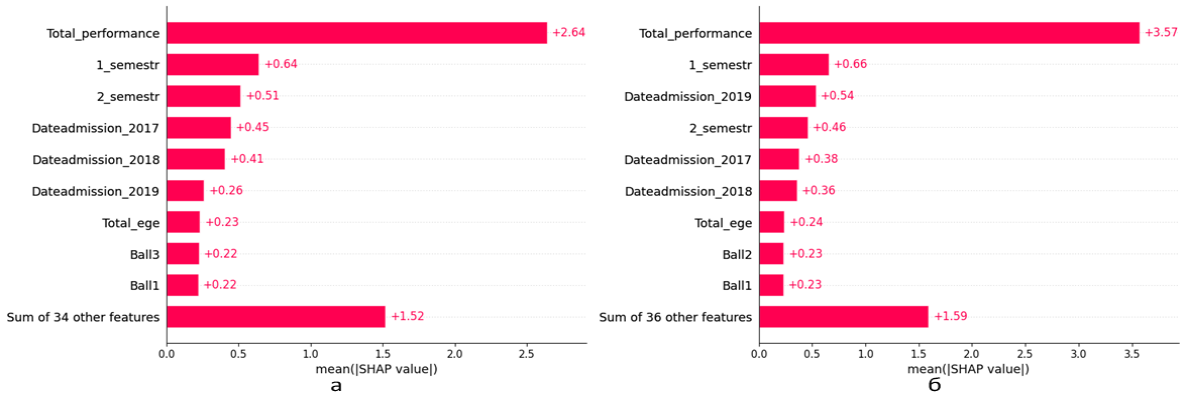


Рис. 5. Столбчатая диаграмма (shap.plots.bar), отражающую результаты усреднения SHAP-значений для модели XGBoost по всем объектам (а – студенты женского пола, б – студенты мужского пола)

ЗАКЛЮЧЕНИЕ

В работе проведено исследование возможностей применения методов машинного обучения в образовательной аналитике на основе построения и обучения моделей досрочного отчисления студентов из университета. Построенные модели способны с высокой точностью определять студентов в группе риска. Основное внимание было сосредоточено на применении методов интерпретации прогнозов обученных моделей. С помощью данных методов удалось выявить наиболее важные входные признаки модели – характеристики студентов, которые максимально влияют на вероятность их досрочного отчисления.

Методы интерпретации моделей помогают не только понять принципы работы моделей машинного обучение, но и проверить, правильно ли соотносится наше собственное мышление с выводами, полученными с их помощью. На основе выявленных факторов можно настроить эффективный процесс обучения, подстраиваться под индивидуальные характеристики студентов и их потребности.

Благодарности

Работа выполнена за счет средств Программы стратегического академического лидерства Казанского (Приволжского) федерального университета ("ПРИОРИТЕТ-2030").

СПИСОК ЛИТЕРАТУРЫ

1. *Груздев И. А., Горбунова Е. В., Фрумин И. Д.* Студенческий отсев в российских вузах: к постановке проблемы // Вопросы образования. 2013. № 2. С. 67–81. https://doi.org/10.17323/1814-9545-2013-2-67-81

2. *Терентьев Е.А., Груздев И.А., Горбунова Е.В.* Суд идёт: дискурс преподавателей об отсеве студентов // Вопросы образования. 2015. № 2. С. 129–151. https://doi.org/10.17323/1814-9545-2015-2-129-151

3. *Горбунова Е.В.* Выбытия студентов из вузов: исследования в России и США // Вопросы образования. 2018. № 1. С. 110–131. https://doi.org/10.17323/1814-9545-2018-1-110-131

4. *Горбунова Е.В.* Влияние адаптации первокурсников к университету на вероятность их отчисления из вуза // Universitas. Журнал о жизни университетов. 2013. № 2 (1). С. 59–84.

5. *Климова Т.А., Ким А.Т., Отт М.А.* Индивидуальные образовательные траектории студентов как условие качественного университетского образования // Университетское управление: практика и анализ. 2023. 27 (1). С. 23–33. https://doi.org/10.15826/umpa.2023.01.003

6. *Мещеряков А.О., Баянова Н.А., Калинина Е.А., Денисов В.А.* Предикторы выбытия студентов медицинского вуза // Медицинское образование и профессиональное развитие. 2022. № 3 (47). URL: https://www.medobr.ru/ru/jarticles/736.html?SSr=0101348cba14ffffffff27c__07e60b0e0e0130-1843 (дата обращения 01.03.2024)

7. *Шмелева Е. Д.* Факторы отсева студентов инженерно-технического профиля в российских вузах // Вопросы образования. 2020. № 3. С. 110–136.

8. *Мухамадиева К.Б.* Машинное обучение в совершенствовании образовательной среды // Образование и проблемы развития общества. 2020. № 4 (13). С. 70–77.

9. *Shrikumar A., Greenside P., Kundaje A.* Learning important features through propagating activation differences // ICML'17. 2017. P. 3145–3153. URL: https://proceedings.mlr.press/v70/shrikumar17a.html (дата обращения 01.03.2024)

10. *Apley D. W., Jingyu Zhu* Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models // Journal of the Royal Statistical Society Series B: Statistical Methodology. 2020. No 4 (82). P. 1059–1086. https://doi.org/10.1111/rssb.12377

11. *Linardatos P., Papastefanopoulos V., Kotsiantis S.* Explainable AI: A Review of Machine Learning Interpretability Methods // Entropy. 2020. https://doi.org/10.3390/e23010018

12. *Rachha A., Seyam M.* Explainable AI in Education: Current Trends, Challenges, And Opportunities // SoutheastCon. 2023. P. 232–239. https://doi.org/10.1109/SoutheastCon51012.2023.10115140

13. *Fan F.L., Xiong J., Li M., Wang G.* On Interpretability of Artificial Neural Networks: A Survey // IEEE Trans Radiat. Plasma Med Sci. 2021. No. 5 6). P. 741–760. https://doi.org/10.1109/trpms.2021.3066428

14. *Fiore U.* Neural Networks in the Educational Sector: Challenges and Opportunities // Balkan Region Conference on Engineering and Business Education. 2019. No 1 (1). P. 332–337. https://doi.org/10.2478/cplbu-2020-0039

15. *Montavon G., Samek W., Müller K.-R.* Methods for interpreting and understanding deep neural networks // Digital Signal Processing. 2018. No. 73. P. 1–15. https://doi.org/10.1016/j.dsp.2017.10.011

16. *Saranya A., Subhashini R.* A systematic review of Explainable Artificial Intelligence models and applications: Recent developments and future trends // Decision Analytics Journal. 2023. No. 7. https://doi.org/10.1016/j.dajour.2023.100230

17. *Murdoch W.J., Singh C., Kumbier K., Abbasi-Asl R., Yu B.* Definitions, methods, and applications in interpretable machine learning // Proceedings of the National Academy of Sciences. 2019. No. 16 (44). P. 22071–22080

18. *Meyer Lauritsen S. et al.* Explainable artificial intelligence model to predict acute critical illness from electronic health records // Nature Communications. 2020. № 11 (1).

19. *Linden T., Jong J., Lu C., Kiri V., Haeffs K., Fröhlich H.* An explainable multimodal neural network architecture for predicting epilepsy comorbidities based on administrative claims data // Frontiers in Artificial Intelligence. 2021. No. 4. https://doi.org/10.3389/frai.2021.610197

20. *Lu Y., Murzakhanov I., Chatzivasileiadis S.* Neural network interpretability for forecasting of aggregated renewable generation // In Proceedings of 2021 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids. 2021. P. 282–288.

21. *Mai-Anh T. Vu et al.* A shared vision for machine learning in neuroscience // Journal of Neuroscience. 2018. 38 (7). P. 1601–1607.

22. *Sundararajan M., Taly A., and Yan Q.* Axiomatic attribution for deep networks // CoRR. 2017. URL: https://arxiv.org/abs/1703.01365

23. *Kokhlikyan N. et al.* Captum: A unified and generic model interpretability library for pytorch // CoRR. 2020. URL: https://arxiv.org/abs/2009.07896 (дата обращения 01.03.2024)

24. *Lundberg S.M., Lee S.-I.* A unified approach to interpreting model predictions // Advances in Neural Information Processing Systems. 2017. No. 30. P. 4765–4774.

25. *Linardatos P., Papastefanopoulos V., Kotsiantis S.* Explainable AI: A Review of Machine Learning Interpretability Methods // Entropy. 2020. https://doi.org/10.3390/e23010018

26. *Sahakyan M., Aung Z., Rahwan T.* Explainable Artificial Intelligence for Tabular Data: A Survey // IEEE Access. 2021. No. 9. P. 135392–135422. https://doi.org/10.1109/ACCESS.2021.3116481

27. *Шобонов Н.А., Булаева М.Н., Зиновьева С.А.* Искусственный интеллект в образовании // Проблемы современного педагогического образования. № 79 (4). 2023. С. 288–290.

28. *Khosravi H. et al.* Explainable Artificial Intelligence in education // Computers and Education: Artificial Intelligence. 2022. No. 3. https://doi.org/10.1016/j.caeai.2022.100074

29. *Гафаров Ф.М., Руднева Я.Б., Шарифов У.Ю.* Прогностическое моделирование в высшем образовании: определение факторов академической успеваемости // Высшее образование в России. 2023. Т. 32. № 1. С. 51–70. https://doi.org/10.31992/0869-3617-2023-32-1-51-7

ANALYSING MACHINE LEARNING MODELS BASED ON EXPLAINABLE ARTIFICIAL INTELLIGENCE METHODS IN EDUCATIONAL ANALYTICS

D. A. Minullin[1] [0000-0001-7713-5251], F. M. Gafarov[2] [0000-0003-4704-154X]

[1, 2]*Kazan federal university*

[1]minullin.dima@mail.ru, [2]fgafarov@yandex.ru

Abstract

The problem of predicting early dropout of students of Russian universities is urgent and therefore requires the development of new innovative approaches to solve it. To solve this problem, it is possible to develop predictive systems based on the use of student data, available in the information systems of universities. This paper investigates machine learning models for predicting early student dropout trained on the basis of student characteristics and performance data. The main scientific novelty of

the work lies in the use of explainable AI methods to interpret and explain the performance of the trained machine learning models. The Explainable AI methods allow us to understand which of the input features (student characteristics) have the greatest influence on the results of the machine learning models. (student characteristics) have the greatest influence on the prediction results of trained models, and can also help to understand why the models make certain decisions. The findings expand the understanding of the influence of various factors on early dropout of students.

***Keywords**: educational analytics, data mining, machine learning, explainable AI*

REFERENCES

1. *Gruzdev I.A., Gorbunova E.V., Frumin I.D.* Studencheskij otsev v rossijskih vuzah: k postanovke problemy [Student dropout in russian higher education institutions: the problem statement] // Educational Studies Moscow. 2013. № 2. P. 67–81. https://doi.org/10.17323/1814-9545-2013-2-67-81

2. *Terentyev E.A., Gruzdev I.A., Gorbunova E.V.* Sud idyot: diskurs prepodavatelej ob otseve studentov [The Court Is Now in Session: Professor Discourse on Student Attrition] // Educational Studies Moscow. 2015. №2. S. 129–151. https://doi.org/10.17323/1814-9545-2015-2-129-151

3. *Gorbunova E.V.* Vybytiya studentov iz vuzov: issledovaniya v Rossii i SSHA [Research on Student Departure in Russia and the U.S.] // Educational Studies Moscow. 2018. №1. S. 110–131. https://doi.org/10.17323/1814-9545-2018-1-110-131

4. *Gorbunova E.V.* Vliyanie adaptacii pervokursnikov k universitetu na veroyatnost' ih otchisleniya iz vuza [The Effect of Adaptation of Freshmen to the University on the Probability of Their Expulsion from the University] // Universitas. 2013. № 2 (1). S. 59–84.

5. *Klimova T.A., Kim A.T., Ott M.A.* Individual'nye obrazovatel'nye traektorii studentov kak uslovie kachestvennogo universitetskogo obrazovaniya [Individual Educational Trajectories as a Condition for High-Quality University Education] // University Management: Practice and Analysis. 2023. 27(1). S. 23–33. https://doi.org/10.15826/umpa.2023.01.003

6. *Meshcheryakov A.O, Bayanova N.A., Kalinina E.A.* Prediktory vybytiya studentov medicinskogo vuza // Medicinskoe obrazovanie i professional'noe razvitie. 2022. № 3 (47).
URL: https://www.medobr.ru/ru/jarticles/736.html?SSr=0101348cba14ffffffff27c__07e60b0e0e0130-1843

7. *Shmeleva E.D.* Faktory otseva studentov inzhenerno-tekhnicheskogo profilya v rossijskih vuzah [Factors of Attrition among Computer Science and Engineering Undergraduates in Russia] // Educational Studies Moscow. 2020. №3. S. 110–136.

8. *Mukhamadieva K.B.* Mashinnoe obuchenie v sovershenstvovanii obrazovatel'noj sredy // Obrazovanie i problemy razvitiya obshchestva. 2020. № 4(13). S. 70–77.

9. *Shrikumar A., Greenside P., Kundaje A*. Learning im-portant features through propagating activation differences // ICML'17. 2017. P. 3145–3153. URL: https://proceedings.mlr.press/v70/shrikumar17a.html

10. *Apley D.V., Zhu J.* Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models // Journal of the Royal Statistical Society Series B: Statistical Methodology. 2020. No. 4 (82). P. 1059–1086. https://doi.org/10.1111/rssb.12377

11. *Linardatos P., Papastefanopoulos V., Kotsiantis S.* Explainable AI: A Review of Machine Learning Interpretability Methods // Entropy. 2020.
https://doi.org/10.3390/e23010018

12. *Rachha A., Seyam M.* Explainable AI In Education: Current Trends, Challenges, And Opportunities // SoutheastCon. 2023. P. 232–239.
https://doi.org/10.1109/SoutheastCon51012.2023.10115140

13. *Fan F.L., Xiong J., Li M., Wang G.* On Interpretability of Artificial Neural Networks: A Survey // IEEE Trans Radiat Plasma Med Sci. 2021. No. 5 (6). P. 741–760. https://doi.org/10.1109/trpms.2021.3066428

14. *Fiore U.* Neural Networks in the Educational Sector: Challenges and Opportunities // Balkan Region Conference on Engineering and Business Education. 2019. No. 1 (1). P. 332–337. https://doi.org/10.2478/cplbu-2020-0039

15. *Montavon G., Samek W., Müller K.-R.* Methods for interpreting and understanding deep neural networks // Digital Signal Processing. 2018. No. 73. P. 1–15. https://doi.org/10.1016/j.dsp.2017.10.011

16. *Saranya A., Subhashini R.* A systematic review of Explainable Artificial Intelligence models and applications: Recent developments and future trends // Decision Analytics Journal. 2023. No. 7. https://doi.org/10.1016/j.dajour.2023.100230

17. *Murdoch W.J., Singh C., Kumbier K., Abbasi-Asl R., Yu B.* Definitions, methods, and applications in interpretable machine learning // Proceedings of the National Academy of Sciences. 2019. No. 16 (44). P. 22071–22080

18. *Lauritsen S.M. et al.* Explainable artificial intelligence model to predict acute critical illness from electronic health records // Nature Communications. 2020. No. 11 (1).

19. *Linden T., Jong J., Lu C., Kiri V., Haeffs K., Fröhlich H.* An explainable multimodal neural network architecture for predicting epilepsy comorbidities based on administrative claims data // Frontiers in Artificial Intelligence. 2021. No. 4. https://doi.org/10.3389/frai.2021.610197

20. *Lu Y., Murzakhanov I., Chatzivasileiadis S.* Neural network in-terpretability for forecasting of aggregated renewable generation // In Proceedings of 2021 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids. 2021. P. 282–288.

21. *Mai-Anh T. Vu et al.* A shared vision for machine learning in neuroscience // Journal of Neuroscience. 2018. 38(7). P. 1601–1607.

22. *Sundararajan M., Taly A., and Yan Q.* Axiomatic attribution for deep networks // CoRR. 2017. URL: https://arxiv.org/abs/1703.01365

23. *Kokhlikyan N. et al.* Captum: A unified and generic model interpretability library for pytorch // CoRR. 2020. URL: https://arxiv.org/abs/2009.07896 (last access 01.03.2024)

24. *Lundberg S.M., Lee S.-I.* A unified approach to interpreting model predictions // Advances in Neural Information Processing Systems. 2017. No. 30. P. 4765–4774.

25. *Linardatos P., Papastefanopoulos V., Kotsiantis S.* Explainable AI: A Review of Machine Learning Interpretability Methods // Entropy. 2020. https://doi.org/10.3390/e23010018

26. *Sahakyan M., Aung Z., Rahwan T.* Explainable Artificial Intelligence for Tabular Data: A Survey // IEEE Access. 2021. No. 9. P. 135392-135422.

https://doi.org/10.1109/ACCESS.2021.3116481

27. *Shobonov N.A., Bulaeva M.N., Zinovieva S.A.* Iskusstvennyj intellekt v obrazovanii [Artificial Intelligence in Education] // Problemy sovremennogo pedagogicheskogo obrazovaniya. № 79 (4). 2023. S. 288–290.

28. *Khosravi H. et al.* Explainable Artificial Intelligence in education // Computers and Education: Artificial Intelligence. 2022. No. 3.
https://doi.org/10.1016/j.caeai.2022.100074

29. *Gafarov F.M., Rudvena Ya.B., Sharifov U.Yu.* Prognosticheskoe modelirovanie v vysshem obrazovanii: opredelenie faktorov akademicheskoj uspevaemosti [Predictive Modeling in Higher Education: Determining Factors of Academic Performance] // Higher Education in Russia. 2023. V. 32. No. 1. S. 51–70.
https://doi.org/10.31992/0869-3617-2023-32-1-51-7

СВЕДЕНИЯ ОБ АВТОРАХ

МИНУЛЛИН Дмитрий Артурович – аспирант, Казанский федеральный университет.

Dmitriy Arturovich MINULLIN – Postgraduate student, Kazan Federal University.

email: minullin.dima@mail.ru

ORCID: 0000-0001-7713-5251.

ГАФАРОВ Фаиль Мубаракович – кандидат физ.-мат. наук, доцент, Казанский федеральный университет.

Fail Mubarakovich GAFAROV – Candidate of Science in Physics and Mathematics, Docent, Kazan Federal University.

email: fgafarov@yandex.ru

ORCID: 0000-0003-4704-154X.

Материал поступил в редакцию 2 мая 2024 года

УДК 514.112, 81, 159.9.07, 004.422.8

ИССЛЕДОВАНИЕ КОГНИТИВНОЙ ФУНКЦИИ ПРИ ГЕНЕРАЦИИ ЭЛЛИПТИЧЕСКИХ ПРЕДЛОЖЕНИЙ В ПЛАНИМЕТРИЧЕСКИХ ЗАДАЧАХ

Пархоменко В. А.[1] [0000-0001-7757-377X], Найденова К. А.[2] [0000-0003-2377-7093], Мартирова Т. А.[3] [0000-0003-0000-6608], Щукин А. В.[4] [0000-0002-9534-824X]

[1, 4]*Санкт-Петербургский политехнический университет Петра Великого, Санкт Петербург*

[2, 3]*Военно-медицинская академия, Санкт Петербург*

[1]vladimir.parkhomenko@spbstu.ru, [2]ksennaidd@gmail.com, [3]martta462@yandex.ru, [4]alexander.schukin@spbstu.ru

Аннотация

Работа посвящена изучению когнитивной функции, связанной с генерацией эллиптических предложений в русском языке. Исследование проводилось на основе тестирования этой когнитивной способности с помощью компьютерной системы, специально разработанной авторами для этой цели. Тестирование этой когнитивной способности предложено и реализовано впервые. Система является расширением Moodle и открыто размещена в репозитории github. Эллиптические конструкции ограничиваются глагольными и именными эллипсисами, которые теоретически возможно полностью восстановить на основе контекста предложения. Исследование проводилось с участием в качестве респондентов студентов СПбПУ. В качестве предметной области были выбраны тексты планиметрических задач. В результате анализа данных тестирования получены следующие результаты: установлено влияние знаний респондента предметной области (планиметрии) на результаты тестирования; обнаружена тенденция к самообучению респондентов, что проявляется в сокращении времени и увеличении баллов по мере прохождения тестов; показано, что респонденты слабо мотивированы, если не видят отзыв на ответ по выполненному заданию.

Обсуждены проблемы дальнейшего развития системы тестирования и её применения при адаптации опросников (заданий) для оценки знаний студентов СПбПУ в области автоматизации обнаружения ошибок в программах, а также диагностики функционального состояния специалистов операторского профиля и

экспресс-диагностики деменции. Перспективным представляется также применение системы для совершенствования процессов синтаксического разбора эллиптических предложений и автоматизации восстановления эллипсисов в предметной области планиметрии.

***Ключевые слова**: онлайн-система тестирования, разработка системы тестирования, когнитивная функция, эллипсис, планиметрия*

ВВЕДЕНИЕ

Первоначальная идея создания системы тестирования когнитивной функции (КФ), связанной с формированием эллиптических предложений в устной и письменной речи, возникла из потребности разработки тестов когнитивных нарушений (КН) для ранней диагностики деменции. КФ генерации эллипсиса в предложении опирается на удержании в памяти человека уже сказанного (написанного, уже выполненного ментально), то есть с функциями планирования, внимания, оперативной способности идентифицировать элементы, повторение которых можно опустить, но которые слушатель (читатель) может восстановить по контексту.

Эллиптическими называются конструкции, которые содержат опущенный, но однозначно восстанавливаемый элемент в предложении (слово или сочетания слов) [1].

Основание эллипсиса мы связываем с мыслительными процессами, реализуемыми при образовании предложений. Важным является изучение связи между когнитивными операциями, то есть процессами манипулирования знаниями, и синтаксисом различных языков. Начало этому направлению работ было положено Р. Джекендоффом [2–4].

Далее рассмотрим только два типа эллипсисов: «глагольный эллипсис» и «именной эллипсис», структура которых основана на именных, глагольных и предложных группах. В литературе эти типы эллипсисов известны: первый как эллипсис глагольной группы (исключение глагола, как изолированного, так и в составе глагольных групп и целых клауз), второй – как эллипсис составляющих именной группы с «сохранением представителя» [5].

Когнитивный подход предполагает, что:

1. полное предложение описывает некоторую когнитивную ситуацию, например, геометрическую конфигурацию, выраженную в тексте задачи по планиметрии;
2. полное предложение мысленно трансформируется в неполное эллиптическое предложение; причем в основе трансформации лежат когнитивные операции, выполняемые по определенным правилам.

Напомним некоторые правила генерации предложений с глагольным и именным эллипсисами [6].

Правило 1. Если имеется ввиду одно и то же действие над несколькими объектами, то после описания этого действия над первым объектом в предложении далее это действие над другими объектами может быть описано без копирования наименования этого действия (пропускается глагол).

Правило 2. Глагол может входить в предложение в составе глагольной группы (ГГр), тогда после полного описания действия над первым объектом возможно в последующих описаниях того же действия над другими объектами опускать не только глагол, но и повторяющиеся составляющие глагольных актантов.

Правило 3. Объект может быть выражен в составе именной группы (ИГ), тогда при упоминании в предложении того же самого объекта возможен пропуск не только его наименования, но и общих повторяющихся составляющих ИГ, характеризующих объект.

Рассмотрим некоторые примеры:

Дана трапеция ABCD с основанием AD. Биссектрисы внешних углов при вершинах A и B пересекаются в точке P, а при вершинах C и D – в точке Q.

Во втором предложении, часть ИГ «биссектрисы внешних углов» и глагол «пересекаются» опущены.

В результате первой операции он должен был добиться уничтожения врага к западу от реки и выйти к ней, в результате второй – создать плацдармы, а затем провести вторжение в глубь Германии (У. Черчилль. Вторая мировая война. Том 6. Часть 2).

ИДЕЯ ПРОЦЕССА ТЕСТИРОВАНИЯ

Задание (вопрос) при тестировании заключается в том, чтобы по примеру с глагольным эллипсисом преобразовать данное в предложение без эллипсиса в

эллиптическое. Тест содержит 10 таких заданий.

Тестирование предваряется текстом с пояснением понятия эллипсиса и примером преобразования некоторого предложения в эллиптическое, например:

«*В равнобедренный прямоугольный треугольник ABC с прямым углом при вершине B вписан прямоугольник MNKB так, что две его стороны MB и KB лежат на катетах, а вершина N лежит на гипотенузе AC.*»

Можно сократить повторение глагола «лежит» следующим образом:

«*В равнобедренный прямоугольный треугольник ABC с прямым углом при вершине B вписан прямоугольник MNKB так, что две его стороны MB и KB лежат на катетах, а вершина N ~~лежит~~ на гипотенузе AC.*»

Далее описан пример задания, который видит учащийся:

«Кликните по избыточным (повторяющимся) словам для их вычеркивания:

В круговой сектор OPQ, центральный угол которого меньше 180 градусов, вписан квадрат ABCD так, что вершины A и B принадлежат дуге сектора, а вершины C и D принадлежат его радиусам.»

КРАТКИЕ СВЕДЕНИЯ О СИСТЕМЕ ТЕСТИРОВАНИЯ

Описание реализованной системы тестирования дано в работах [7, 8]. Программный код системы размещен в открытом доступе в интернете (https://github.com/R-D4S/Plugins-for-Ellipsis-Test).

В первой версии системы мы ограничились только текстами задач по планиметрии, с наиболее четкими и простыми моделями эллипсисов.

Вариант с вычеркиванием слов избавляет нас от проверки возможных орфографических или пунктуационных ошибок при вводе. Далее перечислены общие требования к системе.

1. Наличие типа вопроса «Вычеркивание», позволяющего взаимодействовать с элементами предложения через клик.
2. Наличие возможности отслеживать время, затраченное на решение каждого вопроса.
3. Наличие возможности экспортировать результаты для дальнейшей обработки в сторонних программах.
4. Наличие однозначной идентификации пользователей.

АНАЛИЗ РЕЗУЛЬТАТОВ ПИЛОТНОГО ИССЛЕДОВАНИЯ

В данный момент проведены две сессии тестирования на основе реализованной программной системы. Респондентами были студенты технических специальностей Санкт-Петербургского политехнического университета Петра Великого (СПбПУ). После первой сессии вначале проанализированы ответы 17-ти студентов. Среднее время прохождения теста 11 минут. Средний балл респондентов 6.76 из 10 возможных (10 заданий). Таблица 1 показывает средний балл респондентов и среднее время ответа на каждый вопрос.

Таблица 1. Средний балл респондентов и среднее время ответа на каждый вопрос

Номера вопросов	1	2	3	4	5	6	7	8	9	10
Средний балл	0,56	0,79	0,53	0,74	0,47	0,74	0,82	0,89	0,72	0,72
Среднее время ответа на каждый вопрос (сек)	57,0	47,6	70,5	48,1	43,9	34,3	54,9	36,8	46,7	28,9

В дополнение к баллам были введены качественные оценки результатов прохождения теста: “Верно” (В), “Неверно” (НВ), “Частично правильно” (ЧП) и “Нет ответа” (НО). Оценка “Т” давалась, если множество слов, вычеркнутых респондентом, полностью совпадало со словами, которые должны быть вычеркнуты в правильном ответе. Оценка “ЧП” давалась, если было только частичное совпадение с вычеркиваемыми словами в правильном ответе. Эта оценка могла быть получена, если не был вычеркнут глагол, но в этом случае вычеркивания явно относились к сокращению повторяющихся фрагментов именных групп аргументов глагола. Оценка “НВ” давалась, если ни одного слова не было вычеркнуто из вычеркиваемых слов в правильном ответе. Перечисленные оценки не зависели от того, вычеркнул или нет респондент некоторые слова в предложении, не предусмотренные правильным ответом.

В таблице 2 приведены тексты для 10 заданий с выделенными словами, составляющими правильный ответ. В скобках приведена нумерация с инкрементированием на единицу, так как первым в Moodle Quiz был вопрос с пояснением задания.

Таблица 2 Тексты 10 заданий

N	Предложение в задании
1 (2)	В правильную четырехугольную пирамиду вписан куб так, что четыре его вершины принадлежат боковым ребрам пирамиды, а остальные ~~четыре его вершины принадлежат~~ плоскости ее основания.
2 (3)	В круговой сектор OPQ, центральный угол которого меньше 180 градусов, вписан квадрат ABCD так, что вершины A и B принадлежат дуге сектора, а вершины C и D ~~принадлежат~~ его радиусам
3 (4)	На стороне BC треугольника ABC взята точка A*; серединный перпендикуляр к отрезку A*B пересекает сторону AB в точке M, а серединный перпендикуляр к отрезку A*C ~~пересекает~~ сторону AC в точке N.
4 (5)	Эти равенства означают, что OB касается описанной окружности треугольника ABM, а OD ~~касается~~ описанной окружности треугольника ADM.
5 (6)	Прямая параллельная AB пересекает AC и BC в точках M и N, а прямые параллельные AC и BC ~~пересекают~~ AB в точках P и Q.
6 (7)	Дана окружность радиуса R и касательная к ней; постройте квадрат так, чтобы две его смежные вершины лежали на касательной, а две другие ~~лежали~~ на окружности и вычислите сторону квадрата.
7 (8)	Дана окружность и точки P и Q внутри неё; построить вписанный в эту окружность прямоугольный треугольник, у которого один катет проходит через точку P, а другой ~~катет проходит~~ через точку Q.
8 (9)	Окружность касается сторон AB и AD прямоугольника ABCD и пересекает сторону DC в единственной точке F, а сторону BC ~~пересекает~~ в единственной-точке E.
9 (10)	На плоскости заданы 3 точки A, B и C; построить три окружности к1, к2 и к3 так, чтобы окружности к2 и к3 касались друг друга в точке A, окружности к3 и к1 ~~касались друг друга~~ в точке B, а окружности к1 и к2 ~~касались друг друга~~ в точке C
10 (11)	Ортоцентр треугольника делит одну из высот треугольника в отношении 2 к 1, считая от вершины, а другую ~~из высот делит~~ пополам.

Оценим сложность заданий с точки зрения структуры глагольного эллипсиса. Самые простые задания – это те, в которых вычеркивается только глагол (2, 3, 4, 5, 6, 8). Более сложные предложения – те, где можно вычеркнуть и повторяющиеся фрагменты в именных группах дополнений к глаголам (1, 7, 10). Более сложные предложения – с множественным эллипсисом (9).

Таблица 3 показывает суммарные результаты тестирования по каждому вопросу, выраженные в количестве качественных оценок.

Таблица 3. Распределение качественных оценок ответов респондентов по каждому вопросу (заданию)

	Количество качественных оценок			
Номер вопроса	В	НВ	ЧП	НО
1	5	5	9	4
2	14	3	-	6
3	9	7	-	7
4	7	7	-	9
5	8	7	-	8
6	13	4	-	6
7	13	1	4	5
8	12	6	-	5
9	10	4	4	5
10	9	-	8	6
Sum	110	42	25	61

Анализ Таблицы 3 показывает, что наибольшее число ответов правильные. Отсутствие ответов (довольно большое число) можно объяснить необычностью заданий для многих респондентов – студентов технического профиля (математиков и программистов).

Анализ результатов 17-ти респондентов был произведен вручную. Благодаря этому было возможно сделать некоторые важные наблюдения.

1. Изучение времени ответов респондентов (в сек.) на каждое задание показало, что некоторые задания требовали большего времени на выполнение. Можно предположить, что именно эти задания вызывали у респондента трудности. Так, резкое увеличение времени ответа вызывало задание 3 (4). Рассмотрим

ответ на это задание респондента, одного из лучших по результатам прохождения теста:

«На стороне BC треугольника ABC взята точка A*; серединный перпендикуляр к отрезку A*B пересекает сторону AB в точке M, а серединный перпендикуляр к отрезку A*C пересекает сторону AC в точке N».

В этом случае ответ респондента нельзя рассматривать как правильный, хотя он вычеркнул глагол, однако дополнительное вычеркивание слов "серединный перпендикуляр" неправильное с точки зрения геометрической ситуации, так как рассматривается серединный перпендикуляр к другому отрезку. В этом случае респондент допускает простое удаление повторяемого фрагмента текста с нарушением смысла предложения.

Аналогичная трудность связана также с вопросами 4 (5) и 9 (10):

«4 (5) Эти равенства означают, что OB касается описанной окружности треугольника ABM, а OD касается описанной окружности треугольника ADM.»

В этом предложении нельзя вычеркивать повторяющееся слова «описанной окружности», так как геометрически речь идет об описанной окружности другого треугольника.

9 (10) «Окружность касается сторон AB и AD прямоугольника ABCD и пересекает сторону DC в единственной точке F, а сторону BC пересекает в единственной точке E».

Запрещенный вариант – вычеркивание повторяющегося слова «единственной».

Таким образом, выявлена связь КФ генерации эллипсиса не только с синтаксической структурой предложения, но и с предметной областью (в данном случае, планиметрией).

2. Также отметим, что ответы некоторых респондентов имели тенденцию к обучению, так как время ответа постепенно снижалось от вопроса к вопросу, а оценки в баллах не уменьшались.

Один респондент выразил желание пройти тест три раза. Таблица 4 показывает среднее время (в сек.), потраченное в трех последовательных прохождениях теста.

Таблица 4. Среднее время, затраченное на выполнение теста в трех последовательных прохождениях

Номер прохождения теста	Среднее время, затраченное на выполнение теста (сек)
1-ый проход	112
2-ой проход	7,6
3-ий проход	6,1

ПОВТОРНОЕ ИССЛЕДОВАНИЕ

Для увеличения статистики по тестовому материалу было решено повторить эксперимент. Однако на этом шаге было внесено изменение в пояснение к каждому заданию, акцентируя внимание на сохранение смысла предложения: «Кликните на словах, которые Вы считаете возможным вычеркнуть без изменения смысла предложения».

Первая сессия проходила в мае 2023 (без изменения пояснения), а вторая – в декабре 2023 года, при этом пилотный эксперимент проходил в начале мая, поэтому мы объединили данные в один набор MAY-23. Изменение текста пояснения не показали значимого изменения в качестве ответов. Второй набор данных обозначается далее DEC-23, а объединенный набор данных – MAY&DEC-23.

АВТОМАТИЗАЦИЯ АНАЛИЗА РЕЗУЛЬТАТОВ ТЕСТИРОВАНИЯ

Система разработанных плагинов и тестового материала размещена в Moodle Quiz, где есть встроенная система первичной обработки статистических данных. Однако для разработанного нами нового типа вопроса «вычеркивание» указание учета правильных ответов проводится не совсем корректно (см. рис. 1) и нуждается в доработке. На этапе тестирования продукта это оказалось нецелесообразным, т. к. само понятие «правильного» ответа находится на стадии калибровки.

Рассмотрим подробнее рис. 1. На нем показан анализ ответов на 1 (2) вопрос. Видно, что фактический ответ соотнесен только первому по списку слову (модель ответа «четыре+19»), которое входит в состав правильного ответа. Однако последующим моделям ответа фактические ответы не сопоставляются (например, к «его+20», выделенный красным квадратом).

Анализ ответов

Модель ответа	Фактический ответ	Частичный кредит	Количество	Частота
четыре+19	остальные+18,его+20,её+24,четыре+9	25,00%	1	2,56%
	пирамиды+15,принадлежат+22,четыре+19,его+20	25,00%	1	2,56%
его+20		25,00%	0	0,00%
вершины+21		25,00%	0	0,00%
принадлежат+22		25,00%	0	0,00%
[Не соответствует ни один ответ]		0,00%	0	0,00%
Без ответа		0,00%	0	0,00%

Рис. 1. Пример учета частоты фрагментов ответа встроенными в Moodle Quiz средствами статистической обработки

Кроме того, исходя из специфики поставленных задач, потребовалось сформировать запросы на языке SQL к БД Moodle для извлечения отдельных частей данных, например, времени ответа на каждый вопрос. Дальнейшая обработка данных осуществлялась на языке Python. После выгрузки данных в табличном формате стандартным и разработанным способом были произведены объединение таблиц и формирование новых столбцов, как-то:

- «суммарное время ответов на вопросы»;

- «неверные содержательные вычеркивания для 4, 5 и 9 вопросов» и другие.

С целью сокращения аномальных выбросов в диаграммах рассеивания различных показателей был подобран следующий фильтр для отсеивания ответов (анкет учащихся): «суммарное время 10-ти ответов > 1200 сек, всего отвечено не меньше 6 вопросов, всего сумма баллов не меньше 2.5». По наборам данных получены следующие характеристики:

- MAY-23: приняли участие 61, отфильтровано 7, осталось 54 человека;

- DEC-23: приняли участие 38, отфильтровано 12, осталось 26 человека;

- MAY&DEC-23: приняли участие 99, отфильтровано 21, осталось 78 человека.

ОСНОВНЫЕ РЕЗУЛЬТАТЫ ВТОРОГО ЭТАПА ТЕСТИРОВАНИЯ

После очистки данных от аномалий и формирования дополнительных столбцов были отобраны следующие показатели для проведения корреляционного анализа: общее время прохождения теста (totaltime); время, затраченное на 10 вопросов (totaltime2); итоговые баллы (sumgrades), гендер (genderbin), короткий или полный тексты пояснения (descrBin); вычеркнуто слов (responsed); вычеркнуто слов правильно (StrikedRight). Соответствующая корреляционная матрица приведена на рис. 2.

	totaltime	totaltime2	sumgrades	genderbin	descrBin	responsed
totaltime	1.000000	0.815430	-0.059588	0.082925	-0.002790	-0.041363
totaltime2	0.815430	1.000000	-0.147950	0.168719	-0.080620	0.219409
sumgrades	-0.059588	-0.147950	1.000000	-0.117843	0.001591	0.259291
genderbin	0.082925	0.168719	-0.117843	1.000000	0.173826	-0.052415
descrBin	-0.002790	-0.080620	0.001591	0.173826	1.000000	0.093176
responsed	-0.041363	0.219409	0.259291	-0.052415	0.093176	1.000000
StrikedRight	0.003528	0.062390	0.858449	-0.117787	-0.067662	0.378618

Рис. 2. Корреляционная матрица с учетом «наивного» способа оценивания итогового балла

Под «наивным» способом оценивания итогового балла мы понимаем способ, описанный в начале раздела с пилотным экспериментом и отраженный в оценке sumgrades. Видно, что есть сильная корреляция между этим показателем и правильно вычеркнутыми (0,858), однако, как это мы описываем далее, есть необходимость использования других систем расчета итоговой оценки.

Тем не менее, опишем основные тенденции, которые наблюдаются на этом графике: немного больше времени ответов на вопросы тратят мужчины (корреляция 0.168); женщины отвечают на вопросы несколько лучше (корреляция –0.117).

Более интересную закономерность демонстрирует рис. 3, на котором

видно, что каждый последующий ответ в целом дается респондентам быстрее, чем предыдущий (за исключением 9 (10) и 3 (4) вопросов, которые, по-видимому, являются сложными).

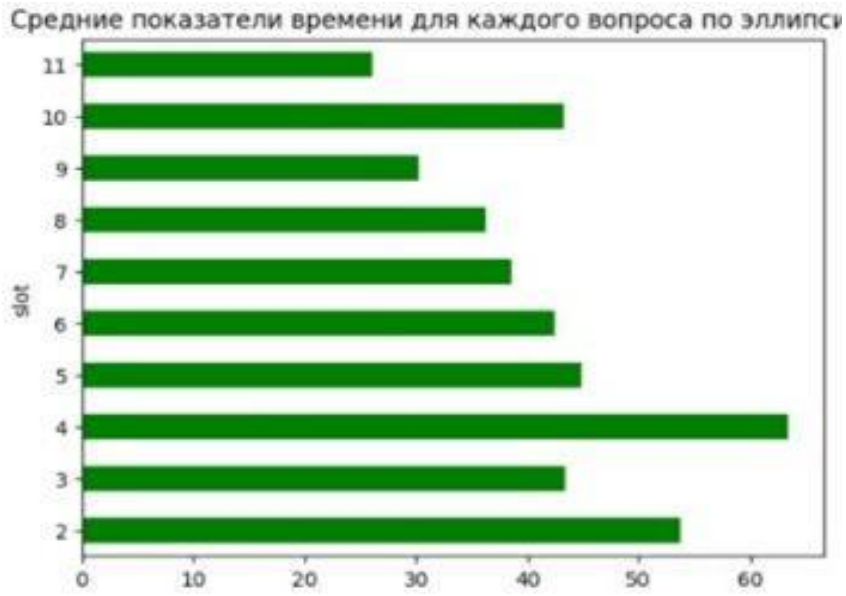


Рис. 3. Тенденция уменьшения времени ответов на вопросы

На рис. 4 для этих же вопросов видно, что последняя часть вопросов дается учащимся лучше (выше средний балл).

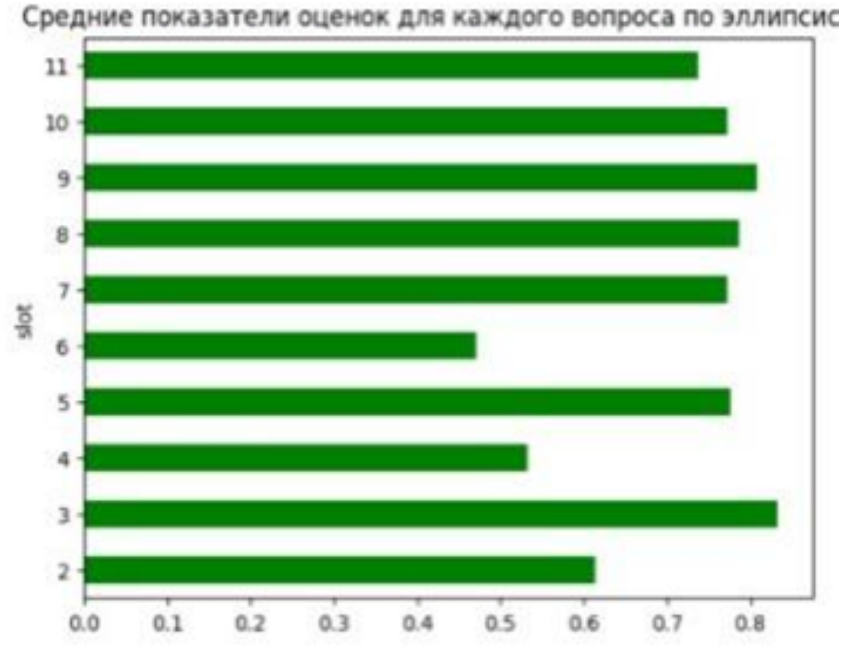


Рис. 4. Тенденция увеличения баллов за ответы на вопросы

Отметим, что закономерности, наблюдаемые в MAY-23 в целом, повторяются в DEC-23, т. е. дублирование информации, что “нахождение эллипсиса (вычеркивание) не должно изменять смысл предложения” не привело к существенным улучшениям итоговых баллов.

Кроме того, сформированы «интересные» подгруппы по фильтрам MAY&DEC-23:

Не нашли более 10 правильных слов (из 21): 13 человек. Вычеркнули более 12 “лишних” слов: 22 человека. Одновременно по двум критериям: 9 человек, причем 7 из них из DEC-23.

Наличие таких подгрупп свидетельствует о слабой мотивации учащихся.

Анализ результатов тестирования показал, что респонденты часто связывают эллиптичность с избавлением от любой избыточности в тексте. Однако они правильно используют как когнитивные правила, представленное в [6], так и знания из области планиметрии. Например, в предложении:

«Окружность касается сторон AB и AD прямоугольника ABCD и пересекает сторону DC в единственной точке F, а ~~сторону~~ BC ~~пересекает~~ в единственной точке E.»

Респондент вычеркнул слово “сторону”, так как обозначение BC, с учетом предыдущей части предложения, говорит, что BC есть сторона треугольника.

Теперь в каждом предложении добавляется множество слов, не связанных с эллипсисом, которые допустимо вычеркивать. Анализ совершённых допустимых вычеркиваний, как в ниже приведенном примере, дает возможность более детально изучать когнитивные методы сокращения избыточности в текстах.

«На плоскости заданы 3 точки A, B и C; построить три окружности к1, к2 и к3 так, чтобы окружности к2 и к3 касались друг друга в точке A, <u>окружности</u> к3 и к1 касались друг друга в точке в точке B, а <u>окружности</u> к1 и к2 касались друг друга в точке C.»

ВЫВОДЫ И БУДУЩИЕ ШАГИ ПО СОВЕРШЕНСТВОВАНИЮ СИСТЕМЫ И ЕЁ ПРИМЕНЕНИЯ

На использованном нами «наивном» подходе к итоговой оценке видно, что учащиеся имеют трудности с мотивацией к прохождению теста. Одновременно с

этим обнаружено, что нахождение эллипсиса сильно зависит от знаний предметной области, и по сути оцениванию должны подвергаться как знание предметной области, так и способность формирования эллипсисов (двухкритериальный подход). В связи с этим мы решили разрабатывать опросники под конкретную целевую аудиторию.

К сожалению, выбранный нами фильтр для отсеивания аномальных выбросов по диаграммам рассеивания учитывает «сумму баллов», рассчитанную по «наивной» схеме, приведенной в пилотном исследовании. Поэтому во время внедрения альтернативных способов расчета, например, на базе двухкритериального подхода, придется заново подбирать фильтр. К счастью, весь процесс исследования автоматизирован, что позволит это сделать относительно оперативно.

Отдельным образом должен быть исследован подход к учету встроенных в Moodle Quiz средств калибровки опросников, включая веса вопросов и другие метрики, см. рис. 5. К сожалению, как это было описано ранее (см., например, рис. 1), учет частоты ответа не приводится в Moodle Quiz корректно, поэтому и подсчет статистики по вопросам должен быть верифицирован на предмет достоверности расчета. В худшем случае необходимо будет переписать расчет данной статистики в автоматизируемом расчете на Python.

Анализ структуры теста

Скачать табличные данные как | Значения, разделяемые запятыми (.csv) | Скачать

Название вопроса	Попытки	Индекс легкости	Стандартное отклонение	Балл случайного угадывания	Намеченный вес	Эффективный вес	Индекс дискримина
задание 1	39	53,21%	27,01%	0,00%	10,00%	8,27%	63,86%
задание 2	39	43,59%	50,24%	0,00%	10,00%	11,33%	58,85%

Рис. 5. Фрагмент предложений по калибровке вопросов встроенными в Moodle Quiz средствами статистической обработки

Возможный подход к оцениванию итоговых баллов

Рассмотрим, как можно изменить качественную оценку по каждому вопросу с учетом слов, которые нельзя вычеркнуть.

Будем считать, что у нас два независимых критерия оценивания. Назовем

первый – Эллипсис, а второй – Геометрия.

Если нет ответа на вопрос, оценка по Эллипсис и Геометрии не производится.

Оценки по Эллипсис не изменяются, их четыре: Верно (В), Неверно (НВ), Частично правильно (ЧП), Нет ответа (НО).

Оценка Геометрия формируется следующим образом:

Задается список слов, которые нельзя вычеркнуть;

Если хоть одно слово из этого списка вычеркнуто, – оценка Геометрия = Неверно (НВ)

Иначе – оценка = Верно (В).

Поскольку оценки по Эллипсис и Геометрия независимы, то мы имеем следующие сочетания оценок:

Верно Э + Верно Г = В;

Верно Э + Неверно Г = ВЭ + НВГ

Неверно Э + Верно Г = НВЭ + ВГ;

Неверно Э + Неверно Г = НВЭ + НВГ;

ЧП Э + Верно Г = ЧПЭ + ВГ = ЧП;

ЧП Э + Неверно Г = ЧПЭ + НГ;

Общий итог (возможный вариант):

Если есть хотя бы НГ или НВЭ, то общий итог – НЕВЕРНО;

Если имеем ВЭ + ВГ, то общий итог – ВЕРНО;

Если ЧПЭ + ВГ, то итог ЧАСТИЧНО ПРАВИЛЬНО.

ЗАКЛЮЧЕНИЕ

В работе проанализированы результаты проведения экспериментов в двух промежутках времени. На основе наивного подхода к оцениванию сформирован фильтр для минимизации аномальных выбросов показателей на диаграмме рассеивания. Сформированы запросы к БД Moodle, получены новые столбцы в таблице данных, например, время выполнения 10 вопросов теста, проведен корреляционный анализ на выборке в 78 респондентов. Выявлены тенденции к сокращению количества времени на вопросы при неухудшении среднего количества баллов. Все расчеты автоматизированы в виде скриптов на Python.

Приведена гипотеза, что способность распознавания ответов сильно зависит от знаний предметной области. Кроме того, выявлена сильная зависимость мотивации и качества ответов. В связи с этим авторы полагают, что работа с целевыми аудиториями и востребованными для этих людей темами поможет увеличить качество собираемого материала. В частности, предполагается собирать данные в рамках:

- курса «Методы тестирования программного обеспечения»;
- диагностики функционального состояния специалистов операторского профиля под нужды заказчика;
- экспресс-диагностики деменции;
- задач по геометрии для школьников.

СПИСОК ЛИТЕРАТУРЫ

1. *Падучева Е.В.* О семантике синтаксиса. М.: URSS, 2019. 296 с.
2. *Jackendoff R.* Semantic and cognition. Cambridge: MIT Press, 1983. 283 p.
3. *Jackendoff R.* Information is in the mind of the beholder // Linguistic and Philosoph. 1985. V. 8. No. 1. P. 23–33.
4. *Jackendoff R.* Grammar as evidence for conceptual structure // Hall, Bresnan and Miller (Eds). Linguistic theory and psychological reality. Cambridge, UK, MIT Press: 1978. P. 201–228.
5. *Путинцева А., Ковригина Л., Шилин И*. Автоматическая классификация эллиптических конструкций в русской спонтанной речи // CEUR Workshop Proceedings. 2018. V. 2233. P. 120–138.
URL: https://ceur-ws.org/Vol-2233/Paper_16.pdf
6. *Naidenova X.* Cognitive elements in forming and understanding sentences // CEUR Workshop Proceedings. 2021. V. 2910. P. 65–73.
URL: http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-2910/short6.pdf
7. *Найденова К.А, Булыкина Е.С., Пархоменко В.А., Щукин А.В., Мартирова Т.А.* Разработка компьютерной системы тестирования когнитивных способностей респондентов на основе предложений с эллипсисами // Электронные библиотеки. 2023. Т. 26. № 3. С. 340–364.
https://doi.org/10.26907/1562-5419-2023-26-3-340-364.

URL: https://rdl-journal.ru/article/view/784/856

8. *Naidenova X., Parkhomenko V., Bulykina E., Schukin A., Lizunov Y., Martirova T.* A System of Software Tools for Investigation of the Cognitive Function of Elliptical Sentence Generation // 2023 Ivannikov Ispras Open Conference (ISPRAS), Moscow, Russian Federation, 2023. P. 112–117.
https://doi.org/10.1109/ISPRAS60948.2023.10508180

EXPERIMENTAL STUDY OF COGNITIVE FUNCTION OF GENERATING ELLIPTICAL SENTENCES IN PLANIMETRIC TASKS

Vladimir Parkhomenko[1] [0000-0001-7757-377X], Xenia Naidenova[2] [0000-0003-2377-7093], Tat'yana Martirova[3] [0000-0003-0000-6608], Alexander Schukin[4] [0000-0002-9534-824X]

[1,4] *Peter the Great Saint Petersburg Polytechnic University, Saint Petersburg*,
[2,3] *Military medical academy, Saint Petersburg*

[1]vladimir.parkhomenko@spbstu.ru, [2]ksennaidd@gmail.com,

[3]martta462@yandex.ru, [4]alexander.schukin@spbstu.ru

Abstract

The paper is devoted to the study of the cognitive function associated with the generation of elliptical sentences in the Russian language. The study is conducted by testing this cognitive ability using a computer system specially developed by the authors for this purpose. Testing of this cognitive ability is proposed and implemented for the first time. The system is an extension of Moodle and is openly hosted in the github repository. Elliptical constructions are limited to verbal and nominal ellipses, which are theoretically possible to be completely reconstructed based on the context of the sentence. The study is conducted with the participation of SPbPU students as respondents. The texts of planimetric tasks are chosen as the subject area. As a result of the analysis of testing data, the following results are obtained: the influence of the respondent's knowledge of the subject area (planimetry) on the test results is established; a ten-

dency towards self-study of respondents was discovered, which is manifested in a reduction in time and an increase in scores as they pass tests; it is shown that respondents are poorly motivated if they do not see feedback on the answer to the completed task. The paper discusses the problems of further development of the testing system and its use in adapting questionnaires (tasks) to assess the knowledge of SPbPU students in the field of automation of bug detection in programs, as well as for diagnosing the functional state of operator specialists and express diagnosis of dementia. It also seems promising to use the system to improve the processes of syntactic parsing of elliptic sentences and automate the restoration of ellipses in the subject area of planimetry.

Keywords*: online testing system, development, experiments, cognitive function, ellipsis, planimetry*

REFERENCES

1. *Paducheva E.V.* O semantike sintaksisa. M.: URSS, 2019. 296 s.

2. *Jackendoff R.* Semantic and cognition. Cambridge: MIT Press, 1983. 283 p.

3. *Jackendoff R.* Information is in the mind of the beholder // Linguistic and Philosoph. 1985. V. 8. No. 1. P. 23–33.

4. *Jackendoff R.* Grammar as evidence for conceptual structure // Hall, Bresnan and Miller (Eds.). Linguistic theory and psychological reality. Cambridge, UK, MIT Press, 1978. P. 201–228.

5. *Putintseva A., Kovrigina L., Shilin I.* Automatic Classification of Elliptical Constructions in Russian Spontaneous Speech // CEUR Workshop Proceedings. 2018. V. 2233. P. 120–138. URL: https://ceur-ws.org/Vol-2233/Paper_16.pdf

6. *Naidenova X.* Cognitive elements in forming and understanding sentences // CEUR Workshop Proceedings. 2021. V. 2910. P. 65–73.
URL: http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-2910/short6.pdf

7. *Naidenova X.A., Bulykina E., Parkhomenko V.A, Schukin A.V., Martirova T.A.* Developing computer system for testing cognitive respondent's abilities based on elliptical sentences // Russian Digital Libraries Journal. 2023. V. 26. No. 3. P. 340–364. https://doi.org/10.26907/1562-5419-2023-26-3-340-364.
URL: https://rdl-journal.ru/article/view/784/856

8. *Naidenova X., Parkhomenko V., Bulykina E., Schukin A., Lizunov Y., Martirova T*. A System of Software Tools for Investigation of the Cognitive Function of Elliptical Sentence Generation. 2023 Ivannikov Ispras Open Conference (ISPRAS), Moscow, Russian Federation, 2023. P. 112–117. https://doi.org/10.1109/ISPRAS60948.2023.10508180

СВЕДЕНИЯ ОБ АВТОРАХ

ПАРХОМЕНКО Владимир Андреевич – старший преподаватель, Санкт-Петербургский политехнический университет Петра Великого, Санкт Петербург

Vladimir Andreevich PARKHOMENKO – Senior Lecturer, Peter the Great Saint Peterburg Polytechnic University, Saint Petersburg

email: vladimir.parkhomenko@spbstu.ru

ORCID: 0000-0001-7757-377X

НАЙДЕНОВА Ксения Александровна – к.т.н., старший научный сотрудник, Военно-медицинская академия, Санкт Петербург.

Xenia Aleksandrovna NAIDENOVA –PhD, Senior Researcher, Military medical academy, Saint Petersburg,

email: ksennaidd@gmail.com

ORCID: 00000-0003-2377-7093

МАРТИРОВА Татьяна Александровна – младший научный сотрудник, Военно-медицинская академия, Санкт Петербург

Tan'yana Aleksandrovna MARTIROVA – Researcher, Military medical academy, Saint Petersburg,

email: martta462@yandex.ru

ORCID: 0000-0003-0000-6608

ЩУКИН Александр Валентинович – к. т. н., доцент, Санкт-Петербургский политехнический университет Петра Великого, г. Санкт-Петербург.

Alexander Valentinovich SCHUKIN – PhD, associate professor, Peter the Great Saint-Petersburg Polytechnic University, Saint-Petersburg,

email: alexander.schukin@spbstu.ru

ORCID: 0000-0002-9534-824X

Материал поступил в редакцию 2 мая 2024 года

УДК 004.4

СИСТЕМА АВТОМАТИЗАЦИИ ЧИСЛЕННОЙ ОЦЕНКИ СХОДСТВА ANDROID-ПРИЛОЖЕНИЙ

В. В. Петров [0009-0004-4213-7328]

Институт информационных технологий и интеллектуальных систем, Казанский (Приволжский) федеральный университет, ул. Кремлёвская, 35, г. Казань, 420008

valeryvpetrov.itis@gmail.com

Аннотация

Работа посвящена проектированию и разработке системы автоматизации численной оценки сходства Android-приложений. Задача оценки сходства приложений сведена к оценке сходства множеств графов потока управления, построенных на основе кода из classes.dex файлов приложений. Значение сходства вычислено на основе матрицы сходства. Для сравнения графов потока управления использованы алгоритмы редактирования графов и расстояние Левенштейна. Сформулированы критерии сходства приложений и исследованы формы их представления. Представлены виды моделей Android-приложений и методы их построения. Разработан прототип системы автоматизации численной оценки сходства Android-приложений. С помощью инструментов параллельного программирования выполнена оптимизация программного решения. Проведены эксперименты и сделан вывод о способности разработанной системы выявлять сходства между Android-приложениями.

Ключевые слова*: сходство Android-приложений, сходство программ, матрица сходства, расстояние редактирования графов потока управления, визуализация матрицы сходства, граф потока управления.*

ВВЕДЕНИЕ

Сходство программ оценивается во многих задачах: обнаружения дублированного или клонированного кода, проверки авторских прав или патентов, повы-

шения утилизации кодовой базы и др. Для решения каждой из приведенных задач прежде всего необходимо определить, что означает «сходство программ», а также предложить способы его измерения.

Опытный программист способен оценить сходство программ по ключевым критериям: наименованию классов, функций, переменных, потока управления, использованию литералов и др. В таком случае часть критериев может быть определена и оценена субъективно и отличаться от сравнения к сравнению не только одним программистом, но и несколькими. В таких условиях возникает необходимость как в формализации критериев, по которым определяется сходство программ, так и в разработке методов вычисления оценки их сходства.

Для оценки сходства между множеством программ необходимо выполнить $\frac{n!}{2(n-2)!}$ попарных сравнений, где n – количество программ. Для задачи нахождения плагиата возможен поиск среди миллионов программ. При этом программы могут состоять из тысяч файлов, каждый из которых может иметь тысячи строк кода. Выполнение сравнения вручную может потребовать годы, в то время как размер программ и их количество растут каждый день. Все эти условия формируют потребность в автоматизации сравнения сходства программ.

Решение проблемы автоматизации численной оценки сходства программ имеет практическую значимость Эта проблема актуальна для индустрии приложений для ОС Android. Согласно исследованию [1], от 5 до 13% Android-приложений, распространяемых на шести Android-площадках, являются клонами. В исследовании [2] показано, что как минимум 141 приложение было клонировано. Также известны случаи плагиата мобильных приложений [3, 4].

Приложения для ОС Android распространяются в виде .apk и .aab артефактов [5]. Артефакты могут быть скачаны с различных площадок, таких как Google Play Store, Huawei App Gallery и др. При этом площадки могут быть не только официальными. Так как перед установкой приложения необходимо скачать на устройство артефакт, возникает ситуация, когда злоумышленник получает прямой доступ к артефакту и способен выполнить MATE-атаку [6]. После этого злоумышленник может опубликовать измененное приложение в интернете под видом исходного и украсть пользовательские данные, ухудшить имидж компании и др.

В рамках настоящей работы спроектирована и разработана система автоматизации численной оценки сходства Android-приложений. Для моделирования Android-приложения использована матрица графов потока управления функций приложения. Сходство графов потока управления оценена при помощи вычисления расстояния редактирования графов. Вычисления итогового значения сходства выполнено с использованием технологий параллельных вычислений. Разработаны инструменты для автоматической и ручной оценок сходства, визуализации матрицы сходства и графов потока управления.

Статья имеет следующую структуру. В разделе 1 представлен обзор существующих решений и проведён сравнительный анализ этих решений. В разделе 2 дана постановка задачи сравнения Android-приложений и введены необходимые обозначения. В разделе 3 представлен сравнительный анализ форм представления Android-приложений и дано обоснование выбора формы представления приложений, используемой в настоящей работе. В разделе 4 проведён сравнительный анализ видов моделей приложений и дано обоснование выбора конкретной модели. В разделе 5 представлены методы построения моделей приложений, проведено их сравнение с целью выбора подходящего метода. В разделе 6 предложен метод оценки сходства моделей приложений на основе расстояния редактирования графов приложений. В разделе 7 описан алгоритм численной оценки сходства Android-приложений. В разделе 8 представлены вспомогательные инструменты для выполнения экспертной оценки сходства приложений. В разделе 9 описаны эксперименты и приведен анализ результатов.

1. ОБЗОР СУЩЕСТВУЮЩИХ РЕШЕНИЙ

В последнее время появилось множество исследований, направленных на выявление клонированных или перепакованных приложений в среде Android (см., например, [7–11]). Эти исследования, как правило, завершаются простым решением «да/нет» о том, является ли приложение копией, но в них отсутствует детальное обоснование этого решения и областей сходства. Тем не менее, в сфере исследований, разработок и даже среди пользователей существует очевидная потребность в понимании различий между разными версиями приложений. Например, разработчикам маркетплейсов и их пользователям часто требуется определить изменения в последнем обновлении приложения, чтобы убедиться, что они

соответствуют рекламируемым «новым» функциям. Разработчики приложений могут анализировать изменения в собственных приложениях, чтобы оценить их влияние на рейтинг в маркетплейсе. Тем временем исследователи могут разрабатывать системы рекомендаций по изменениям, анализируя версии приложений, и методы обнаружения вредоносного содержимого в переупакованных вредоносных программах.

Задача обнаружения переупакованных приложений представляет собой серьезную проблему. В последнее время научное сообщество, учитывая постоянно растущее количество приложений, занимается созданием быстрых методов оценки сходства на основе ресурсов приложений или машинного обучения. Однако результаты применения этих методов все еще требуют тщательной проверки путем детального сравнения потенциальных пар переупакованных приложений.

К сожалению, ведущие методы обнаружения клонов или переупакованных приложений основаны на сложных внутренних эвристиках, которые трудно воспроизвести, а разработанные прототипы инструментов часто недоступны для развития исследований в этой области [10]. Многие исследования по обнаружению переупакованных приложений [8, 11] не предоставляют инструменты, которые могло бы повторно использовать исследовательское сообщество. Насколько известно, такие общедоступные инструменты, как Androguard [12], FSquaDRA [13] и SimiDroid [14], являются основными ресурсами для изучения сходства приложений. Androguard анализирует сходства на уровне кода, FSquaDRA – на основе ресурсов. Однако ни Androguard, ни FSquaDRA не описывают различия между приложениями, что ограничивает их возможности для глубокого анализа сходств. SimiDroid призван восполнить этот пробел в исследованиях, создав фреймворк с открытым исходным кодом, который легко масштабируется и способен объяснять сходства и различия приложений. SimiDroid оснащен тремя разработанными плагинами для сравнения сходства на уровне кода, Android-компонентов и ресурсов приложения. В статье [14] SimiDroid проведен сравнительный анализ с Androguard и FSquaDRA. Результаты анализа показали, что методы, основанные на сравнении кода, дают более точные оценки сходства, чем подходы, основанные на сравнении ресурсов. SimiDroid был протестирован на реальных приложениях в несколь-

ких тематических исследованиях, продемонстрировав свою способность объяснять сходства в различных контекстах. Однако SimiDroid не позволяет исследователям изменять формулу, используемую для расчета баллов сходства.

Отличительной особенностью настоящего исследования являются сравнение скомпилированных Android-приложений в формате .apk с использованием графа потока управления в качестве модели программы, а также применение матрицы сходства для сравнения моделей.

2. ПОСТАНОВКА ЗАДАЧИ

Даны два Android-приложения P_1 и P_2. Необходимо вычислить $sim(P_1, P_2)$ – степень сходства приложений P_1 и P_2. При этом:

- P_1 и P_2 могут иметь одну из форм представления: *исходный код, скомпилированный код, скомпилированный артефакт*;
- $sim(P_1, P_2) \in [0; 1]$;
- $sim(P_1, P_2) = 0$ означает, что программы не имеют никакого сходства;
- $sim(P_1, P_2) = 1$ означает, что программы абсолютно идентичны;
- $sim(P_1, P_2) = sim(P_2, P_1)$.

Для построения модели программы используется функция $m: P \to M$, которая выделяет характеристики программ и объединяет их в структуру данных, которая называется моделью программы, причем:

- Характеристики программы выделяются при помощи *статического* или *динамического анализа* кода программы.
- $m(P_1) = M_1, m(P_2) = M_2$ являются моделями программ P_1 и P_2 соответственно.
- Значение $m(P)$ может быть получено только из P.
- Программа P_2 называется *производной* от программы P_1, если над ней были выполнены преобразования, не нарушающие смысла программы P_1. К таким преобразованиям относятся: обфускация (запутывание кода программы), изменение графа потока управления, шифрование констант и др.
- Если программа P_2 является производной от P_1, то $M_1 = M_2$.

3. ВЫБОР ФОРМЫ ПРЕДСТАВЛЕНИЯ ANDROID-ПРИЛОЖЕНИЯ

Android-приложения могут быть представлены в трех формах:

- Исходный код – в репозитории с приложением или на компьютере разработчика.
- Скомпилированный код – при промежуточной компиляции исходного кода в байт-код JVM [15] или байт-код ART [16].
- Скомпилированный артефакт – при компиляции приложения. В результате компиляции создаются файлы в форматах .apk или .aab, которые могут быть установлены на смартфон.

С точки зрения преобразования формы представления, исходный код содержит в себе наибольшее количество авторской информации, интеллектуальной собственности, так как исходный код создается человеком, его разработчиком. Скомпилированный код и артефакт получаются при помощи стандартных инструментов: компиляторов и др., которые преобразуют форму представления по известным алгоритмам. При этом компилятор способен удалить часть информации из исходного кода, например, комментарии, которые будет невозможно восстановить при декомпиляции кода. Поэтому количество полезной информации в исходном коде больше, чем в скомпилированном артефакте. С другой стороны, во время работы приложения выполняются скомпилированный, оптимизированный или обфусцированный коды, а не исходный.

С точки зрения доступности формы представления, скомпилированный артефакт имеет наибольшую доступность, так как может быть скачан из официального магазина приложений; любого интернет-ресурса; устройства, на котором установлено приложение, в отличие от исходного кода, доступ к которому, как правило, ограничен владельцем репозитория или компьютера.

В рамках данной работы в качестве сравниваемых программ выступают Android-приложения в форме скомпилированного артефакта – .apk файла. Выбор такой формы представления обоснован ее наибольшей доступностью и фактическим использованием при работе программы.

Архив .apk содержит следующие файлы (см., например, [17]):

- скомпилированный Dalvik / ART байт-кода – classes.dex. Таких файлов может быть несколько;
- манифеста приложения – AndroidManifest.xml;
- метаинформации – META-INF;
- ресурсов приложения – assets, res;

- скомпилированных нативных библиотек – lib;
- скомпилированных ресурсов – resources.arsc.

Каждый из этих файлов содержит информацию о работе приложения и может быть использован при оценке сходства; resources.arsc, META-INF, AndroidManifest.xml содержат типовую информацию о приложении и не несут особой ценности при копировании. Более того, содержимое этих файлов кратно меньше, чем у classes.dex, assets, res, lib. Директории assets и res содержат макеты приложения, файлы анимации, статические ресурсы и др. При реверс-инжиниринге этот код может быть использования для плагиата пользовательского интерфейса приложения; classes.dex и lib содержат скомпилированный код, наибольшее количество полезной информации о работе приложения и занимают основную часть .apk файла. В lib находится скомпилированный C/C++ код. Использование нативного кода при разработке Android-приложений является узконаправленным решением для специфичных задач – отрисовки сложной графики, ресурсоемких вычислений и др. – и на практике встречается редко.

В рамках данной работы предложено рассматривать файлы classes.dex в качестве источника информации об Android-приложении, так как они содержат скомпилированный исходный код и занимают значительную часть .apk файла.

4. ВЫБОР МОДЕЛИ ANDROID-ПРИЛОЖЕНИЯ

Исходно classes.dex представляет собой бинарный код для среды выполнения Android: Dalvik или ART. Для построения моделей на основе бинарного кода, инструкций, блоков инструкций, функций достаточно декомпилировать файл .apk и выполнить поверхностный анализ содержимого. Такой подход позволяет добиться высокой скорости построения в сравнении с другими моделями. Однако ни одна из них не описывает структуру приложения и является чувствительной к обфускации.

Следующие модели, помимо декомпиляции .apk, требуют дополнительных преобразований для построения, что снижает общее время работы алгоритма. Абстрактное синтаксическое дерево (abstract syntax tree, AST) описывает структуру программы и выполняемые инструкции, что является ее преимуществом (см., например, [19]). Основной недостаток AST заключается в том, что дерево строится на основе исходного кода программы, и в этом случае файл classes.dex

необходимо предварительно декомпилировать до .class файлов и потом снова декомпилировать до .java, .kt файлов. Дважды выполненная декомпиляция приводит к потерям информации, что критично для модели, использующей AST.

Модель на основе иерархии классов может быть построена на основе Android-приложения, так как при его разработке используются объектно-ориентированные языки. Модель описывает различные зависимости между классами: наследование, композицию и др. (см., например, [20]). Недостатком модели на основе иерархии классов является то, что она не описывает фактическую реализацию класса, то есть модель не в полной мере отражает содержимое программы.

Граф зависимостей программы [21] описывает поток управления функции и поток данных, что является преимуществом среди прочих моделей. Единственным недостатком является то, что граф не содержит информации о других вызываемых функциях.

Граф вызовов [22] также может быть использован в качестве модели Android-приложения: он описывает его структуру и не зависит от используемого языка программирования. Однако современные обфускаторы способы изменять поток управления, что влияет на чувствительность модели. Более того, граф вызовов не описывает содержимое функций.

Граф потока управления [23] в отличие от двух предыдущих моделей описывает выполняемые блоки инструкций. Также граф описывает структуру программы и имеет размерность меньше, чем у AST.

В рамках данной работы предложено в качестве модели Android-приложения использовать граф потока управления, построенный на основе скомпилированного кода файлов classes.dex. Несмотря на то, что граф потока управления также чувствителен к обфускации потока управления и требует дополнительных преобразований для построения, он является наиболее подходящим среди прочих при моделировании программ. Этот вывод подтверждается в систематическом обзоре литературы по теме сходства программ [18].

5. ВЫБОР МЕТОДА ПОСТРОЕНИЯ МОДЕЛИ

Для построения графа потока управления Android-приложения могут быть

использованы как статический, так и динамический анализ приложения. При статическом подходе анализ Android-приложения никогда не выполняется, а модель строится на основе кода из файлов classes.dex. При динамическом анализе Android-приложения он запускается на виртуальном или реальном устройстве, и исследуется его поведение во время выполнения.

Статический анализ приложения эффективен, потому что может исследовать набор всех возможных путей выполнения путем аппроксимации поведения программы. Это важно, потому что некоторые сценарии работы Android-приложений трудно вызвать динамически: выполнение задач по расписанию, работы приложения в фоне и др.

Основное преимущество динамического анализа заключается в том, что он отображает фактическое выполнение Android-приложения, а запутывания кода, применяемые к classes.dex, оказывают меньшее влияние на поведение программы.

В рамках настоящей работы предложено строить граф потока управления Android-приложения при помощи статического анализа. Выбор такого метода преимущественно обусловлен непоследовательной (асинхронной) природой выполнения Android-приложений. Взаимодействие с базовыми компонентами Android осуществляется по системе обратных вызовов: изменение интерфейса, обращение в сеть, работа в фоновом режиме и др., что значительно усложняет покрытие всех сценариев использовании приложения в случае использования динамического анализа.

6. ВЫБОР МЕТОДА ОЦЕНКИ СХОДСТВА МОДЕЛЕЙ

Метод сравнения на основе максимального общего подграфа используется для оценки сходства графов (см., например, [24]). Большим преимуществом этого метода является возможность контролировать размерность общего подграфа, что ускоряет его построение. Однако данный метод имеет один значительный недостаток: наличие нескольких общих подграфов. Эта проблема особенно актуальна при построении максимального общего подграфа для графов малого размера. Такой вывод подтвержден в исследовании [25].

Для оценки сходства графов потока управления в рамках настоящей работы

использован оптимизированный алгоритм вычисления расстояния редактирования [26]. Этот алгоритм требует меньше памяти и времени, чем вариации алгоритма A* (A star), и способен учитывать содержимое блоков инструкций графа потока управления.

В рамках настоящей работы расстояние редактирования графа вычисляется по формуле [26]

$$GED(g_1, g_2) = \min_{e_1,\dots,e_k \in \gamma(g_1,g_2)} \sum_{i=1}^{k} c(e_i), \quad (1)$$

где GED – расстояние между графами g_1 и g_2; $GED(g_1, g_2) \geq 0$;

e_i – одна из операций редактирования: вставка, удаление, замена;

$e_1, e_2, \dots, e_k \in \gamma(g_1, g_2)$ – путь редактирования длины k;

$\gamma(g_1, g_2)$ – множество путей редактирования g_1 в g_2;

$c(e_i)$ – функция стоимости операции редактирования, $c(e_i) = 1$.

Для приведения значения расстояния редактирования графа (1) в интервал [0; 1] предлагается использовать следующую формулу:

$$\widehat{GED}(g_1, g_2) = \frac{GED(g_1, g_2)}{\max(size(g_1), size(g_2))}, \quad (2)$$

где $\widehat{GED}(g_1, g_2)$ – расстояние редактирования между графами g_1, g_2;

$\widehat{GED}(g_1, g_2) \in [0; 1]$; $size(g)$ – размер графа g, $size(g) = |V| + |E|$.

Принятие решения о редактировании вершины графа потока управления выполняется по формуле

$$decision = sim(S_1, S_2) < threshold, \quad (3)$$

где $decision = \{\text{Ложь}, \text{Истина}\}$ — решение о редактировании;

$threshold$ – порог принятия решения, $threshold \in [0; 1]$;

S_1, S_2 – сравниваемые последовательности;

$M = len(S_1), N = len(S_2)$ – длины последовательностей;

$sim(S_1, S_2) = \max(M, N) - D(S_1, S_2)$.

$$D(S_1, S_2) = D(M, N),$$

где D – расстояние Левенштейна [27], $D \in [0; \max(M, N)]$.

Если значение *sim* меньше порогового, то вершина редактируется.

7. ОПИСАНИЕ АЛГОРИТМА

Алгоритм автоматизации численной оценки сходства Android-приложений можно разделить на основные этапы:

1. Подготовка к построению моделей сравниваемых приложений.
2. Построение моделей (графов потока управления) первого и второго Android-приложения.
3. Построение матрицы сходства на основе расстояния редактирования графов потока управления.
4. Нахождение пар наиболее схожих элементов.
5. Вычисление итогового значения сходства.

Для ускорения работы алгоритма построения матрицы сходства разработанная программа имеет механизмы параллелизации вычисления как на уровне строк матрицы, так и на уровне ее ячеек. Для этого создается конечное множество процессов, между которыми распределяется задача вычисления строк матрицы сходства. Внутри каждого процесса создается конечное множество потоков, каждый из которых вычисляет сходство между двумя графами потока управления.

Помимо параллелизации при вычислении расстояния редактирования используется параметр, устанавливающий верхнюю границу времени вычисления. При достижении этой границы в качестве результата алгоритм возвращает значение расстояния редактирования, актуальное на момент остановки.

8. ВСПОМОГАТЕЛЬНЫЕ ИНСТРУМЕНТЫ

В этом разделе приведено краткое описание программных инструментов,

разработанных для получения детальной информации о ходе выполнения и результатах оценки сходства. Функция calculate_apks_similarity.py в режиме оценки сходства Android-приложений выводит в консоль информацию («логи») о выполнении алгоритма: списки генерируемых графов потока управления, результаты вычисления сходства и др. Сгенерированные файлы содержат полезную для исследователя информацию, представленную в текстовом виде. Однако логи не визуализируют матрицу сходства и пары наиболее схожих графов потока управления. Для повышения скорости, простоты и качества экспертного анализа результатов работы calculate_apks_similarity.py была разработана дополнительная программа.

Интерфейс программы построен в виде окна, на котором изображается вычисленная calculate_apks_similarity.py матрица сходства. Отображаемая матрица сходства обрабатывает нажатия на ячейки. При нажатии на ячейку программа генерирует .pdf файла в котором находятся визуализированные графы потока управления. Граф на первой странице соответствует строке матрицы, на второй – столбцу. Сгенерированный .pdf файл имеет имя dots_<номер строки>_<номер столбца>_<значение сходства>.pdf (например, dots_1_23_1.0.pdf) и находится в папке visualize_m_comp. Папка visualize_m_comp находится в папке переданной аргументом --output_dir calculate_apks_similarity.py.

9. ЭКСПЕРИМЕНТЫ

На данный момент не существует стандартизированных критериев оценки сходства Android-приложений. Существует решение суда № А40-20593/2017 в котором присутствует формулировка «Схожесть наименований двух программ для ЭВМ и их основное назначение не могут служить основанием для вывода об их тождественности, так как для подобного вывода необходимы специальные знания (заключение эксперта)» [28]. Что касается научных исследований, то в ряде статей сравнивают собственные разработки с аналогами (например, в [29] k-gramm сравнивается с моделью предложенной Харуаки Тамада). В других выполняется оценка существующих программ без сравнения аналогов (например, в [30] код изменяется вручную и затем вычисляется сходство между исходной и измененной программой).

Наиболее простыми сценариями проверки сходства двух Android-приложений являются крайние значения интервала сходства: 0 – одно приложение пустое (нет кода) и любое Android-приложение (есть код); 1 – сравнение приложения с самим собой. Так как модель программы должна является её инвариантом (не изменяться при трансформациях с сохранением семантики), то в качестве ещё одного эксперимента можно использовать сравнение Android-приложения до обфускации и после (обфускация не должна влиять на значение сходства, либо влиять незначительно). Ожидается, что чем больше техник обфускации будет применено к Android-приложению, тем меньшим сходством оно будет обладать при сравнении его с исходным вариантом. Однако в систематическом обзоре литературы [17] подчеркивается, что на данный момент отсутствует модель, которая является устойчивой к любым трансформациям с сохранением семантики.

В рамках данной работы программное решения оценивается по следующим сценариям:

1. Android-приложение сравнивается с пустым .apk файлом.
2. Android-приложение сравнивается с самим собой.
3. Android-приложение сравнивается с вариантом этого приложения, в котором были изменены имена классов, пакетов при помощи ProGuard.
4. Android-приложение сравнивается с вариантом этого приложения, оптимизированным при помощи ProGuard вариант (опция -optimizationpasses 5).
5. Производится сравнение двух разных Android-приложений.

Сгенерированные файлы, информация о выполнении программ в рамках каждого эксперимента размещаются в GitHub репозитории работы в папке с соответствующим номером в папке «exp» (например, для первого эксперимента эта папка exp/1).

Описанные ниже эксперименты выполнялись на компьютере со следующей конфигурацией:

- наименование: MacBook Pro;
- операционная система: macOS Monterey (версия 12.4);
- процессор: Apple M1 Pro с 10 ядрами;
- объем оперативной памяти: 16 ГБ.

Краткое описание результатов экспериментов представлено в Таблице 1.

Значение колонки «Верно сопоставленные пары сходства» вычислено как отношение корректно найденных пар к общему числу выбранных пар сходств и найдено экспертно на основе анализа графов потока управления.

Таблица 1. Описание результатов экспериментов.

№	$sim(P_1, P_2$	Время вычисления, сек	Верно сопоставленные пары сходства	ins_block_sim_threshold	ged_timeout_sec, сек	processes_count	threads_count
1	0.0	2.3	0%	0.95	60	10	45
2	0.99	5.3	100%	0.95	60	10	45
3	0.9	3.6	100%	0.95	60	10	45
4	0.2	3.4	80%	0.95	60	10	45
5	0.19	64.4	23%	0.95	60	10	45

В первом эксперименте результат полностью соответствует ожиданиям. Так как второй .apk файл не содержал classes.dex, то построенная модель состояла из пустого множества графов потока управления. Поэтому dots_2.csv не содержит ни одной записи. Матрица сходства имеет размерность 17:0.

Во втором эксперименте матрица сходства симметрична по диагонали, что говорит о сравнении двух идентичных .apk файлов (см. рис. 1). Большинство ячеек матрицы имеют черный цвет и соответствуют конструкторам класса R (класс с идентификаторами ресурсов в Android).

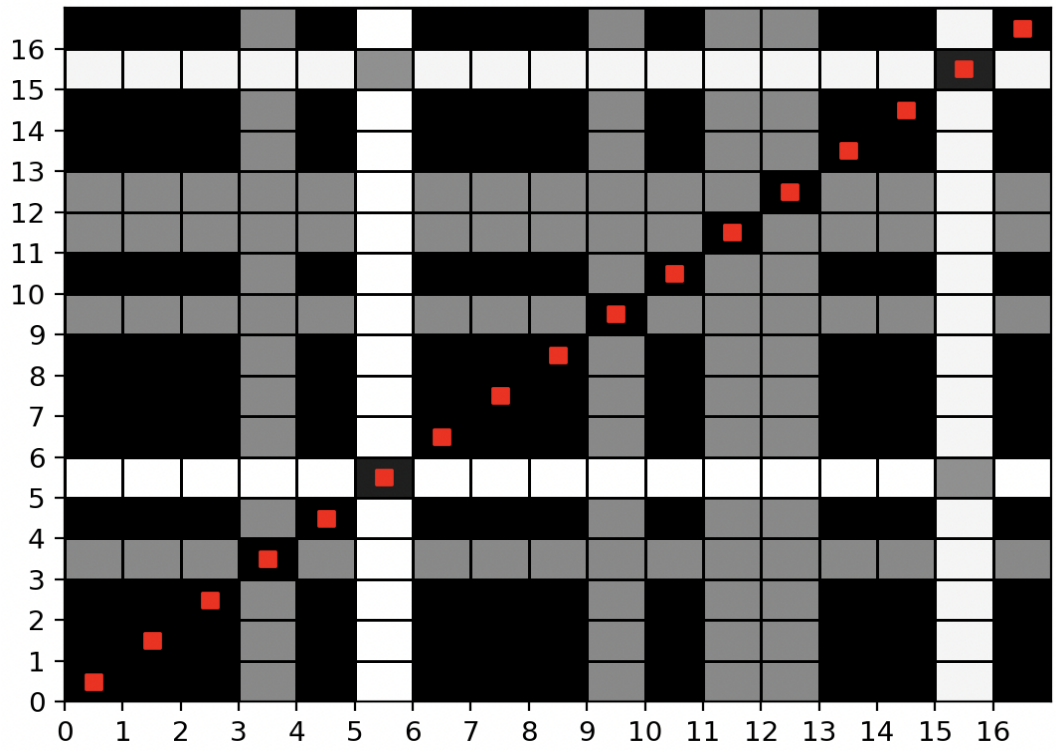


Рис. 1. Визуализация матрицы сходства из эксперимента 2

Примеры графов потока управления для ячеек 2:4, 10:16 представлены на рис. 2 слева и справа соответственно.

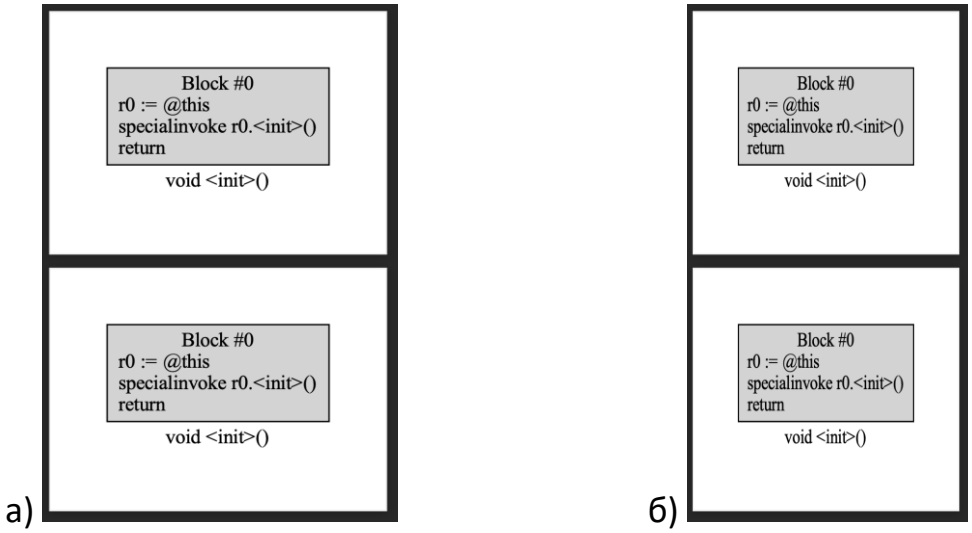


Рис. 2. Визуализация графов потока управления из эксперимента 2: R.layout void <init>().dot и R.xml void <init>().dot (а), R.style void <init>().dot и Unused void <init>().dot (б)

Строки и столбцы с серыми ячейками как правило имеют заметные структурные различия и соответствуют более сложным функциям программы. На рис. 3 представлен графы потока управления для ячейки 4:3, на рис. 4 для 15:5.

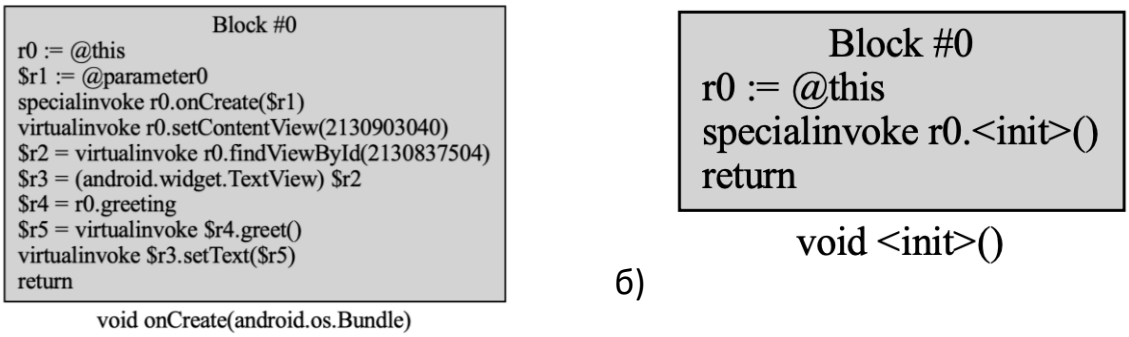


Рис. 3. Визуализация графов потока управления из эксперимента 2: MainActivity void onCreate(android.os.Bundle).dot (а) и R.xml void <init>().dot (б)

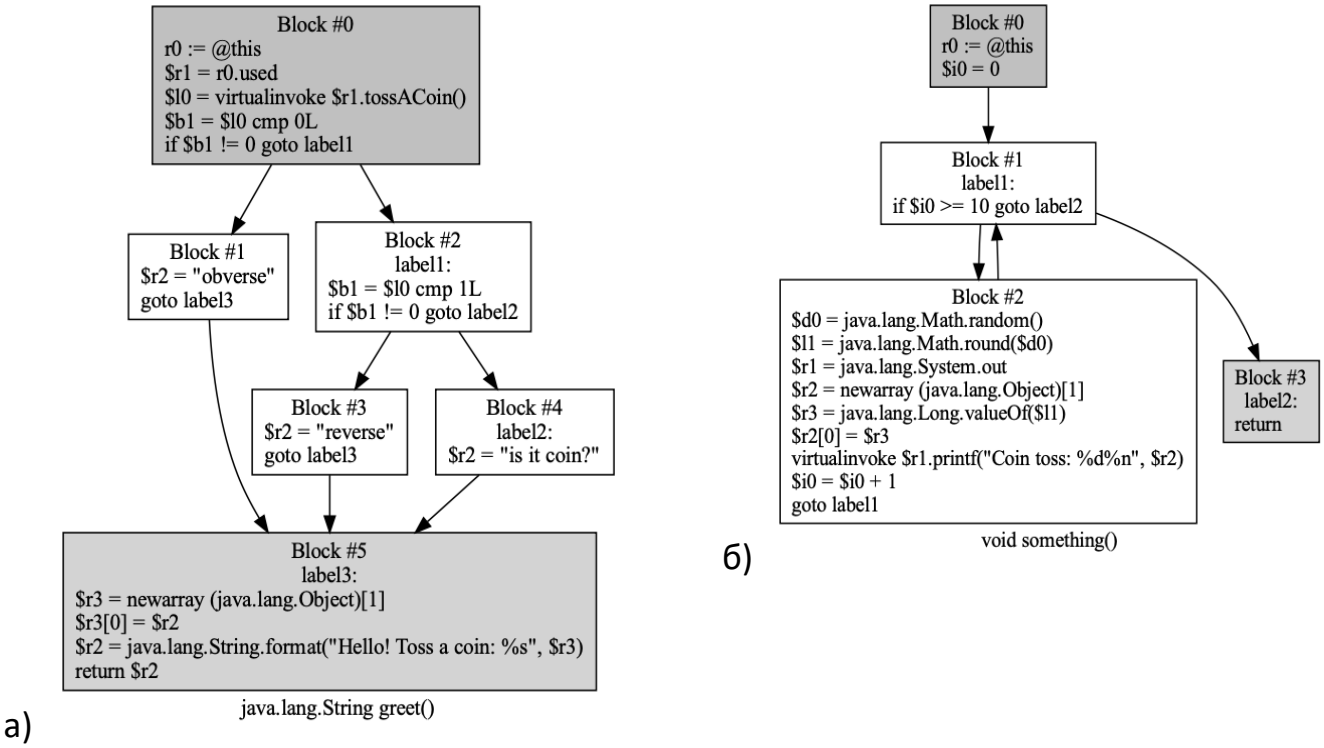


Рис. 4. Визуализация графов потока управления из эксперимента 2: Greeting java.lang.String greet().dot (а) и Unused void something().dot (б)

Значение сходства 0.99 приближено к 1.0. Программа сформировала корректный список пар соответствия графов потока управления. Однако

допустила ошибки при вычислении сходства для графов 5:5 (0.89), 15:15 (0.88).

В отличие от матрицы сходства из второго эксперимента, в данном случае пары графов не образуют диагональ (см. рис. 5).

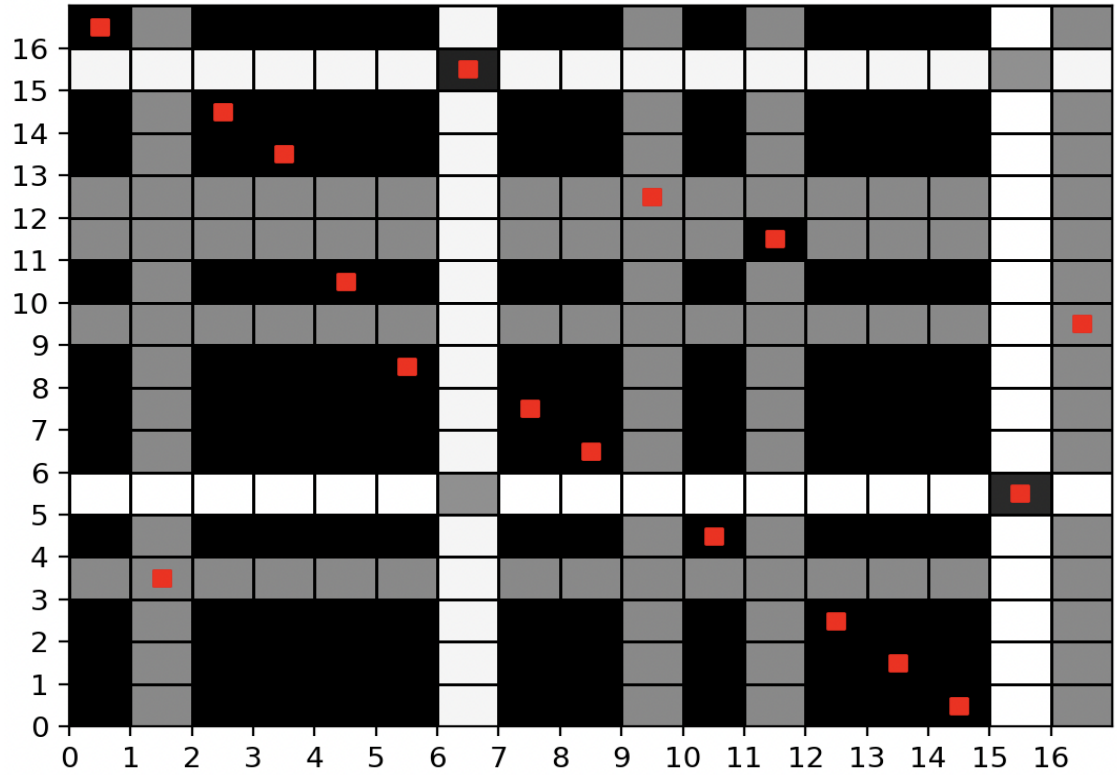


Рис. 5. Визуализация матрицы сходства из эксперимента 3

Это связано с тем, что имена методов при обфускации .apk изменяются, как и порядок их обхода при формировании списка графов. На рис. 6 видно, что структурно графы идентичны, но в блоках инструкции на третьей строке видно незначительное различие в наименовании классов «com.example.simpleapplication.something.Used» и «a.b».

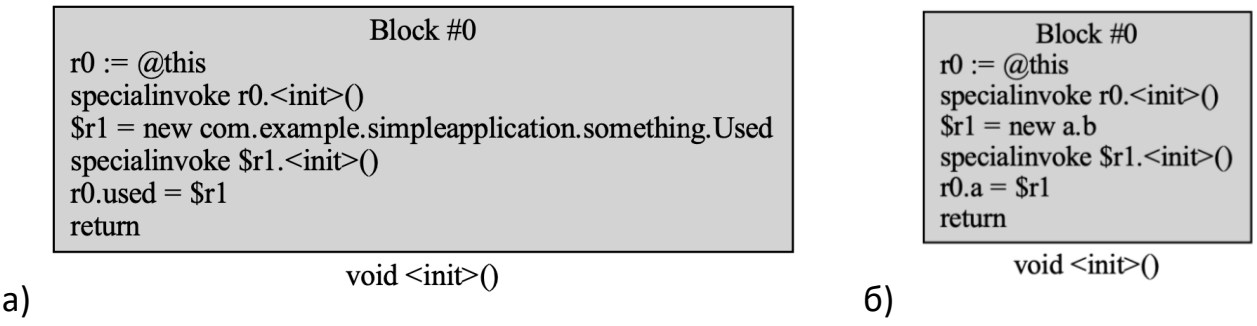


Рис. 6. Визуализация графов потока управления из эксперимента 3: Greeting void <init>().dot (а) и b void <init>().dot (б)

В строке 5 также видна разница в наименовании свойства класса «used» и «a». Именно эти различия в именах приводят к уменьшению значения сходства. Значение сходства 0.9 приближено к 1.0. Программа сформировала корректный список пар соответствия графов потока управления. Однако допустила ошибки при вычислении сходства блоков инструкций 3:1 (0.5), 5:15 (0.85) и др.

В четвертом эксперименте после выполнения оптимизации .apk файла количество классов сократилось с 17 до 5 (см. рис. 7).

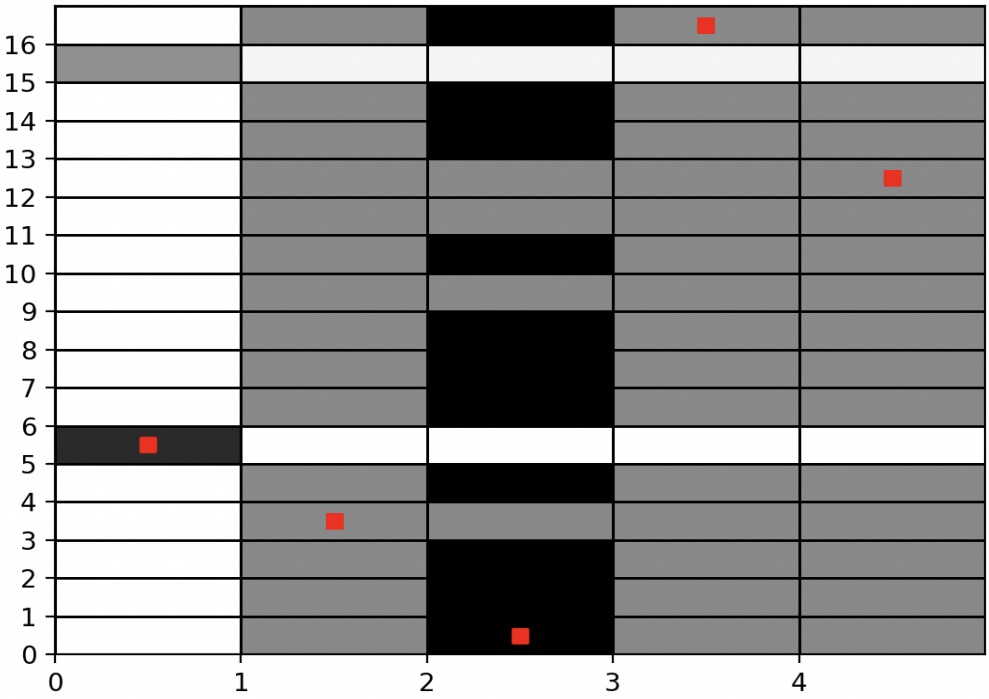


Рис. 7. Визуализация матрицы сходства из эксперимента 4

Помимо случаев, уже рассмотренных в третьем эксперименте, когда переименование снижает сходство, в этом примере есть один новый сценарий. На рис. 8 графы потока управления имеют структурное сходство.

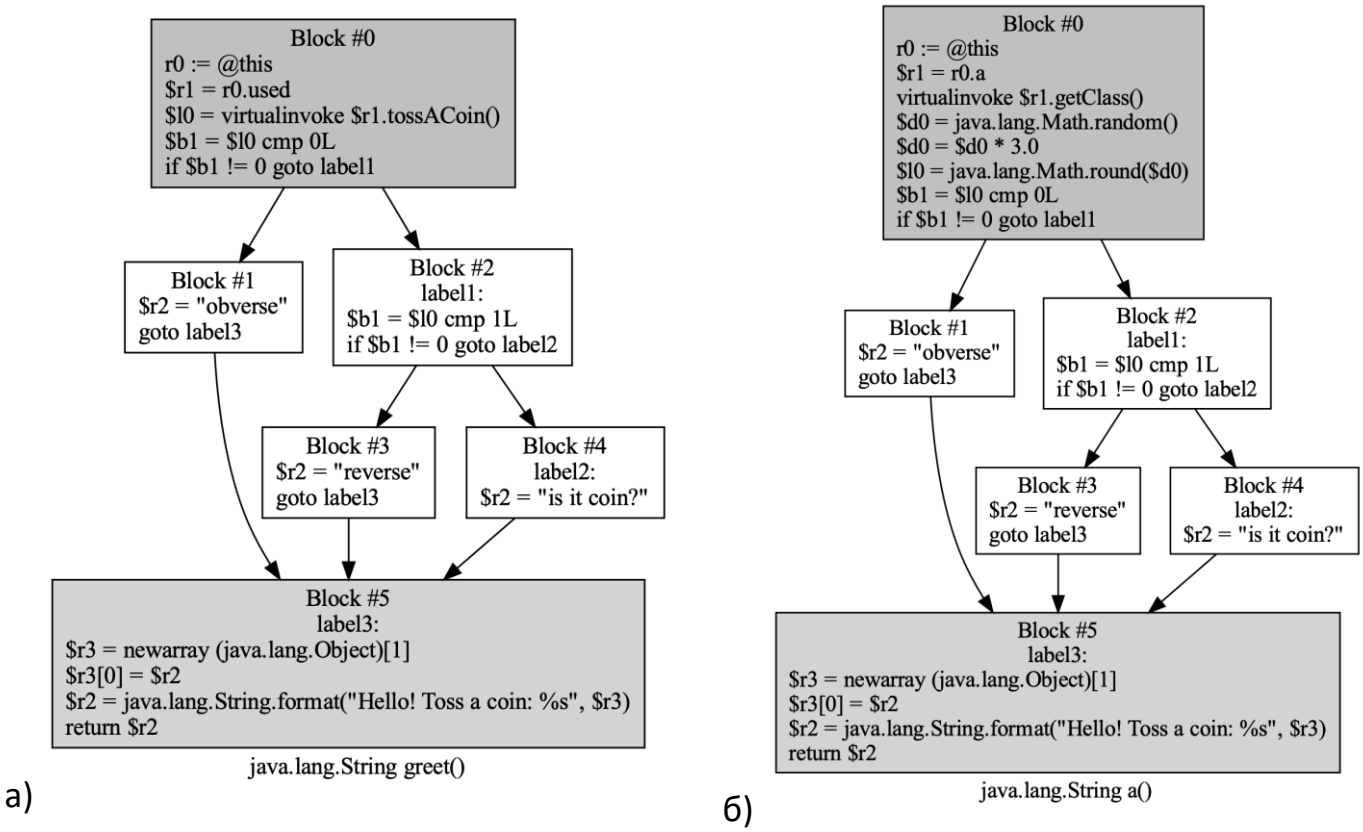


Рис. 8. Визуализация графов потока управления из эксперимента 4: Greeting java.lang.String greet().dot (а) и a java.lang.String a().dot (б)

Однако в «Block #0» графа б) был перенесен код из класса «com.example.simpleapplication.something.Used», при этом сам класс был удален из сборки. Значение сходства 0.2 показывает, что модификации графа потока управления оказывают сильное влияние на итоговое значение сходства. Несмотря на низкое значение сходства, программа верно сопоставила 4 из 5 графов потока управления. При использовании опции --ins_block_sim_threshold 0.5 сходство повышается до 0.29 (+0.09). Однако понижение значения порога эквивалентности блоков инструкций повышает количество схожих между собой графов потока управления, что может привести к формированию некорректных пар сходства. Данный вывод проиллюстрирован на рис. 9.

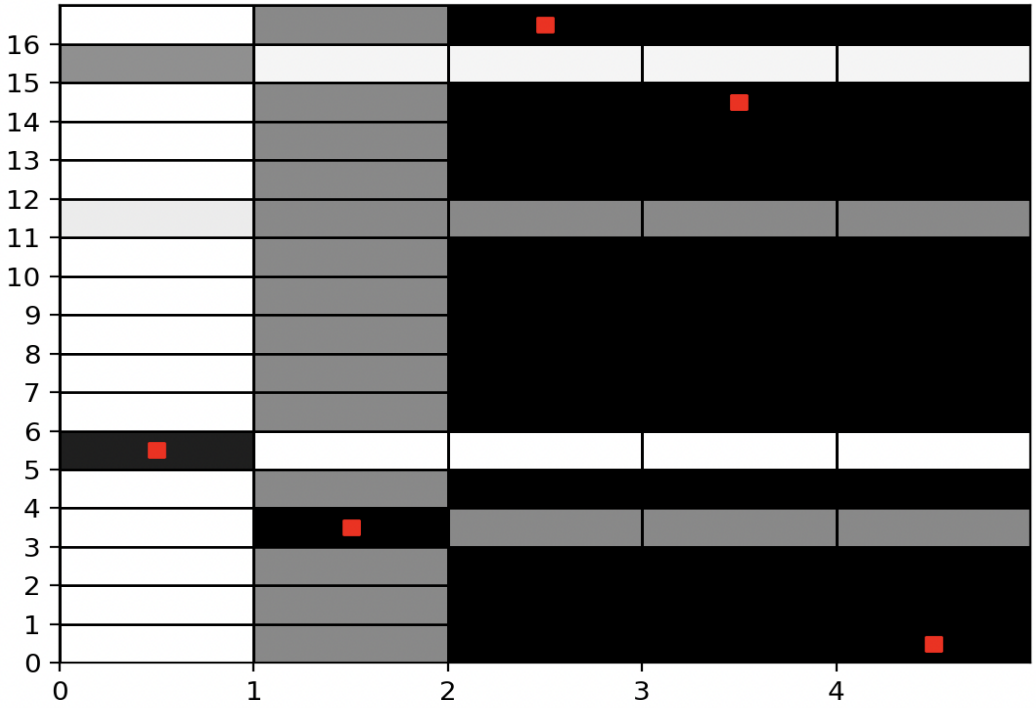


Рис. 9. Визуализация матрицы сходства из эксперимента 4

В пятом эксперименте значение сходства 0.19 показывает, что два принципиально разных Android-приложения имеют незначительное сходство (см. рис. 10). Пары сходства, сформированные программой, преимущественно не имеют схожей структуры. Пары со сходством, приближенным к 1, представляют собой графы потоков простых конструкторов классов и не несут полезной информации о сходстве всего приложения.

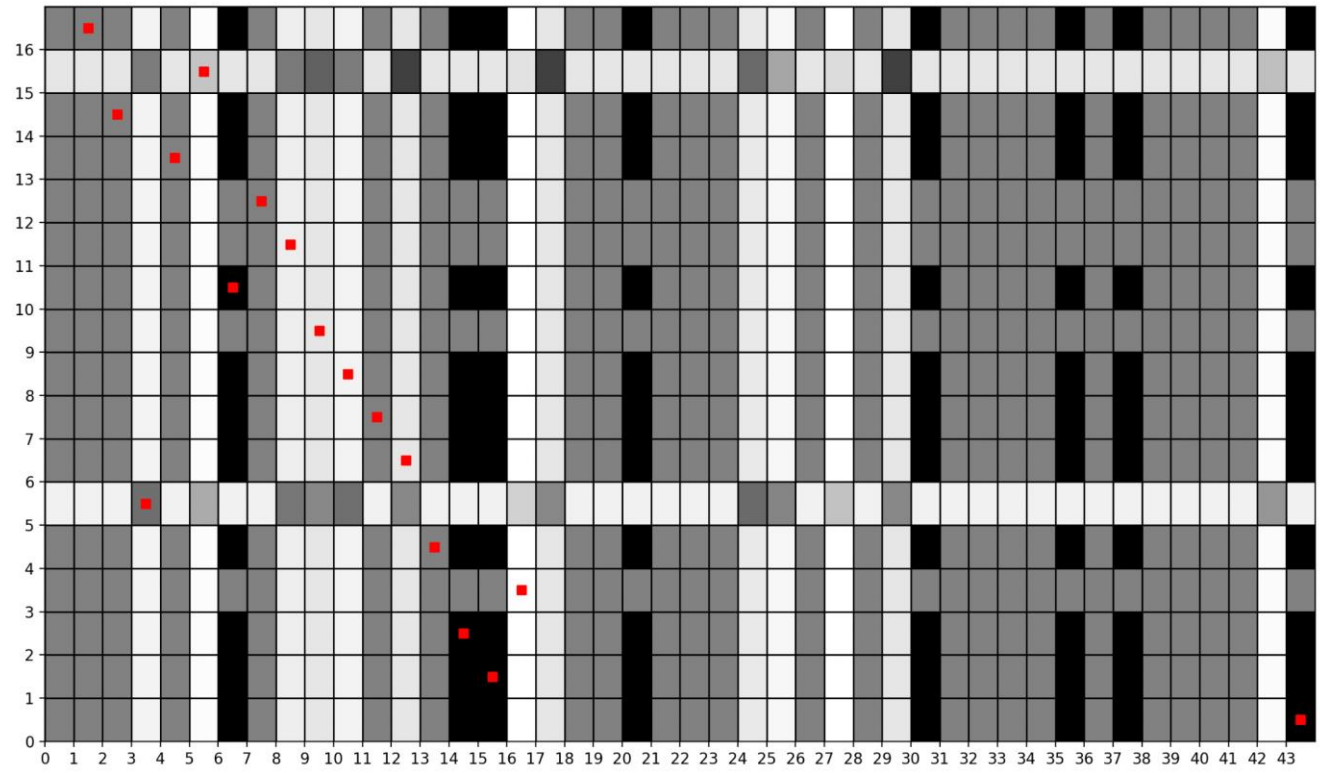


Рис. 10. Визуализация матрицы сходства из эксперимента 5

ЗАКЛЮЧЕНИЕ

Работа посвящена проектированию и разработке системы автоматизации численной оценки сходства Android-приложений. Были выполнены следующие задачи.

- Проведен анализ форм представления программ, видов моделей программ, методов построения моделей программ, видов сходства программ.
- Предложена модель представления Android-приложений, методы ее построения и алгоритм оценки сходства моделей.
- Спроектирован и разработан программный модуль для построения модели программы и численной оценки сходства.

Задача численной оценки сходства Android-приложений имеет ряд практических применений. Например, Google Play Store, RuStore, Huawei AppGallery и др. могут оценить сходство публикуемых приложений для поиска плагиатов. Функционал поиска плагинов позволит площадкам оповещать разработчиков-авторов о краже их интеллектуальной собственности, оповещать пользователей площадок

о подозрительности приложений с высокой степенью сходства, запрещать публиковать такие приложения и др. Также численная оценка сходства Android-приложений может быть использована для оценки работы обфускаторов (например, ProGuard, R8), при анализе кражи интеллектуальной собственности в судебных делах и прочих сценариях.

Разработанная система является прототипом решения задачи численной оценка сходства Android-приложений. Выполненные в работе эксперименты показали состоятельность разработанного решения. Среди преимуществ отметим: высокую скорость работы, репрезентативность модели с точки зрения структуры программы, инструменты для экспертного анализа. Среди недостатков укажем: чувствительность к обфускации. Для более точной оценки эффективности работы системы потребуются дополнительные эксперименты.

Дальнейшие развитие исследования предполагает повышение точности вычисления расстояния редактирования графов. Несмотря на то, что программа корректно формирует пары схожих графов потока управления, значения их сходства не равно 1.0. Также необходимо решить вопрос учёта оценки графов малой мощности.

Прототип разработанной системы прошел государственную регистрацию в виде программы для ЭВМ в Федеральной службе по интеллектуальной собственности (Роспатент) [31].

Ряд результатов, полученных в проведенном исследовании, представлен и обсуждён на конференции «Научный сервис в сети Интернет 2023» [32].

Текущая версия приложения размещена в открытом доступе в репозитории GitHub и доступна по ссылке: https://github.com/valeryvpetrov-dev/android-apps-similarity.

СПИСОК ЛИТЕРАТУРЫ

1. *Zhou W., Zhou Y., Jiang X., Ning P.* Detecting repackaged smartphone applications in third-party android marketplaces // Second ACM conference on Data and Application Security and Privacy. 2012. P. 317–326. https://doi.org/10.1145/2133601.2133640

2. *Crussell J., Gibler C., Chen H.* Attack of the clones: Detecting cloned applications on android markets // European Symposium on Research in Computer Security.

2012. P. 37–54. https://doi.org/10.1007/978-3-642-33167-1_3

3. Market Shocker! Iron Soldiers XDA Beta Published by Alleged Thief // Android Headline. URL: https://www.androidheadlines.com/2011/01/market-shocker-iron-soldiers-xda-beta-published-by-alleged-thief.html.

4. Fake Mobile Apps Steal Facebook Credentials, Cryptocurrency-Related Keys // TREND MICRO. URL: https://www.trendmicro.com/en_us/research/22/e/fake-mobile-apps-steal-facebook-credentials--crypto-related-keys.html.

5. Android App Bundle frequently asked questions // Android developers. URL: https://developer.android.com/guide/app-bundle/faq

6. *Akhunzada A., Sookhak M., Anuar N.B., Gani A., Ahmed E., Shiraz M., Furnell S., Hayat A., Khan M.K.* Man-At-The-End attacks: Analysis, taxonomy, human aspects, motivation and future directions // Journal of Network and Computer Applications. 2015. No. 48. P. 44–57. https://doi.org/10.1016/j.jnca.2014.10.009

7. *Chen J., Alalfi M.H., Dean T.R., Zou Y.* Detecting android malware using clone detection // Journal of Computer Science and Technology. 2015. No. 30. P. 942–956. https://doi.org/10.1007/s11390-015-1573-7

8. *Wang H., Guo Y., Ma Z., Chen X.* Wukong: A scalable and accurate two-phase approach to android app clone detection // Proceedings of the 2015 International Symposium on Software Testing and Analysis. 2015. P. 71–82. https://doi.org/10.1145/2771783.2771795

9. *Chen K., Liu P., Zhang Y.* Achieving accuracy and scalability simultaneously in detecting application clones on android markets // Proceedings of the 36th International Conference on Software Engineering. 2014. P. 175–186. https://doi.org/10.1145/2568225.2568286

10. *Li L., Bissyandé TF., Papadakis M., Rasthofer S., Bartel A., Octeau D., Klein J., Traon L*. Static analysis of android apps: A systematic literature review // Information and Software Technology. 2017. No. 88. P. 67–95. https://doi.org/10.1016/j.infsof.2017.04.001

11. *Guan Q., Huang H., Luo W., Zhu S*. Semantics-based repackaging detection for mobile apps // Engineering Secure Software and Systems: 8th International Symposium. 2016. No. 8. P. 89–105. https://doi.org/10.1007/978-3-319-30806-7_6

12. *Desnos A*. Android: Static analysis using similarity distance / Desnos A. // 2012 45th Hawaii international conference on system sciences. 2012. P. 5394–5403.

https://doi.org/10.1109/HICSS.2012.114

13. *Zhauniarovich Y., Gadyatskaya O., Crispo B., La Spina F., Moser E*. FSquaDRA: Fast detection of repackaged applications // Data and Applications Security and Privacy XXVIII: 28th Annual IFIP WG 11.3 Working Conference. 2014. No. 28. P. 130–145. https://doi.org/10.1007/978-3-662-43936-4_9

14. *Li L., Bissyandé TF., Klein J*. Simidroid: Identifying and explaining similarities in android apps // 2017 IEEE Trustcom/BigDataSE/ICESS. 2017. P. 136–143. https://doi.org/10.1007/s11390-019-1918-8

15. The Java® Virtual Machine Specification // Oracle.
URL: https://docs.oracle.com/javase/specs/jvms/se7/html/

16. Android Runtime (ART) and Dalvik // Android Open Source Project.
URL: https://source.android.com/docs/core/runtime/.

17. *Ratazzi E.P*. Understanding and improving security of the Android operating system // PhD dissertation; Syracuse University, 2016.
URL: https://surface.syr.edu/etd/592/

18. *Cesare S., Xiang Y*. Software similarity and classification – 1. Springer London, 2012. 88 p. https://doi.org/10.1007/978-1-4471-2909-7

19. *Jones J*. Abstract Syntax Tree Implementation Idioms // Proceedings of the 10th conference on pattern languages of programs (plop2003). 2003. P. 26.
URL: https://hillside.net/plop/plop2003/Papers/Jones-ImplementingASTs.pdf

20. *Heck A.J.P.* OOP: Class Hierarchy // Persoonlijke pagina's van FNWI-medewerkers Personal pages of Science staff.
URL: https://staff.fnwi.uva.nl/a.j.p.heck/Courses/JAVAcourse/ch3/s1.html

21. *Ferrante J., Ottenstein K.J., Warren J.D*. The program dependence graph and its use in optimization // ACM Transactions on Programming Languages and Systems (TOPLAS). 1987. No. 9 (3). P. 319–349. https://doi.org/10.1145/24039.24041

22. *Callahan D., Carle A., Hall M.W., Kennedy K*. Constructing the procedure call multigraph // IEEE Transactions on Software Engineering. 1990. No. 16(4). P. 483–487. https://doi.org/ 10.1109/32.54302

23. *Allen F.E*. Control flow analysis // ACM Sigplan Notices. 1970. No. 5(7). P. 1–19. https://doi.org/10.1145/800028.808479

24. *Kruegel C., Kirda E., Mutz D., Robertson W., Vigna G*. Polymorphic worm detection using structural information of executables // Recent Advances in Intrusion

Detection: 8th International Symposium. 2006. No. 8. P. 207–226. https://doi.org/10.1007/11663812

25. *Marcelli A., Quer S., Squillero G*. The maximum common subgraph problem: A portfolio approach // arXiv:1908.06418 preprint. 2019. URL: https://www.researchgate.net/publication/335258488_The_Maximum_Common_Subgraph_Problem_A_Portfolio_Approach

26. *Abu-Aisheh Z., Raveaux R., Ramel J.Y., Martineau P.* An exact graph edit distance algorithm for solving pattern recognition problems // 4th International Conference on Pattern Recognition Applications and Methods. 2015. No. 1. https://doi.org/10.5220/0005209202710278

27. *Левенштейн В.И.* Двоичные коды с исправлением выпадений, вставок и замещений символов // Доклады Академии наук СССР. 1965. № 163.4. С. 845–848.

28. Критерии сходства программ // ООО «АйТи-Лекс». URL: http://www.it-lex.ru/legal-cases/skhodstvo-programm/

29. *Myles G., Collberg C*. K-gram based software birthmarks // Proceedings of the 2005 ACM symposium on Applied computing. 2005. P. 314–318. https://doi.org/10.1145/1066677.1066753

30. *Liu C., Chen C., Han J., Yu P.S*. GPLAG: detection of software plagiarism by program dependence graph analysis // Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. 2006. . 872–881. https://doi.org/10.1145/1150402.1150522

31. Свидетельство о государственной регистрации программы для ЭВМ № 2023665834 Российская Федерация. Система автоматизации численной оценки сходства Android-приложений: № 2023664873: заявл. 16.07.2023: опубл. 20.07.2023 / В.В. Петров. – EDN NELKIS.

32. *Петров В.В.* Система автоматизации численной оценки сходства Android-приложений // Научный сервис в сети Интернет: труды XXV Всероссийской научной конференции (18–21 сентября 2023 г., онлайн). М.: ИПМ им. М.В. Келдыша, 2023. С. 283–297. https://doi.org/10.20948/abrau-2023-33

AUTOMATED SYSTEM FOR NUMERICAL SIMILARITY EVALUATION OF ANDROID APPLICATIONS

V. V. Petrov [0009-0004-4213-7328]

Institute of Information Technology and Intelligent Systems, Kazan (Volga region) Federal University, Kremlyovskaya ul., 35, Kazan, 420008

valeryvpetrov.itis@gmail.com

Abstract

This paper is devoted to the design and development of a system for automating numerical similarity assessment of Android applications. The task of application similarity evaluation is reduced to the similarity evaluation of sets of control flow graphs constructed based on code from classes.dex files of applications. The similarity value was calculated based on the similarity matrix. The algorithms of graph editing and Levenshtein distance were used to compare control flow graphs. Application similarity criteria were formulated and their representation forms were investigated. Types of Android application models and methods of their construction are presented. A prototype of the system for automating the numerical evaluation of Android-applications similarity is developed. Optimization of the software solution is performed with the help of parallel programming tools. Experiments are carried out and the conclusion is made about the ability of the developed system to detect similarities between Android applications.

Keywords*: Android application similarity, program similarity, similarity matrix, control flow graph edit distance, similarity matrix visualisation, control flow graph.*

REFERENCES

1. *Zhou W., Zhou Y., Jiang X., Ning P.* Detecting repackaged smartphone applications in third-party android marketplaces // Second ACM conference on Data and Application Security and Privacy. 2012. P. 317–326. https://doi.org/10.1145/2133601.2133640

2. *Crussell J., Gibler C., Chen H.* Attack of the clones: Detecting cloned applications on android markets // European Symposium on Research in Computer Security. 2012. P. 37–54. https://doi.org/10.1007/978-3-642-33167-1_3

3. Market Shocker! Iron Soldiers XDA Beta Published by Alleged Thief // Android Headline. URL: https://www.androidheadlines.com/2011/01/market-shocker-iron-soldiers-xda-beta-published-by-alleged-thief.html.

4. Fake Mobile Apps Steal Facebook Credentials, Cryptocurrency-Related Keys // TREND MICRO. URL: https://www.trendmicro.com/en_us/research/22/e/fake-mobile-apps-steal-facebook-credentials--crypto-related-keys.html.

5. Android App Bundle frequently asked questions // Android developers. URL: https://developer.android.com/guide/app-bundle/faq

6. *Akhunzada A., Sookhak M., Anuar N.B., Gani A., Ahmed E., Shiraz M., Furnell S., Hayat A., Khan M.K.* Man-At-The-End attacks: Analysis, taxonomy, human aspects, motivation and future directions // Journal of Network and Computer Applications. 2015. No. 48. P. 44–57. https://doi.org/10.1016/j.jnca.2014.10.009

7. *Chen J., Alalfi M.H., Dean T.R., Zou Y.* Detecting android malware using clone detection // Journal of Computer Science and Technology. 2015. No. 30. P. 942–956. https://doi.org/10.1007/s11390-015-1573-7

8. *Wang H., Guo Y., Ma Z., Chen X.* Wukong: A scalable and accurate two-phase approach to android app clone detection // Proceedings of the 2015 International Symposium on Software Testing and Analysis. 2015. P. 71–82. https://doi.org/10.1145/2771783.2771795

9. *Chen K., Liu P., Zhang Y.* Achieving accuracy and scalability simultaneously in detecting application clones on android markets // Proceedings of the 36th International Conference on Software Engineering. 2014. P. 175–186. https://doi.org/10.1145/2568225.2568286

10. *Li L., Bissyandé TF., Papadakis M., Rasthofer S., Bartel A., Octeau D., Klein J., Traon L*. Static analysis of android apps: A systematic literature review // Information and Software Technology. 2017. No. 88. P. 67–95. https://doi.org/10.1016/j.infsof.2017.04.001

11. *Guan Q., Huang H., Luo W., Zhu S*. Semantics-based repackaging detection for mobile apps // Engineering Secure Software and Systems: 8th International Symposium. 2016. No. 8. P. 89–105. https://doi.org/10.1007/978-3-319-30806-7_6

12. *Desnos A*. Android: Static analysis using similarity distance / Desnos A. // 2012 45th Hawaii international conference on system sciences. 2012. P. 5394–5403. https://doi.org/10.1109/HICSS.2012.114

13. *Zhauniarovich Y., Gadyatskaya O., Crispo B., La Spina F., Moser E*. FSquaDRA: Fast detection of repackaged applications // Data and Applications Security and Privacy XXVIII: 28th Annual IFIP WG 11.3 Working Conference. 2014. No. 28. P. 130–145. https://doi.org/10.1007/978-3-662-43936-4_9

14. *Li L., Bissyandé TF., Klein J*. Simidroid: Identifying and explaining similarities in android apps // 2017 IEEE Trustcom/BigDataSE/ICESS. 2017. P. 136–143. https://doi.org/10.1007/s11390-019-1918-8

15. The Java® Virtual Machine Specification // Oracle. URL: https://docs.oracle.com/javase/specs/jvms/se7/html/

16. Android Runtime (ART) and Dalvik // Android Open Source Project. URL: https://source.android.com/docs/core/runtime/.

17. *Ratazzi E.P*. Understanding and improving security of the Android operating system // PhD dissertation; Syracuse University, 2016. URL: https://surface.syr.edu/etd/592/

18. *Cesare S., Xiang Y*. Software similarity and classification – 1. Springer London, 2012. 88 p. https://doi.org/10.1007/978-1-4471-2909-7

19. *Jones J*. Abstract Syntax Tree Implementation Idioms // Proceedings of the 10th conference on pattern languages of programs (plop2003). 2003. P. 26. URL: https://hillside.net/plop/plop2003/Papers/Jones-ImplementingASTs.pdf

20. *Heck A.J.P.* OOP: Class Hierarchy // Persoonlijke pagina's van FNWI-medewerkers Personal pages of Science staff. URL: https://staff.fnwi.uva.nl/a.j.p.heck/Courses/JAVAcourse/ch3/s1.html

21. *Ferrante J., Ottenstein K.J., Warren J.D*. The program dependence graph and its use in optimization // ACM Transactions on Programming Languages and Systems (TOPLAS). 1987. No. 9 (3). P. 319–349. https://doi.org/10.1145/24039.24041

22. *Callahan D., Carle A., Hall M.W., Kennedy K*. Constructing the procedure call multigraph // IEEE Transactions on Software Engineering. 1990. No. 16(4). P. 483–487. https://doi.org/ 10.1109/32.54302

23. *Allen F.E*. Control flow analysis // ACM Sigplan Notices. 1970. No. 5(7). P. 1–19. https://doi.org/10.1145/800028.808479

24. *Kruegel C., Kirda E., Mutz D., Robertson W., Vigna G*. Polymorphic worm detection using structural information of executables // Recent Advances in Intrusion Detection: 8th International Symposium. 2006. No. 8. P. 207–226.

https://doi.org/10.1007/11663812

25. *Marcelli A., Quer S., Squillero G*. The maximum common subgraph problem: A portfolio approach // arXiv:1908.06418 preprint. 2019.
URL: https://www.researchgate.net/publication/335258488_The_Maximum_Common_Subgraph_Problem_A_Portfolio_Approach

26. *Abu-Aisheh Z., Raveaux R., Ramel J.Y., Martineau P.* An exact graph edit distance algorithm for solving pattern recognition problems // 4th International Conference on Pattern Recognition Applications and Methods. 2015. No. 1.
https://doi.org/10.5220/0005209202710278

27. *Levenshtein V.I*. Binary codes with correction of deletions, insertions and substitutions of symbols // Reports of the USSR Academies of Sciences. 1965. No. 163.4. P. 845–848. URL:
https://www.mathnet.ru/links/ebbbb75259f2fb388db92a54ec642b7d/dan31411.pdf

28. Criteria for similarity of programs // IT-Lex LLC.
URL: http://www.it-lex.ru/legal-cases/skhodstvo-programm/.

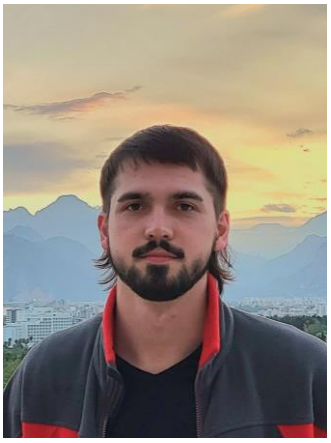
29. *Myles G., Collberg C*. K-gram based software birthmarks // Proceedings of the 2005 ACM symposium on Applied computing. 2005. P. 314–318.
https://doi.org/10.1145/1066677.1066753

30. *Liu C., Chen C., Han J., Yu P.S*. GPLAG: detection of software plagiarism by program dependence graph analysis // Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. 2006. P. 872–881.
https://doi.org/10.1145/1150402.1150522

31. Certificate of state registration of computer program No. 2023665834 Russian Federation. System of automation of numerical estimation of Android-applications similarity: No 2023664873: applied. 16.07.2023: published 20.07.2023 / V.V. Petrov. – EDN NELKIS.

32. *Petrov V.V*. Automation system for numerical assessment of the similarity of Android applications // Scientific service on the Internet: proceedings of the XXV All-Russian Scientific Conference (September 18–21, 2023, online). M.: IPM im. M.V. Keldysh, 2023. P. 283–297. https://doi.org/10.20948/abrau-2023-33

СВЕДЕНИЯ ОБ АВТОРЕ



ПЕТРОВ Валерий Владимирович – магистр программной инженерии, аспирант Института информационных технологий и интеллектуальных систем Казанского (Приволжского) федерального университета.

Valery PETROV – Magister of Software Engineering, Institute of Information Technology and Intelligent Systems, Kazan (Volga Region) Federal University.

Current scientific interests: Android application similarity, program similarity, static analysis, software modeling.

email: valeryvpetrov.itis@gmail.com;

ORCID: 0009-0004-4213-7328

Материал поступил в редакцию 24 апреля 2024 года

Переработанная версия – 16 мая 2024 года

УДК 004.4

КЛАССИФИКАЦИЯ ИЗОБРАЖЕНИЙ С ПОМОЩЬЮ СВЕРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ

С. А. Филиппов[0009-0007-9015-7982]

Институт информационных технологий и интеллектуальных систем Казанского федерального университета, ул. Кремлевская, 35, г. Казань, 420008

woppilif@icloud.com

Аннотация

Для классификации изображений в настоящее время можно применить множество различных инструментов, каждый из которых направлен на решение определенного спектра задач. В статье проведен краткий обзор библиотек и технологий для классификации изображений. Построена архитектура простой свёрточной нейронной сети для классификации изображений.

Были проведены эксперименты по распознаванию изображений с такими популярными нейронными сетями, как VGG16 и ResNet 50. Обе нейронные сети показали хорошие результаты. Однако ResNet 50 переобучилась из-за того, что в наборе данных присутствовали однотипные изображения для обучения, поскольку в данной нейронной сети больше слоев, позволяющих считывать признаки объектов на изображениях. С обученными моделями был проведен сравнительный анализ по распознаванию изображений, специально подготовленных для этого эксперимента.

***Ключевые слова**:* распознавание изображений, нейронная сеть, сверточная нейронная сеть, классификация изображений, машинное обучение

ВВЕДЕНИЕ

Машинное обучение — направление, которое в настоящее время актуально для решения различных задач. Такие задачи, как генерация изображений [1], генерация музыки [2], создание сцен для игр [3], озвучивание видеороликов в интернете [4], синхронный перевод [5] — лишь небольшой список того, каким потенциалом обладает машинное обучение.

Названная технология развивается большинством крупных компаний, разрабатывающих программное обеспечение. Такие компании, как показывает практика, могут как создавать свои собственные библиотеки для работы с машинным обучением, так и вносить вклад в разработки с открытым исходным кодом.

Так, например, компания Google разработала библиотеку под названием TensorFlow [6] — открытую библиотеку для машинного обучения, позволяющую решать большой спектр задач. На официальном сайте, посвященном этой библиотеке, представлены многочисленные примеры и варианты использования. В контексте этой статьи рассмотрен вопрос классификации изображений встроенными средствами этой библиотеки с использованием дополнительных функций библиотеки Keras [7].

Для классификации изображений применяют сверточные нейронные сети (Convolutional neural network, CNN) [8]. Во время обучения нейронной сети этого типа выделяют определенные признаки, на основе которых происходит обучение.

Библиотека TensorFlow позволяет спроектировать собственную нейронную сеть и установить специфические параметры обучения. В случае, когда в разработке собственной архитектуры сети нет необходимости, можно воспользоваться готовыми решениями. VGG16 [9] и ResNet 50 [10] — яркий пример спроектированных нейронных сетей, готовых для решения задач классификации изображений.

Далее описаны эксперименты по разработке собственной архитектуры нейронной сети, а также представлены результаты вычислительных экспериментов для сетей VGG16 и ResNet 50.

1. ПРОЕКТИРОВАНИЕ НЕЙРОННОЙ СЕТИ ДЛЯ РАСПОЗНАВАНИЯ ИЗОБРАЖЕНИЙ

Для проектирования нейронной сети с целью распознавания изображений с помощью CNN необходимо: подготовить наборы данных для обучения и тестирования, определить параметры размеров изображений, выстроить слои нейронной сети в определённом порядке, настроить эти слои и при этом для повышения точности предсказаний с помощью таких средств, как оптимизация гиперпараметров [11], увеличение данных [12], методы регуляризации [13], провести более

тонкую настройку эмпирическим путем, а также интерпретировать результаты обучения и сделать выводы.

В качестве набора данных для обучения нейронной сети использовался дата-сет Fruits 360 [14]. Из него было выбрано десять классов изображений: 'Apple Golden 1', 'Apricot', 'Beetroot', 'Cherry 1', 'Corn', 'Guava', 'Lemon', 'Orange', 'Tomato 1', 'Watermelon'. Для каждого класса в данном наборе присутствует несколько изображений, показывающих объект на 360 градусов. Так, в наборе присутствует 5038 изображений для обучения и 1681 изображение для тестирования. Некоторые из них представлены на рисунке 1.

Рис. 1. Пример данных для обучения из набора данных.

В качестве языка программирования для разработки нейронной сети был выбран Python. Разработка осуществлялась в интегрированной среде разработки PyCharm [15] на компьютере под управлением MacOS [16] на чипе M1 [17].

Процесс обучения занял три часа без применения дискретной графической карты, так как стандартные средства библиотеки TensorFlow осуществляют работу с CPU для вычислений, а не с GPU. Для большей производительности была установлена дополнительная библиотека TensorFlow MacOS [18] для процессоров M1,

что позволило получить доступ к GPU и увеличить производительность в несколько раз.

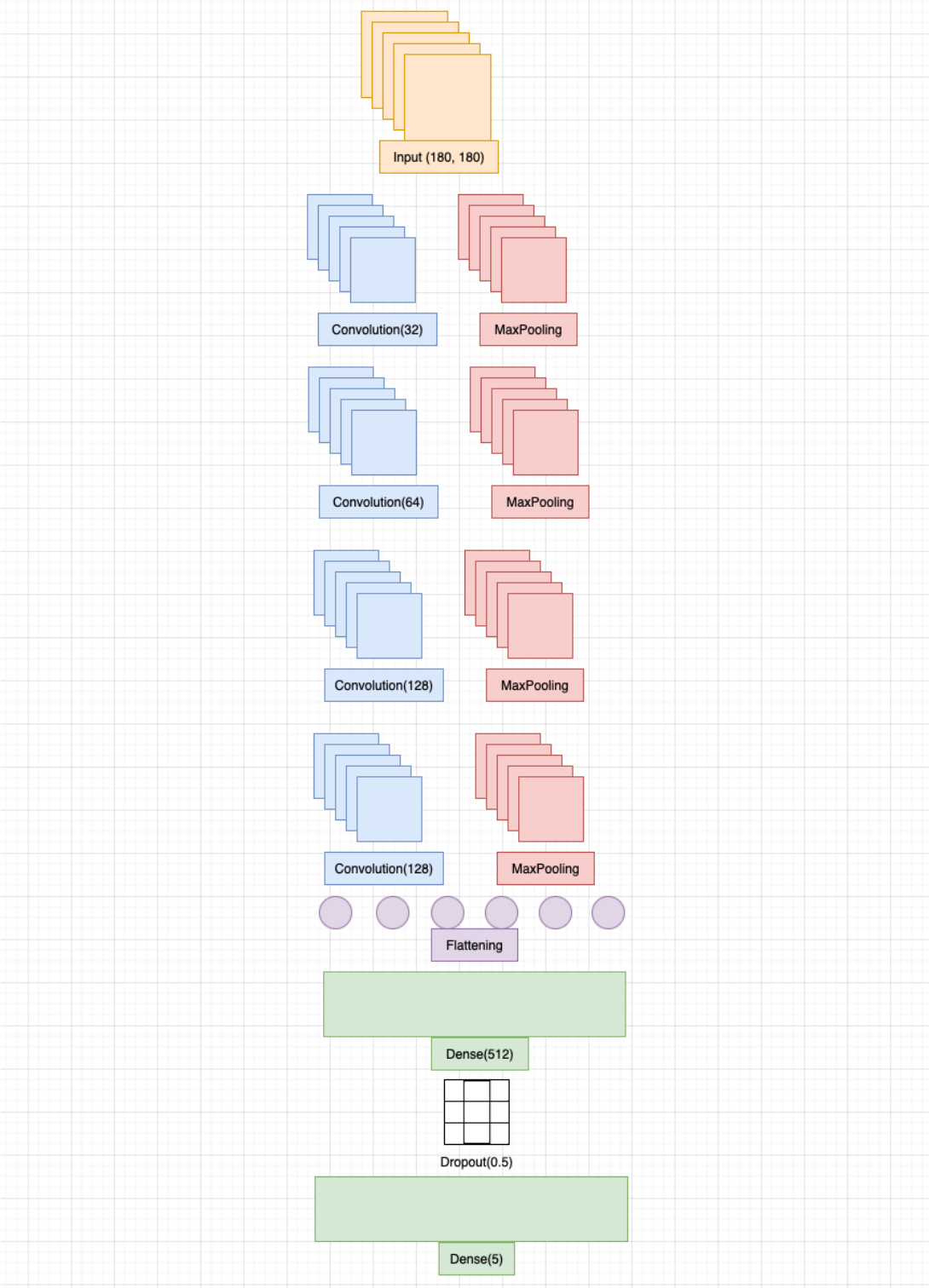


Рис. 2. Слои нейронной сети.

Далее были выстроены слои нейронной сети, которые можно увидеть на рисунке 2. Слои были выделены разными цветами, чтобы показать, как они упорядочены. Соответственно, нейронная сеть состоит из следующих слоев:

1. Слой Conv2D [19] с 32 фильтрами размером 3x3 и функцией активации ReLU, который обрабатывает входное изображение размером *m* x *n* x 3 (3 соответствует RGB-каналам изображения), где *m* — ширина изображения, а *n* — его высота. Этот слой выполняет первичное извлечение признаков из входного изображения.

2. Слой MaxPooling2D [20] с размером фильтра 2x2, который уменьшает размерность выходных данных в два раза. Это позволяет уменьшить количество параметров, которые необходимо обучить, и избежать переобучения модели.

3. Последовательность из двух слоев Conv2D и MaxPooling2D, аналогичных первым двум, но с большим количеством фильтров (64 и 128 соответственно). Эти слои выполняют более сложное извлечение признаков из входного изображения.

4. Еще один слой Conv2D с 128 фильтрами размером 3x3 и функцией активации ReLU, за которым следует слой MaxPooling2D с размером фильтра 2x2. Эти слои позволяют еще более точно извлечь признаки из изображения.

5. Слой Flatten [21], который преобразует выходные данные из последнего слоя в одномерный массив.

6. Полносвязный слой Dense [22] с 512 нейронами и функцией активации ReLU, который выполняет обработку извлеченных признаков и уменьшение их размерности до 512.

7. Слой Dropout [23], который помогает предотвратить переобучение модели, выбрасывая случайно выбранные нейроны во время обучения.

8. Полносвязный слой Dense с 5 нейронами и функцией активации softmax [24], который генерирует вероятности принадлежности изображения к каждому из 5 классов. Это позволяет модели выбрать наиболее подходящий класс для данного изображения.

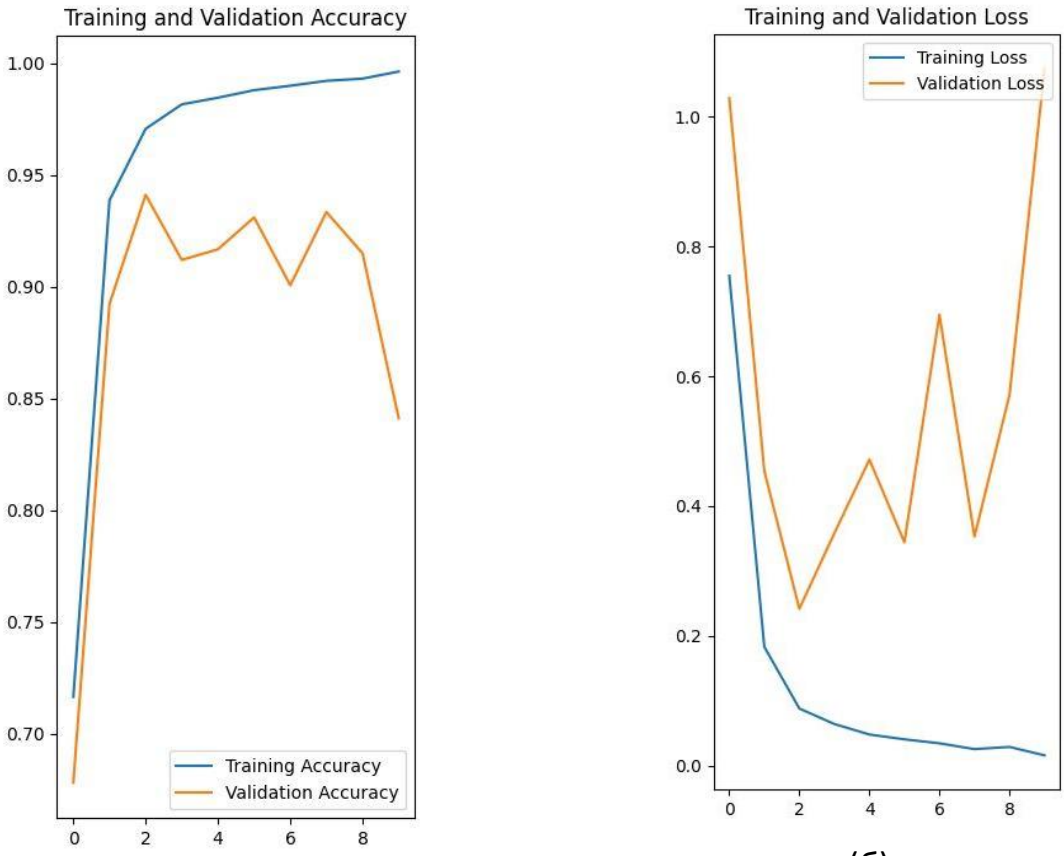
Далее было проведено обучение нейронной сети. Для ясности введем несколько терминов:

1. Точность — это показатель того, насколько хорошо работает модель классификации. Обычно она выражается в процентах и представляет собой количество правильных прогнозов из общего числа. Точность является бинарной, то есть она может быть истинной или ложной только для конкретной выборки. Это значение часто строится и отслеживается во время обучения, и иногда оно используется для оценки общей точности модели. Точность легче понять, чем потери, что является еще одним показателем производительности модели.

2. Функция потерь, также известная как функция затрат, используется для измерения точности прогнозов модели. Она учитывает вероятность или неопределенность прогноза в зависимости от того, насколько близок прогноз к истинному значению. Это позволяет нам получить более детальное представление о том, насколько хорошо работает модель.

2. РЕЗУЛЬТАТЫ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ С РАЗРАБОТАННОЙ НЕЙРОННОЙ СЕТЬЮ

Ниже приведены результаты обучения нейронной сети, которое состояло из 10 эпох. После обучения были получены значения точности обучения и функции потерь, которые отображены на рисунке 3.



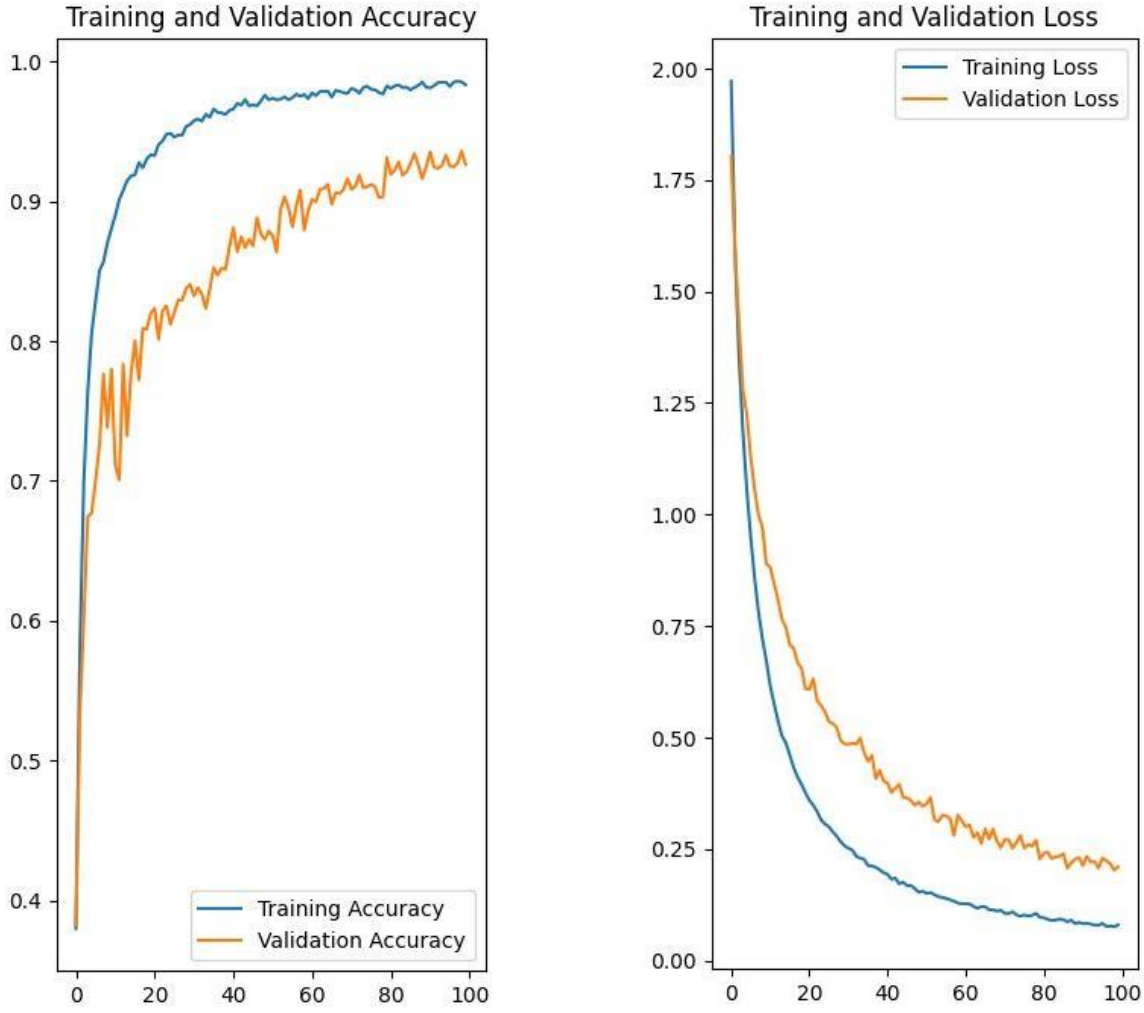
(а) (б)

Рис. 3. Графики точности и потерь для разработанной нейронной сети.

Из графика точности (рис. 3 а) видно, что нейронная сеть на первом этапе обучения показывает результаты, близкие к истинным значениям. Далее положительные результаты заметно снижаются, и на графике видно, как в процессе обучения нейронная сеть деградирует. На графике потерь (рис. 3 б) отражена аналогичная ситуация.

3. РЕЗУЛЬТАТЫ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ С НЕЙРОННОЙ СЕТЬЮ VGG16

VGG16 по сравнению с разработанной моделью имеет большее количество слоев для выявления признаков изображений, что положительно сказывается на показателях точности модели, как показано на рис. 4 а.

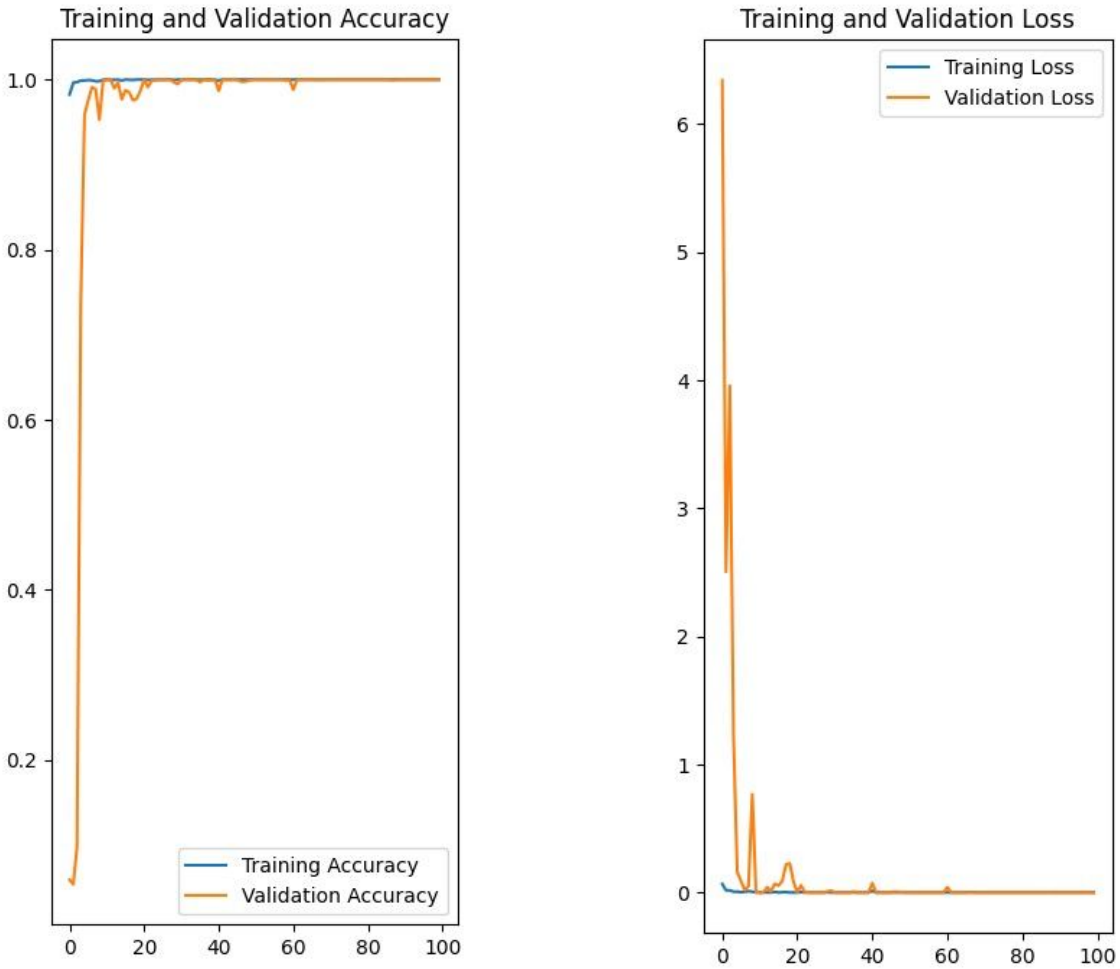


(а) (б)

Рис. 4. Графики точности и потерь для VGG16.

Постепенное обучение модели показано на первых этапах обучения – тогда, когда линии на графике точности пересекались, а после стали расходиться. Если бы графики все время пересекались, то это бы говорило о том, что обучаемая сеть переобучилась. Переобучение возникает тогда, когда обучение начинает происходить от различных шумов в данных или объем данных для обучения слишком мал. Далее, модель не может классифицировать тестовые данные верно, так как она обучена на слишком большом количестве ложных параметров.

4. РЕЗУЛЬТАТЫ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ С НЕЙРОННОЙ СЕТЬЮ RESNET 50



(а) (б)

Рис. 5. Графики точности и потерь для ResNet 50.

ResNet 50 – альтернатива для VGG 16, с большим количеством слоев. Слои в этой нейронной сети следуют правилу: количество фильтров на каждом уровне равно количеству выходных данных модели, что влияет на ее производительность. Признаки с изображений считываются в большем объеме, что влияет на показатели обучения сети в положительную сторону. Такую модель можно применять для классификации изображений с большим количеством объектов на них. В случае, если изображения содержат мало объектов для считывания признаков о них, можно получить переобученную модель уже на первых трех десятках эпох ее обучения, как это показано на рисунках 5(а) и 5(б).

5. РЕЗУЛЬТАТЫ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ

После обучения нейронных сетей был выбран набор данных для тестирования функции распознавания изображений. Результаты представлены на рисунках 6 и 7.

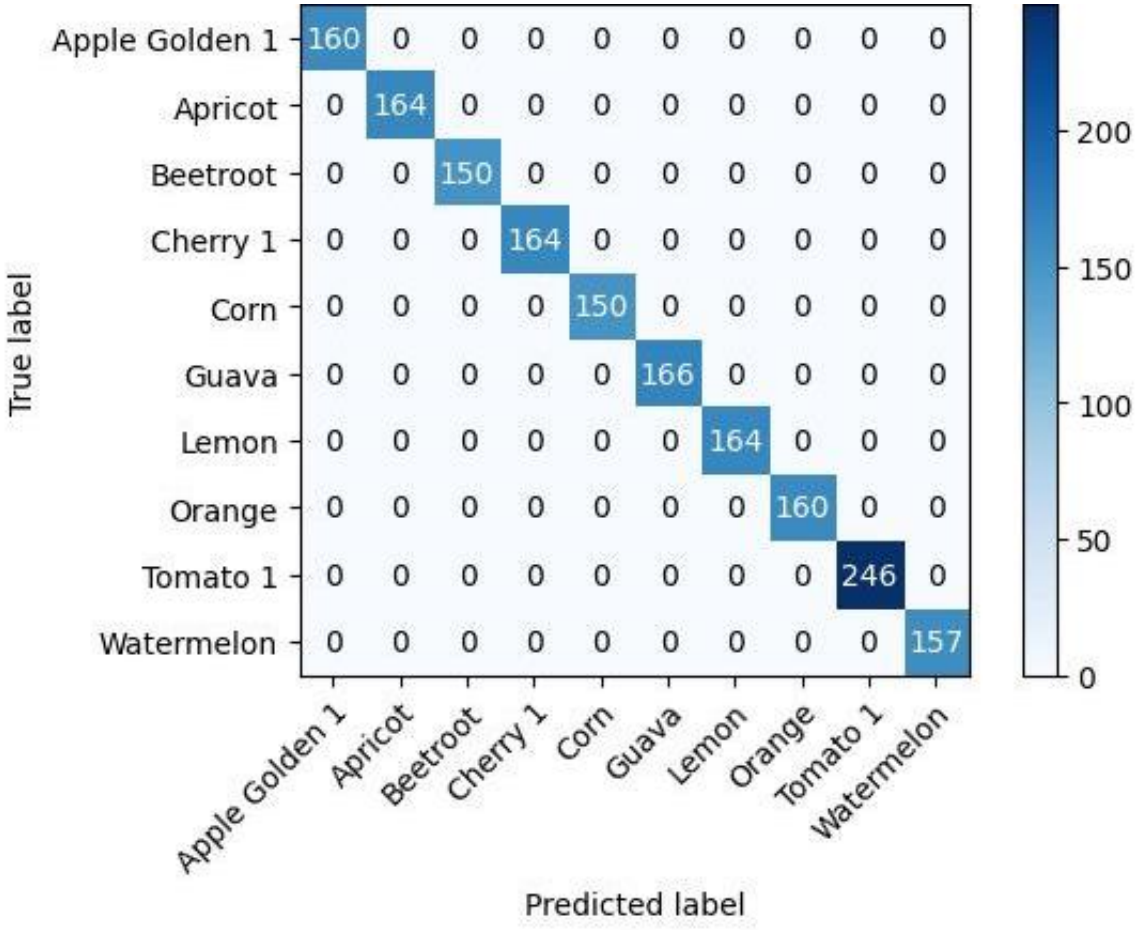


Рис. 6. Результаты эксперимента по классификации для модели ResNet 50.

Приведенная матрица на оси ординат показывает те значения, которые являются истинными, а на оси абсцисс – те, которые были предсказаны. Здесь наглядно видно результат переобучения. Нейронная сеть ни разу не ошиблась, что говорит о её переобучении.

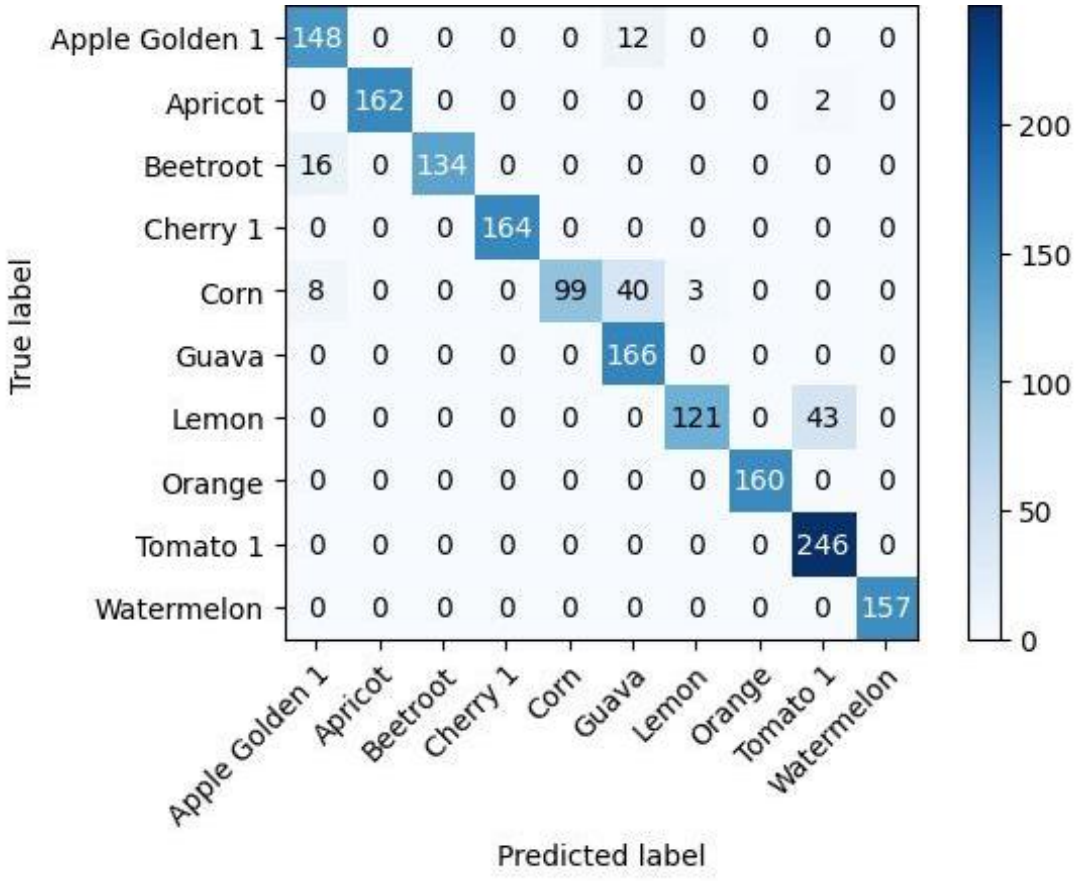


Рис. 7. Результаты эксперимента по классификации для модели VGG16.

ЗАКЛЮЧЕНИЕ

Машинное обучение позволяет решать задачи классификации изображений как с использованием самостоятельно разработанной модели нейронной сети, так и с использованием известных решений для этих задач. Вычислительные эксперименты показали, что с задачей лучше всего справилась модель VGG 16 – модель не переобучилась и показала результаты лучше своего аналога ResNet 50. В подготовленном наборе данных присутствовали изображения, в которых нет шумов и объектов, не относящихся к данным для обучения. Так, было определено, что для решения задач классификации изображений необязательно использовать такие большие модели, как ResNet 50, и был сделан вывод, что для эффективного обучения и получения результатов необходимо иметь хорошо подготовленный набор данных для решения специфичной задачи машинного обучения.

Благодарности

Выражаю благодарность доценту Института информационных технологий и интеллектуальных систем Казанского федерального университета М.О. Таланову за консультации по данной теме.

СПИСОК ЛИТЕРАТУРЫ

1. *Фисько Д.В.* Обзор методов условной генерации изображений нейросетевыми моделями // Актуальные вопросы современной науки и технологий. 2021. С. 57–62.

2. *Мосин Е.Д., Белов Ю.С.* Генерация музыки с использованием двунаправленной рекуррентной нейронной сети // Научное обозрение. Технические науки. 2023. № 1. С. 10–14. https://doi.org/10.17513/srts.1419

3. *Козар Б.А., Кугуракова В.В., Сахибгареева Г.Ф.* Структуризация сущностей естественного текста с использованием нейронных сетей для генерации трехмерных сцен // Программные продукты и системы. 2022. Т. 35. № 3. С. 329–339. https://doi.org/10.15827/0236-235X.139.329-339.

4. *Пантюхин Д.В.* Нейронные сети синтеза речи голосовых помощников и поющих автоматов // Речевые технологии/Speech Technologies. 2021. № 3-4. С. 3–16. https://doi.org/10.58633/2305-8129_2021_3-4_3

5. *Шамансуров Ш.* Влияние искусственного интеллекта на развитие области синхронного перевода // Oriental renaissance: Innovative, educational, natural and social sciences. 2023. Т. 3. № 23. С. 305–309.
https://doi.org/10.5281/zenodo.10365801

6. *Pang B., Nijkamp E., Wu Y.N.* Deep learning with tensorflow: a review // Journal of Educational and Behavioral Statistics. 2020. Vol. 45. No. 2. P. 227–248. https://doi.org/10.3102/1076998619872761

7. *Ketkar N., Ketkar N.* Introduction to Keras // Deep learning with python: a hands-on introduction. 2017. P. 97–111.
https://doi.org/10.1007/978-1-4842-2766-4_7

8. *Convolutional Neural Networks.*
URL: https://www.ibm.com/topics/convolutional-neural-networks

9. *Sapijaszko G., Mikhael W.B.* An overview of recent convolutional neural network algorithms for image recognition // 2018 IEEE 61st International midwest symposium on circuits and systems (MWSCAS). IEEE, 2018. P. 743–746. https://doi.org/10.1109/MWSCAS.2018.8623911

10. *He K. et al.* Deep residual learning for image recognition // Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. P. 770–778. https://doi.org/10.48550/arXiv.1512.03385

11. *Nguyen V.* Bayesian optimization for accelerating hyper-parameter tuning // 2019 IEEE second international conference on artificial intelligence and knowledge engineering (AIKE). IEEE. 2019. P. 302–305. https://doi.org/10.1109/AIKE.2019.00060

12. *Poojary R., Raina R., Mondal A.K.* Effect of data-augmentation on fine-tuned CNN model performance // IAES International Journal of Artificial Intelligence. 2021. Vol. 10. No. 1. P. 84. https://doi.org/10.11591/ijai.v10.i1.pp84-92

13. *Tian Y., Zhang Y.* A comprehensive survey on regularization strategies in machine learning // Information Fusion. 2022. Vol. 80. P. 146–166. https://doi.org/10.1016/j.inffus.2021.11.005

14. Fruits 360. URL: https://www.kaggle.com/datasets/moltean/fruits

15. PyCharm, Quick start guide. URL: https://www.jetbrains.com/help/pycharm/quick-start-guide.html

16. MacOS. URL: https://www.apple.com/za/macos/what-is/

17. *Zhang Z.* Analysis of the Advantages of the M1 CPU and Its Impact on the Future Development of Apple // 2021 2nd International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE). IEEE, 2021. P. 732–735. https://doi.org/10.1109/ICBASE53849.2021.00143

18. Tensorflow Macos. URL: https://developer.apple.com/metal/tensorflow-plugin/

19. Conv2D layer. URL: https://keras.io/api/layers/convolution_layers/convolution2d/

20. MaxPooling2D layer. URL: https://keras.io/api/layers/pooling_layers/max_pooling2d/

21. Flatten layer. URL: https://keras.io/api/layers/reshaping_layers/flatten/

22. Dense layer. URL: https://keras.io/api/layers/core_layers/dense/

23. Dropout layer.
URL: https://keras.io/api/layers/regularization_layers/dropout/
24. Layer activation functions, Softmax function.
URL: https://keras.io/api/layers/activations/#softmax-function

IMAGE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS

S. A. Filippov[0009-0007-9015-7982]

Institute of Information Technologies and Intelligent Systems of Kazan Federal University, 35 Kremlevskaya str., Kazan, 420008

woppilif@icloud.com

Abstract

Nowadays, many different tools can be used to classify images, each of which is aimed at solving a certain range of tasks. This article provides a brief overview of libraries and technologies for image classification. The architecture of a simple convolutional neural network for image classification is built. Image recognition experiments have been conducted with popular neural networks such as VGG 16 and ResNet 50. Both neural networks have shown good results. However, ResNet 50 overfitted due to the fact that the dataset contained the same type of images for training, since this neural network has more layers that allow reading the attributes of objects in the images. A comparative analysis of image recognition specially prepared for this experiment was carried out with the trained models.

***Keywords**:* image recognition, neural network, convolutional neural network, image classification, machine learning

REFERENCES

1. *Fisko D.V.* Overview of methods for conditional image synthesis // Aktual'nye voprosy sovremennoj nauki i tehnologij. 2021. S. 57–62.
2. *Mosin E.D., Belov Yu.S.* Music generation using a bidirectional recurrent neural network // Nauchnoe obozrenie. Technical science. 2023. No. 1. P. 10–14. https://doi.org/10.17513/srts.1419

3. *Kozar B.A., Kugurakova V.V., Sakhibgareeva G.F.* Structuring natural text entities using neural networks for generating 3d-scenes // Software & Systems. 2022. V. 35. No. 3. S. 329–339. https://doi.org/10.15827/0236-235X.139.329-339.

4. *Pantiukhin D.V.* Neural networks for speech synthesis of voice assistants and singing machines // Rechevye tehnologii. 2021. No. 3-4. S. 3–16. https://doi.org/10.58633/2305-8129_2021_3-4_3

5. *Shamansurov Sh.* Vlijanie iskusstvennogo intellekta na razvitie oblasti sinhronnogo perevoda // Oriental renaissance: Innovative, educational, natural and social sciences. 2023. V. 3. No. 23. S. 305–309. https://doi.org/10.5281/zenodo.10365801

6. *Pang B., Nijkamp E., Wu Y.N.* Deep learning with tensorflow: A review // Journal of Educational and Behavioral Statistics. 2020. Vol. 45. No. 2. P. 227–248. https://doi.org/10.3102/1076998619872761

7. *Ketkar N., Ketkar N.* Introduction to Keras // Deep learning with python: a hands-on introduction. 2017. P. 97–111. https://doi.org/10.1007/978-1-4842-2766-4_7

8. *Convolutional Neural Networks.* URL: https://www.ibm.com/topics/convolutional-neural-networks

9. *Sapijaszko G., Mikhael W.B.* An overview of recent convolutional neural network algorithms for image recognition // 2018 IEEE 61st International midwest symposium on circuits and systems (MWSCAS). IEEE, 2018. P. 743–746. https://doi.org/10.1109/MWSCAS.2018.8623911

10. *He K. et al.* Deep residual learning for image recognition // Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. P. 770–778. https://doi.org/10.48550/arXiv.1512.03385

11. *Nguyen V.* Bayesian optimization for accelerating hyper-parameter tuning // 2019 IEEE second international conference on artificial intelligence and knowledge engineering (AIKE). IEEE. 2019. P. 302–305. https://doi.org/10.1109/AIKE.2019.00060

12. *Poojary R., Raina R., Mondal A.K.* Effect of data-augmentation on fine-tuned CNN model performance // IAES International Journal of Artificial Intelligence. 2021. Vol. 10. No. 1. P. 84. https://doi.org/10.11591/ijai.v10.i1.pp84-92

13. *Tian Y., Zhang Y.* A comprehensive survey on regularization strategies in machine learning // Information Fusion. 2022. Vol. 80. P. 146–166. https://doi.org/10.1016/j.inffus.2021.11.005

14. Fruits 360. URL: https://www.kaggle.com/datasets/moltean/fruits

15. PyCharm, Quick start guide. URL: https://www.jetbrains.com/help/pycharm/quick-start-guide.html

16. MacOS. URL: https://www.apple.com/za/macos/what-is/

17. *Zhang Z.* Analysis of the Advantages of the M1 CPU and Its Impact on the Future Development of Apple // 2021 2nd International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE). IEEE, 2021. P. 732–735. https://doi.org/10.1109/ICBASE53849.2021.00143

18. Tensorflow Macos. URL: https://developer.apple.com/metal/tensorflow-plugin/

19. Conv2D layer. URL: https://keras.io/api/layers/convolution_layers/convolution2d/

20. MaxPooling2D layer. URL: https://keras.io/api/layers/pooling_layers/max_pooling2d/

21. Flatten layer. URL: https://keras.io/api/layers/reshaping_layers/flatten/

22. Dense layer. URL: https://keras.io/api/layers/core_layers/dense/

23. Dropout layer. URL: https://keras.io/api/layers/regularization_layers/dropout/

24. Layer activation functions, Softmax function. URL: https://keras.io/api/layers/activations/#softmax-function

СВЕДЕНИЯ ОБ АВТОРЕ

ФИЛИППОВ Сергей Алексеевич – магистрант второго курса Института информационных технологий и интеллектуальных систем Казанского федерального университета.

Sergey Alekseevich FILIPPOV – second-year master's student at the Institute of Information Technologies and Intelligent Systems of the Kazan Federal University.

email: woppilif@icloud.com;

ORCID: 0009-0007-9015-7982

Материал поступил в редакцию 24 мая 2024 года

УДК 514.12, 514.14

МНОГОМЕРНАЯ ГЕОМЕТРИЯ НА ФАКУЛЬТАТИВНЫХ ЗАНЯТИЯХ СО ШКОЛЬНИКАМИ И СТУДЕНТАМИ МЛАДШИХ КУРСОВ

В. В. Шурыгин[1] [0000-0002-4325-214X], В. В. Шурыгин (мл.)[2] [0000-0001-9771-1447]

[1, 2]Казанский федеральный университет, Казань

[1]Vadim.Shurygin@kpfu.ru, [2]1Vadim.Shurygin@kpfu.ru

Аннотация

Рассмаотрены некоторые подходы к преподаванию многомерной геометрии на факультативных занятиях, направленные на развитие у школьников и студентов многомерной геометрической интуиции. Особое внимание уделено использованию групп преобразований при исследовании геометрии правильных многогранников.

Ключевые слова: Трехмерная сфера, многомерная геометрия, четырехмерная геометрия, четырехмерный куб, четырехмерный правильный многогранник, 24-ячейка.

МНОГОМЕРНОЕ ЕВКЛИДОВО ПРОСТРАНСТВО

Прямоугольная система координат *Oxyz* в трехмерном геометрическом пространстве E_3 задается точкой *O* и тройкой единичных взаимно ортогональных (перпендикулярных) векторов **i**, **j**, **k**. Набор (*O*; **i**,**j**,**k**) называется ортонормированным репером. Координатами вектора **a** из E_3 по отношению к реперу (*O*; **i**,**j**,**k**) являются коэффициенты $\{a_x, a_y, a_z\}$ разложения

$$\mathbf{a}=a_x\mathbf{i}+a_y\mathbf{j}+a_z\mathbf{k}$$

вектора **a** по векторам репера. Координатами точки *A* являются координаты (x_A,y_A,z_A) ее радиус-вектора $\mathbf{r}_A=\overrightarrow{OA}$:

$$\mathbf{r}_A=x_A\mathbf{i}+y_A\mathbf{j}+z_A\mathbf{k}.$$

Вектор $\overrightarrow{AB}$ можно представить в виде

$$\overrightarrow{AB}=\overrightarrow{AO}+\overrightarrow{OB}=\mathbf{r}_B-\mathbf{r}_A,$$

поэтому координатами вектора $\overrightarrow{AB}$ являются разности координат точек *B* и *A*. Скалярное произведение векторов (**a**, **b**) вычисляется по формуле

$$(\mathbf{a},\mathbf{b})=a_xb_x+a_yb_y+a_zb_z.$$

Модуль вектора **a** и угол φ между векторами **a** и **b** вычисляются по формулам

$$|\mathbf{a}| = \sqrt{(\mathbf{a},\mathbf{a})}, \quad \cos\varphi = \frac{(\mathbf{a},\mathbf{b})}{|\mathbf{a}||\mathbf{b}|}, \tag{1}$$

расстояние $d(A,B)$ между точками A и B вычисляется по формуле

$$d(A,B) = |\overrightarrow{AB}|. \tag{2}$$

Евклидово n-мерное пространство устроено аналогично трехмерному евклидову пространству. Приведем некоторые необходимые определения, относящиеся к строению этого пространства.

Множество точек этого пространства обозначается символом E_n, а множество векторов символом $\mathbf{E}_n$. Для векторов из $\mathbf{E}_n$ определены операции сложения **a** + **b** и умножения вектора на вещественное число $\lambda\mathbf{a}$, в частности, можно образовывать линейные комбинации

$$\lambda^1\mathbf{a}_1+\lambda^2\mathbf{a}_2+\ldots+\lambda^m\mathbf{a}_m \tag{3}$$

векторов $\mathbf{a}_1, \mathbf{a}_2,\ldots,\mathbf{a}_m$. Линейная комбинация, все коэффициенты которой равны нулю, является нулевым вектором. Такая линейная комбинация называется тривиальной. Если нулевому вектору равна только тривиальная линейная комбинация векторов $\mathbf{a}_1, \mathbf{a}_2,\ldots, \mathbf{a}_m$, эти векторы называются линейно независимыми, при этом никакой из этих векторов нельзя представить в виде линейной комбинации других. В противном случае векторы $\mathbf{a}_1, \mathbf{a}_2,\ldots,\mathbf{a}_m$ называются линейно зависимыми. Множество $\mathbf{E}_n$ с операциями сложения векторов и умножения вектора на число является векторным пространством размерности n, то есть любые n+1 векторов в $\mathbf{E}_n$ являются линейно зависимыми, и имеется набор $\{\mathbf{e}_i\}$, i=1,…,n, состоящий из линейно независимых векторов. Такой набор $\{\mathbf{e}_i\}$, i=1,…,n, называется базисом в $\mathbf{E}_n$. Всякий вектор **a** из $\mathbf{E}_n$ можно представить в виде линейной комбинации

$$\mathbf{a}=a^1\mathbf{e}_1+a^2\mathbf{e}_2+\ldots+a^n\mathbf{e}_n \tag{4}$$

векторов базиса. Коэффициенты $a^1, a^2,\ldots,a^n$ разложения (4) называются координатами вектора **a**.

В пространстве $\mathbf{E}_n$ определена операция скалярного произведения векторов, то есть билинейное соответствие (билинейная форма)

$$g\colon \mathbf{E}_n\times\mathbf{E}_n \ni \{\mathbf{a}, \mathbf{b}\}\to(\mathbf{a},\mathbf{b})\in\mathbf{R},$$

относящее двум векторам вещественное число, такое что соответствие

$$\mathbf{E}_n \ni \mathbf{a}\to(\mathbf{a}, \mathbf{a})\in\mathbf{R}$$

является положительно определенной квадратичной формой. Таким образом,

скалярный квадрат $\mathbf{a}^2=(\mathbf{a}, \mathbf{a})$ всякого ненулевого вектора является положительным числом. Число $|\mathbf{a}|=\sqrt{(\mathbf{a},\mathbf{a})}$ называется модулем или длиной вектора **a**. Если (**a**, **b**)=0, векторы **a** и **b** называются ортогональными.

Базис $\{\mathbf{e}_i\}$, i=1,…,n, в $\mathbf{E}_n$ называется ортонормированным, если он состоит из взаимно ортогональных векторов единичной длины.

В координатах в $\mathbf{E}_n$, определяемых ортонормированным базисом, скалярное произведение векторов вычисляется по формуле

$$(\mathbf{a}, \mathbf{b})=a^1b^1+a^2b^2+\ldots+a^nb^n.$$

При этом модуль вектора **a** и угол φ между векторами **a** и **b** вычисляются по формулам (1).

Имеют место соотношения между точками и векторами, аналогичные соотношениям в E_3: двум точкам A и B пространства E_n однозначно относится вектор $\overrightarrow{AB} \in \mathbf{E}_n$, а точке $A \in E_n$ и вектору $\mathbf{v} \in \mathbf{E}_n$ соответствует единственная точка $B \in E_n$, такая что $\mathbf{v}=\overrightarrow{AB}$, и при этом выполняется равенство треугольника $\overrightarrow{AB}+\overrightarrow{BC}=\overrightarrow{AC}$.

Расстояние $d(A,B)$ между точками A и B в E_n определяется формулой (2).

Набор $(O,\{\mathbf{e}_i\})$, состоящий из точки $O \in E_n$ и ортонормированного базиса $\{\mathbf{e}_i\}$ пространства $\mathbf{E}_n$, называется ортонормированным репером в E_n. Координатами точки A в системе координат, определяемой ортонормированным репером (такая система координат называется прямоугольной), являются координаты ее радиус-вектора $\mathbf{r}_A=\overrightarrow{OA}$:

$$\mathbf{r}_A=x_A{}^1\mathbf{e}_1+x_A{}^2\mathbf{e}_2+\ldots+x_A{}^n\mathbf{e}_n.$$

Все евклидовы n-мерные пространства изоморфны (эквивалентны). Изоморфизм (эквивалентность) пространств E_n и E'_n осуществляется выбором в них некоторых ортонормированных реперов $(O, \{\mathbf{e}_i\})$ и $(O', \{\mathbf{e}'_i\})$ и установлением соответствия между точками, имеющими одинаковые координаты. Евклидово пространство $\mathbf{R}^n$, точки и векторы которого представляют собой наборы из n чисел (x^i) и $\{a^i\}$, i=1,…,n, называется *стандартным* n-мерным евклидовым пространством.

В векторном пространстве $\mathbf{E}_n$ подпространство $\mathbf{L}_m$ размерности m задается набором из m линейно независимых векторов $\mathbf{a}_1$, $\mathbf{a}_2$, …, $\mathbf{a}_m$ и состоит из всех векторов, представимых в виде линейных комбинаций (3). Векторы $\mathbf{x} \in \mathbf{E}_n$, ортогональные всем векторам $\mathbf{a}_1$, $\mathbf{a}_2$,…,$\mathbf{a}_m$, являются решениями системы уравнений

$$(\mathbf{a}_1, \mathbf{x})=0, (\mathbf{a}_2, \mathbf{x})=0,\ldots,(\mathbf{a}_m, \mathbf{x})=0,$$

и образуют подпространство $\mathbf{L}_m^{\perp}$ размерности $n-m$, называемое ортогональным дополнением подпространства $\mathbf{L}_m$. Если $\mathbf{b}_\alpha$, $\alpha=1,\ldots,n-m$, – линейно независимые решения этой системы, то подпространство $\mathbf{L}_m$ может быть задано также следующей системой уравнений:

$$(\mathbf{b}_1, \mathbf{x})=0,\ (\mathbf{b}_2, \mathbf{x})=0,\ldots,(\mathbf{b}_{n-m}, \mathbf{x})=0, \tag{5}$$

или, в координатах, системой линейных однородных уравнений

$$b_\alpha{}^1x^1+b_\alpha{}^2x^2+\ldots+b_\alpha{}^nx^n=0,\quad \alpha=1,\ldots,n-m. \tag{6}$$

Плоскость π_m размерности m (m-плоскость) в пространстве E_n определяется принадлежащей ей точкой M_0 и m-мерным направляющим подпространством $\mathbf{L}_m$ и состоит из точек M, удовлетворяющих соотношению

$$\overrightarrow{M_0M}\in\mathbf{L}_m\Leftrightarrow\mathbf{r}-\mathbf{r}_0\in\mathbf{L}_m, \tag{7}$$

где $\mathbf{r}=\overrightarrow{OM}$ и $\mathbf{r}_0=\overrightarrow{OM_0}$ — радиус-векторы точек M и M_0. Если (x^i) — координаты точки M, а $(x_0{}^i)$ — координаты точки M_0, то из (6) следует, что m-плоскость π_m задается системой линейных уравнений

$$b_\alpha{}^1x^1+b_\alpha{}^2x^2+\ldots+b_\alpha{}^nx^n=b_\alpha,\ \alpha=1,\ldots,n-m,$$

где $b_\alpha=b_\alpha{}^1x_0{}^1+b_\alpha{}^2x_0{}^2+\ldots+b_\alpha{}^nx_0{}^n$.

Задача. Убедиться, что всякая m-плоскость является евклидовым пространством размерности m.

Плоскость π_{n-1} размерности $n-1$ в пространстве E_n называется *гиперплоскостью*. Всякий вектор из одномерного ортогонального дополнения направляющего подпространства гиперплоскости π_{n-1} называется *нормальным вектором* этой гиперплоскости. Из изложенного выше следует, что гиперплоскость с нормальным вектором $\mathbf{a}$, проходящая через точку M_0 с радиус-вектором $\mathbf{r}_0$, имеет уравнение

$$(\mathbf{a}, \mathbf{r}-\mathbf{r}_0)=0\ \Leftrightarrow\ a^1x^1+a^2x^2+\ldots+a^nx^n+a=0, \tag{8}$$

где $a=-(a^1x_0{}^1+a^2x_0{}^2+\ldots+a^nx_0{}^n)$. Имеет место не зависящая от размерности пространства формула расстояния от точки $M_1(\mathbf{r}_1)$ до гиперплоскости (8)

$$\operatorname{dist}(M_1,\pi)=\frac{|(\mathbf{a},\mathbf{r}_1-\mathbf{r}_0)|}{|\mathbf{a}|}, \tag{9}$$

которая в координатах принимает вид (см., например, [13], с. 61)

$$\operatorname{dist}(M_1,\pi)=\frac{|a^1x_1^1+a^2x_1^2+\ldots+a^nx_1^n+a|}{(a^1)^2+(a^2)^2+\ldots+(a^n)^2}.$$

Гиперплоскость (8) выделяет в пространстве E_n два *замкнутых полупространства*, определяемых соответственно неравенствами

$$(\mathbf{a}, \mathbf{r}-\mathbf{r}_0)\le 0 \text{ и } (\mathbf{a}, \mathbf{r}-\mathbf{r}_0)\ge 0. \qquad (10)$$

Гиперплоскость (8) входит в каждое из полупространств (10) и ограничивает его; если ее удалить из полупространства, получится открытое полупространство, состоящее из точек, лежащих (строго) по одну сторону от гиперплоскости.

Плоскость l размерности один называется *прямой*. Направляющее подпространство $\mathbf{L}_1$ прямой l задается любым ненулевым вектором $\mathbf{a}\in\mathbf{L}_1$, называемым направляющим вектором прямой l. Условие (7) в случае прямой принимает вид

$$\overrightarrow{M_0M} \;||\; \mathbf{a} \Leftrightarrow \mathbf{r}-\mathbf{r}_0=t\mathbf{a},$$

поэтому прямая в E_n с направляющим вектором $\mathbf{a}$, проходящая через точку M_0, определяется параметрическим уравнением

$$\mathbf{r}=\mathbf{r}_0+t\mathbf{a},\ t\in\mathbf{R}. \qquad (11)$$

В частности, уравнение прямой, проходящей через две точки A и B, имеет вид

$$\mathbf{r}=\mathbf{r}_A+t\overrightarrow{AB},\quad t\in\mathbf{R}. \qquad (12)$$

Подмножество $[A,B]$ прямой (12), состоящее из точек, для которых $t\in[0;1]$, называется (замкнутым) *отрезком* с концами A и B или отрезком, соединяющим точки A и B.

В координатах уравнение (11) принимает вид

$$x^i=x_0{}^i+ta^i,\quad i=1,\ldots,n,\quad t\in\mathbf{R}.$$

ШАР И КУБ

Естественные объекты, имеющиеся в евклидовом пространстве любой размерности, это шар и куб.

Шаром, точнее, замкнутым шаром, в E_n радиуса $a>0$ с центром в точке M_0 называется подмножество

$$B[M_0,a]=\{M\in E_n \mid d(M_0,M)\le a\},$$

состоящее из точек, удаленных от точки M_0 на расстояние не большее a. Множество точек, удаленных от точки M_0 на расстояние, равное a, называется *сферой*. Сфера является границей шара.

Если в E_n выбрана прямоугольная система координат с началом в точке M_0,

то шар и сфера задаются соответственно следующими неравенством и уравнением:

$$(x^1)^2+(x^2)^2+\ldots+(x^n)^2\le a^2 \quad \text{и} \quad (x^1)^2+(x^2)^2+\ldots+(x^{n+1})^2=a^2.$$

Сфера с уравнением

$$(x^1)^2+(x^2)^2+\ldots+(x^n)^2=1$$

в пространстве $\mathbf{R}^{n+1}$ называется *стандартной n-мерной сферой* (*n-сферой*) и обозначается $\mathbf{S}^n$.

Куб размерности n (*n-куб*) с длиной ребра, равной *a*, представляет собой подмножество в евклидовом пространстве E_n, определяемое в некоторой прямоугольной системе координат системой неравенств

$$0\le x \le a, \quad i=1,\ldots,n.$$

Все *n*-мерные кубы подобны, поэтому для изучения их геометрии можно рассматривать куб C_n, заданный системой неравенств

$$0\le x \le 1, \quad i=1,\ldots,n. \tag{13}$$

Куб, заданный системой неравенств

$$-1\le x^i\le 1, \quad i=1,\ldots,n \tag{14}$$

в пространстве $\mathbf{R}^n$, называется *стандартным n-мерным кубом*. Он симметричен относительно начала координат и координатных плоскостей всех размерностей.

Если из *n* неравенств в системе неравенств (13), определяющих куб C_n, выбрать некоторые *m*, а остальные положить равными нулю или единице, то получится *m*-куб, лежащий в соответствующей координатной плоскости или в плоскости, параллельной координатной. Этот *m*-куб является *m*-гранью (гранью размерности *m*) куба C_n.

Задача. Показать, что число *m*-граней у *n*-куба равно $2^{n-m}C_n^m$.

Граница *n*-куба состоит из (*n*−1)-граней, число которых равно 2*n*.

Из определения *n*-куба (13) следует, что он как множество является объединением (*n*−1)-кубов

$$C_{n-1}(t)=\{\mathbf{x}\in C_n \mid x^n = t, \ t\in[0;1]\},$$

являющихся пересечениями куба C_n с плоскостями $x^n=t$, то есть C_n заметается в пространстве E_n при движении (*n*−1)-куба $C_{n-1}(t)$ в направлении *n*-ой координатной оси при изменении *n*-ой координаты от 0 до 1. При этом (*n*−2)-грани куба $C_{n-1}(t)$ в процессе движения заметают (*n*−1)-кубы, образующие вместе с (*n*−1)-

кубами $C_{n-1}(0)$ и $C_{n-1}(1)$ границу куба C_n. На рис. 1 этот факт проиллюстрирован для случая n=3, а на рис. 2 изображена конструкция 4-куба.

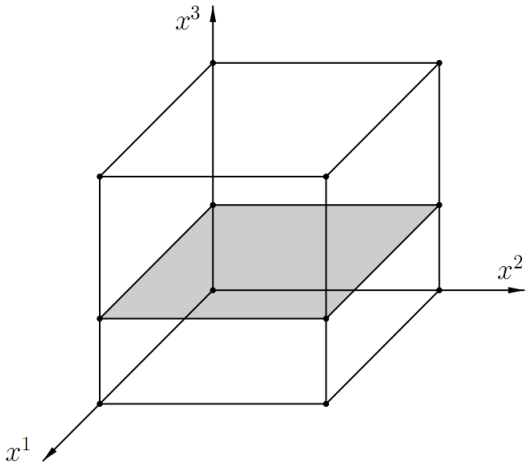


Рис. 1.

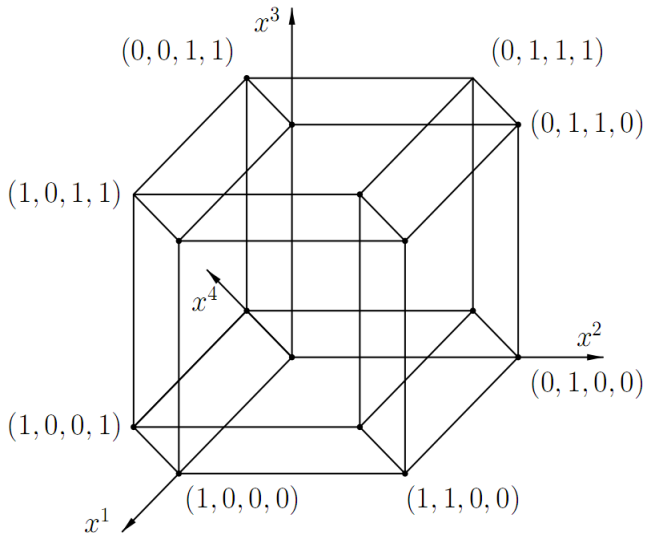


Рис. 2.

Замечание. Надо иметь в виду, что все рисунки многомерных объектов (как и трехмерных) представляют собой проекции некоторых многомерных конфигураций на двумерную плоскость.

Задача. Сдвигая 4-куб, заданный в координатной плоскости x^5=0 пространства E_5, в направлении пятой координатной оси, получить изображение 5-куба,

аналогичное рис. 2 (удобнее построить изображение параллелепипеда, у которого ребра, направленные параллельно пятой координатной оси, имеют длину, большую длин ребер 4-куба).

Уравнениями (каждым по отдельности)

$$x^1=0,\ x^1=1,\ x^2=0,\ x^2=1,\ x^3=0,\ x^3=1,\ x^4=0,\ x^4=1,$$

задаются восемь 3-плоскостей, пересечения которых с 4-кубом дают восемь 3-кубов, являющихся 3-гранями 4-куба. В совокупности эти 3-грани образуют границу 4-куба в E_4 или его поверхность. У каждых двух 3-граней, за исключение противоположных, лежащих в параллельных 3-плоскостях, имеется общая 2-грань (квадрат), по которой эти 3-грани соединяются. Например, грани $x^1=0$ и $x^4=1$ соединены по общей 2-грани, определяемой системой уравнений $x^1=0$, $x^4=1$. Если в 3-кубе, расположенном в 4-мерном пространстве, зафиксировано положение 2-грани (квадрата), то 3-куб может вращаться вокруг этой грани, как квадрат в трехмерном пространстве может вращаться вокруг неподвижного ребра. Ребро, перпендикулярное зафиксированной грани, может вращаться вокруг своей вершины в 2-плоскости, перпендикулярной этой грани.

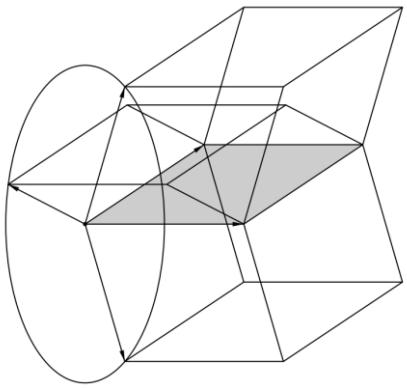


Рис. 3.

Смежные 3-грани границы 4-куба можно вращать вокруг их общей 2-грани. Если границу 4-куба представить как набор из восьми 3-кубов, соединенных по некоторым граням, то, разъединяя часть соединенных граней (разрезая границу по некоторым 2-граням) и вращая кубы, составляющие границу таким образом, чтобы все они оказались в одной трехмерной плоскости, можно получить развертку границы 4-куба, представляющую собой многогранник в трехмерном пространстве.

Совокупность 3-граней и 2-граней можно рассматривать как граф, вершинами которого являются 3-грани, а ребрами — 2-грани, соединяющие соответствующие 3-грани. Тогда для получения развертки границы 4-куба, надо удалить такой набор ребер указанного графа, чтобы остался связный граф без циклов. Одна из разверток определяется, например, графом

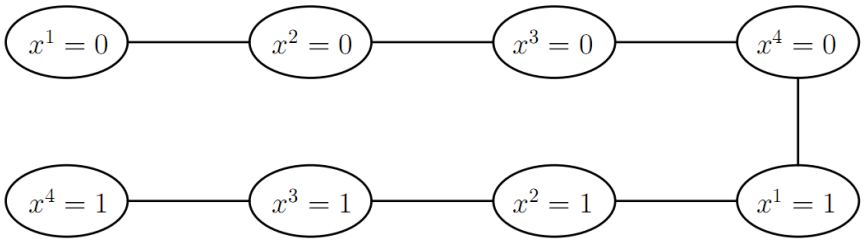


Рис. 4.

Графу, изображенному на рис. 5,

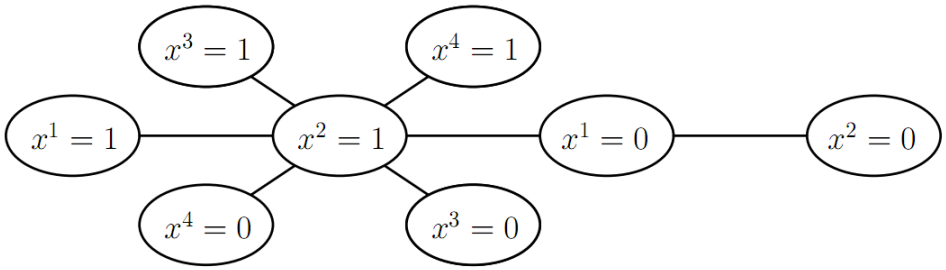


Рис. 5.

соответствует развертка, приведенная на рис. 6.

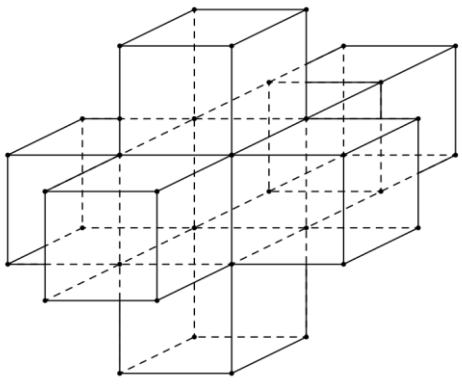


Рис. 6.

Если смотреть на трехмерный куб со стороны одной из его 2-граней, представляя его как тело в реальном (физическом) пространстве, то ребра, перпендикулярные этой 2-грани, будут восприниматься как лежащие на пересекающихся прямых. В проективной интерпретации точка пересечения этих прямых является несобственной точкой пространства E_3, рассматриваемого как часть проективного пространства P_3 (см., например, [12]). При этом трехмерный куб будет изображаться следующим рисунком, на котором две 2-грани куба изображаются квадратами, а четыре трапециями:

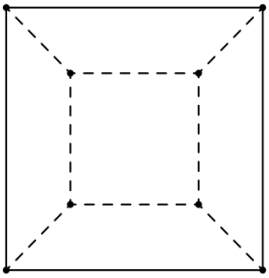


Рис. 7.

Аналогичным образом можно изобразить 4-куб, располагая его ребра, параллельные 4-ой координатной оси, на пересекающихся прямых. При этом его 3-грани будут изображаются двумя кубами и шестью усеченными пирамидами (см. рис. 8):

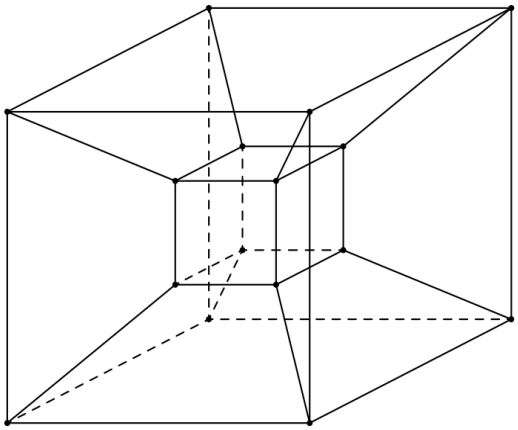


Рис. 8.

Рис. 8 позволяет наглядным образом продемонстрировать, как получается

развертка 4-куба, представленная на рис. 6: у всех граничных 3-кубов, изображенных усеченными пирамидами, кроме одного, надо оставить не разрезанными только 2-грани, общие с большим (наружным) 3-кубом, а у одного оставить не разрезанной еще и общую 2-грань с малым (внутренним) 3-кубом, а потом «выдавить» (повернуть вокруг 2-граней наружного куба) все пирамиды и внутренний куб «наружу».

Вращая (n−1)-грани n-куба вокруг (n−2)-граней, можно аналогично четырехмерному случаю получить развертку границы n-куба как некоторый многогранник в (n−1)-мерном пространстве.

Задача. Описать развертки границы 5-куба и n-куба аналогичные развертке, соответствующей рис. 6.

По определению, сферой $S(M_0,a)$ радиуса $a>0$ с центром в точке M_0 с радиус-вектором $\mathbf{r}_0$ в пространстве E_n называется подмножество в E_n, состоящее из точек, удаленных от точки M_0 на расстояние a. Таким образом, сфера $S(M_0,a)$ определяется уравнением

$$(\mathbf{r} - \mathbf{r}_0)^2=a^2. \qquad (15)$$

Задача. Все вершины стандартного n-куба (14) находятся на одинаковом расстоянии от начала координат — центра n-куба, поэтому вокруг этого n-куба можно описать сферу. Вычислить ее радиус.

Задача. Использовав развертку, вычислить кратчайшее расстояние вдоль границы n-куба между двумя противоположными вершинами.

Решение этой задачи, а также некоторые другие задачи из области многомерной геометрии, можно найти в материалах математической олимпиады [6].

МНОГОГРАННИКИ В E_n

Куб является примером многогранника в пространстве E_n, более того, куб является правильным многогранником. Для аккуратного рассмотрения других многогранников в E_n необходимо сформулировать ряд определений.

Открытым шаром в E_n радиуса $a>0$ с центром в точке M_0 называется подмножество

$$B(M_0,a)=\{M\in E_n \mid d(M_0,M)<a\},$$

состоящее из точек, удаленных от точки M_0 на расстояние строго меньшее a. Сфера $S(M_0,a)$ является границей шара $B(M_0,a)$.

Пусть S — подмножество в E_n. Точка $M \in S$ называется *внутренней точкой* подмножества S, если в S содержится некоторый открытый шар с центром в точке M. Множество всех внутренних точек подмножества S называется *внутренностью* подмножества S.

Подмножество $S \subset E_n$ называется ограниченным, если оно содержится в некотором открытом шаре $B(M_0,a)$ (можно при этом всегда указать такой шар с центром в начале координат).

Подмножество $S \subset E_n$ называется *выпуклым*, если вместе с каждыми двумя своими точками A, $B \in S$ оно содержит весь отрезок $[A, B]$. Наименьшее выпуклое множество, содержащее данное подмножество S, называется его *выпуклой оболочкой*.

Выпуклым многогранником называется ограниченное подмножество $S \subset E_n$ с непустой внутренностью, являющееся пересечением

$$S = \bigcap_{\alpha=1}^{N} R_\alpha \tag{16}$$

конечного числа замкнутых полупространств R_α.

При рассмотрении выпуклого многогранника естественно предполагать, что среди полупространств R_α нет лишних.

Задача. Показать, что выпуклый многогранник является выпуклым множеством.

Пересечение $S \cap H_\alpha$ выпуклого многогранника S с гиперплоскостью H_α, ограничивающей полупространство R_α из определения (16) этого многогранника, называется *гранью* или ($n-1$)-*гранью* многогранника S.

Задача. Гиперплоскость H_α является евклидовым пространством размерности $n-1$. Убедиться, что грань $S \cap H_\alpha$ является выпуклым многогранником в пространстве H_α.

Поскольку грань многогранника S является выпуклым многогранником, то можно рассматривать ее грани. Грани ($n-1$)-граней многогранника S называются его ($n-2$)-гранями. По индукции определяются k-грани многогранника S как грани его ($k+1$)-граней. 0-грани многогранника S называются его вершинами, а 1-грани — его ребрами.

Задача. Показать, что выпуклый многогранник является выпуклой оболочкой множества своих вершин.

Далее под многогранником S всегда имеется в виду выпуклый многогранник.

Флагом n-мерного многогранника называется набор его k-граней по одной для каждого k=0, 1,…,n−1:

$$\boldsymbol{F}=\{F_k,\ k=0,\ 1,\ldots,\ n-1\}=\{F_0,\ F_1,\ \ldots,\ F_{n-1}\},$$

такой что $F_k \subset F_{k+1}$, то есть набор, состоящий из некоторой вершины F_0, ребра F_1, содержащего вершину F_0, 2-грани F_2, содержащей ребро F_1, и так далее.

Движением пространства E_n называется взаимно однозначное отображение φ: $E_n \ni M \to M' \in E_n$, которое устанавливает соответствие между точками, имеющими одинаковые координаты по отношению к двум ортонормированным реперам $(O,\{\mathbf{e}_i\})$ и $(O',\{\mathbf{e}'_i\})$, то есть если $\overrightarrow{OM}=x^1\mathbf{e}_1+x^2\mathbf{e}_2+\ldots+x^n\mathbf{e}_n$, то $\overrightarrow{O'M'}=x^1\mathbf{e}'_1+x^2\mathbf{e}'_2+\ldots+x^n\mathbf{e}'_n$.

Правильным многогранником в E_n называется выпуклый многогранник P, такой что для любых двух флагов $\boldsymbol{F}$ и $\boldsymbol{F}'$ существует движение φ пространства E_n, при котором многогранник P отображается на себя и флаг $\boldsymbol{F}$ отображается на флаг $\boldsymbol{F}'$. Все такие движения называются *симметриями* ([8, с. 210]) правильного многогранника, они образуют группу, обозначаемую $G(P)$.

Задача. Показать, что для двух заданных флагов симметрия правильного многогранника P, при которой один флаг переходит в другой, единственна и, следовательно, число элементов в группе $G(P)$ равно числу флагов.

Задача. Показать, что число элементов в группе куба $G(C_n)$ равно $2^n n!$.

Задача. Используя результат предыдущей задачи, показать, что группа стандартного куба состоит из всех отображений, порождаемых перестановками координатных осей и симметриями относительно координатных гиперплоскостей.

Пусть A_α, α=1,…,m, — вершины правильного многогранника P и $\mathbf{r}_\alpha$ — радиус-вектор вершины A_α по отношению к некоторому ортонормированному реперу. Точка C с радиус-вектором $\mathbf{r}_C=(1/m)(\mathbf{r}_0+\mathbf{r}_1+\ldots+\mathbf{r}_m)$ остается неподвижной при всех преобразованиях группы $G(P)$. Она называется центром многогранника P.

СИМПЛЕКС

Выпуклая оболочка множества точек $\{A_1,A_2,\ \ldots,\ A_{n+1}\}$ в E_n, находящихся в общем положении (то есть таких, что векторы $\{\overrightarrow{A_1A_i}\}$, i=2,…,n+1, линейно независимы), называется *симплексом* (*n-симплексом*) с вершинами A_1, A_2,…, A_{n+1}. Если

все расстояния между вершинами симплекса равны, симплекс называется правильным. Правильный n-симплекс, вершина A_i которого расположена на i-ой координатной оси $(n+1)$-мерного пространства E_{n+1} и имеет радиус-вектор $\mathbf{e}_i$, задается следующей системой, состоящей из уравнения и неравенств:

$$x^1+x^2+\ldots+x^{n+1}=1,\ x\geq 0. \qquad (17)$$

Этот симплекс расположен в гиперплоскости пространства E_{n+1}. Симплекс, заданный в $\mathbf{R}^{n+1}$ системой (17), называется *стандартным n-симплексом*.

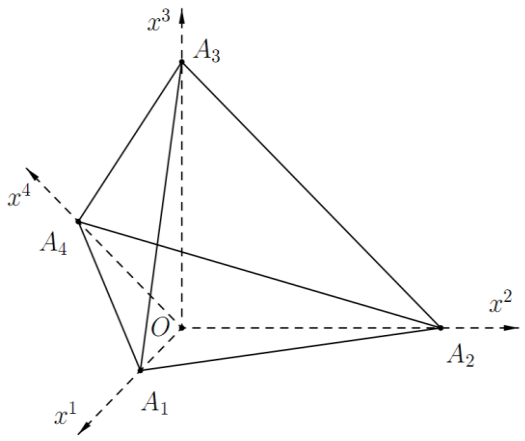


Рис. 9.

Грани n-симплекса являются $(n-1)$-симплексами, а k-грани — k-симплексами. В случае стандартного n-симплекса грань, противоположная вершине A_i, задается уравнением $x^i=0$.

Грани n-симплекса, имеющие уравнения $x^i=0$ и $x=0$, можно вращать вокруг их общей $(n-2)$-грани, задаваемой системой уравнений $x^i=0$, $x=0$. Использовав это, можно построить развертки границы симплекса. Графу на рис. 10

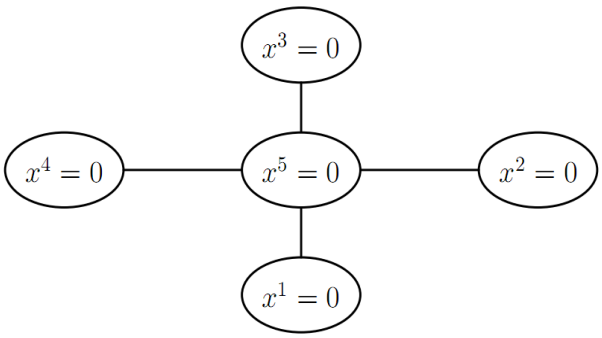


Рис. 10.

соответствует развертка границы 4-симплекса, изображенная на рис. 11.

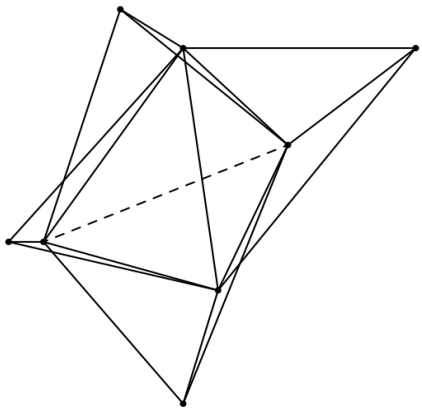


Рис. 11.

Эта развертка аналогична развертке тетраэдра, изображенной на рис. 12.

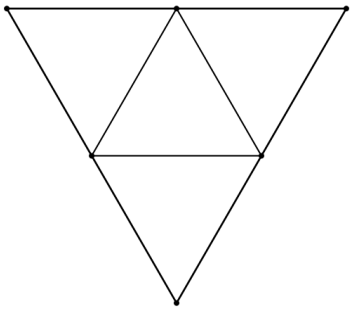


Рис. 12.

Задача. Описать другие возможные развертки для границы 4-симплекса. Привести пример развертки границы n-симплекса.

Задача. Подсчитать число элементов в группе симметрий правильного n-симплекса. Показать, что для стандартного симплекса эта группа состоит из всех отображений, порождаемых перестановками координатных осей пространства $\mathbf{R}^{n+1}$.

ДВОЙСТВЕННЫЙ ПРАВИЛЬНЫЙ МНОГОГРАННИК

Пусть P — правильный многогранник в E_n с вершинами A_α, α=1,…,m, которые заданы радиус-векторами $\mathbf{a}_\alpha$ относительно ортонормированного репера с началом в центре многогранника. Двойственным к P называется многогранник P^*, представляющий собой следующее пересечение полупространств:

$$P^* = \bigcap_{\alpha=1}^{m} R_\alpha^*, \ R_\alpha^* = \{M(\mathbf{r}) \mid (\mathbf{a}_\alpha, \mathbf{r}) \le 1\}. \quad (18)$$

Из определения (18) следует, что вершине A_α многогранника P соответствует грань ($n-1$)-грань) многогранника P^*, лежащая в гиперплоскости $H_\alpha^* = \{M(\mathbf{r}) \mid (\mathbf{a}_\alpha, \mathbf{r})=1\}$, нормальным вектором которой является радиус-вектор $\mathbf{a}_\alpha$ вершины A_α. Ребру $[A_\alpha A_\beta]$ многогранника P при этом будет соответствовать ($n-2$)-грань многогранника P^*, лежащая в пересечении $H_\alpha^* \cap H_\beta^*$. В общем случае k-грани F многогранника P, определенной $k+1$ вершинами, радиус-векторы которых $\mathbf{a}_{\alpha_1}$, $\mathbf{a}_{\alpha_2}$,..., $\mathbf{a}_{\alpha_k}$ линейно независимы, будет соответствовать ($n-k-1$)-грань F^* многогранника P^*, лежащая в пересечении $\bigcap_{u=1}^{k+1} H_{\alpha_u}^*$.

Задача. Показать, что отношение двойственности правильных многогранников — взаимное, то есть $(P^*)^*=P$.

Задача. Показать, что группы $G(P)$ и $G(P^*)$ симметрий многогранников P и P^* совпадают.

Задача. Показать, что выпуклая оболочка центров граней правильного многогранника P является правильным многогранником, подобным многограннику P^*.

КОКУБ

Правильный многогранник, двойственный кубу, называется *кокубом*. Кокубом трехмерного пространства является октаэдр. Кокуб пространства $\mathbf{R}^n$, двойственный стандартному кубу (14), называется *стандартным n-мерным кокубом*. Стандартный кокуб задается неравенством

$$|x^1|+|x^2|+\ldots+|x^n| \le 1.$$

Действительно, вершины стандартного куба имеют координаты

$$(\pm 1, \pm 1, \ldots, \pm 1), \quad (19)$$

следовательно, гиперплоскости, содержащие грани двойственного многогранника, имеют уравнения

$$\pm x^1 \pm x^2 \pm \ldots \pm x^{n+1} = 1, \quad (20)$$

и вершинами кокуба оказываются центры граней куба, то есть точки

$$(\pm 1, 0, 0, \ldots, 0), \ (0, \pm 1, 0, \ldots, \pm 1,), \ldots, \ (0, 0, \ldots, 0, \pm 1) \quad (21)$$

на координатных осях с координатами ±1.

Координатные гиперплоскости пространства E_n делят E_n на 2^n координатных n-гранных углов. В каждом таком координатном угле находится одна из вершин (19) стандартного куба, обозначим эту вершину A, и координаты всякой точки, лежащей внутри этого n-гранного угла, имеют такие же знаки, что и соответствующие координаты вершины A. Например, координатный угол, определяемый системой неравенств $x^i \geq 0$, $i=1,\ldots,n$, содержит вершину куба (1, 1, …, 1). Гиперплоскость (20), соответствующая точке A, отрезает от координатного угла симплекс, одной из вершин которого является начало координат, а оставшимися вершинами — вершины (21) кокуба, координаты которых совпадают с соответствующими координатами точки A. Этот симплекс составляет $(1/2^n)$-ю часть кокуба. Грань симплекса, противоположная началу координат, является и гранью кокуба. Таким образом, все $(n-1)$-грани кокуба являются правильными $(n-1)$-симплексами. Следовательно, и k-грани кокуба при любом k являются k-симплексами. В частности, грань кокуба, лежащая в координатном угле, определяемом системой неравенств $x^i \geq 0$, $i=1,\ldots,n$, является $(n-1)$-симплексом, определяемом системой $x^1+x^2+\ldots+x^n=1$, $x^i \geq 0$, $i=1,\ldots,n$.

4-симплекс с вершинами O, A_1, A_2, A_3, A_4 на рисунке 9 представляет собой 1/16 часть стандартного 4-кокуба, а 3-симплекс с вершинами A_1, A_2, A_3, A_4 (тетраэдр) является его гранью. Для того чтобы изобразить весь кокуб, на рис. 9 нужно добавить еще 15 граней – 3-симплексов, высекаемых 3-плоскостями (20) (при $n=4$) в координатных четырехгранных углах.

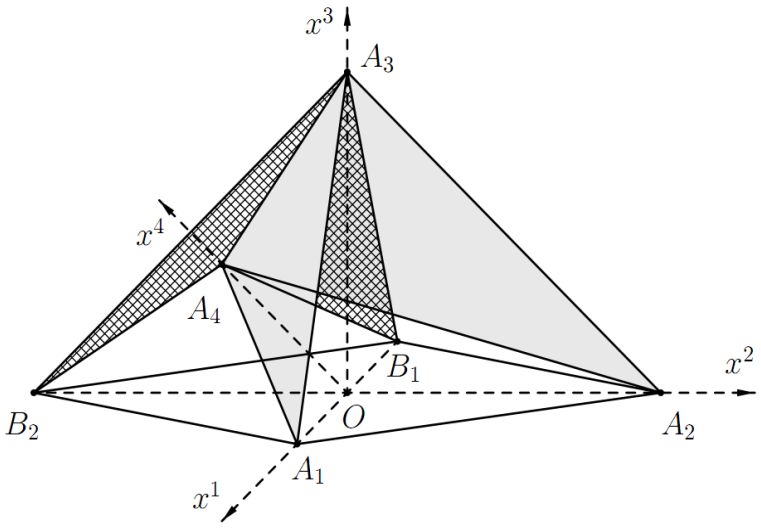


Рис. 13.

На рис. 13 изображена часть стандартного 4-кокуба, расположенная в 1/4 части пространства, определенной системой неравенств $x^3 \geq 0$, $x^4 \geq 0$, ограниченная четырьмя гранями – 3-симплексами $A_1A_2A_3A_4$, $B_1A_2A_3A_4$, $B_1B_2A_3A_4$, $A_1B_2A_3A_4$, где точки B_1 и B_2 – вершины кокуба с координатами (−1, 1, 1, 1) и (1, −1, 1, 1). Указанные 3-симплексы соединены (последовательно) общими 2-гранями $A_2A_3A_4$, $B_1A_3A_4$, $B_2A_3A_4$ и $A_1A_3A_4$, выделенными на рисунке. Полностью, со всеми гранями, 4-кокуб изображен на рис. 14.

Линиями большей толщины на рис. 14 изображен октаэдр, расположенный в 3-плоскости $Ox^1x^2x^3$. Кокуб представляет собой пару пирамид с вершинами в точках (0, 0, 0, 1) и (0, 0, 0, −1), надстроенных над этим октаэдром.

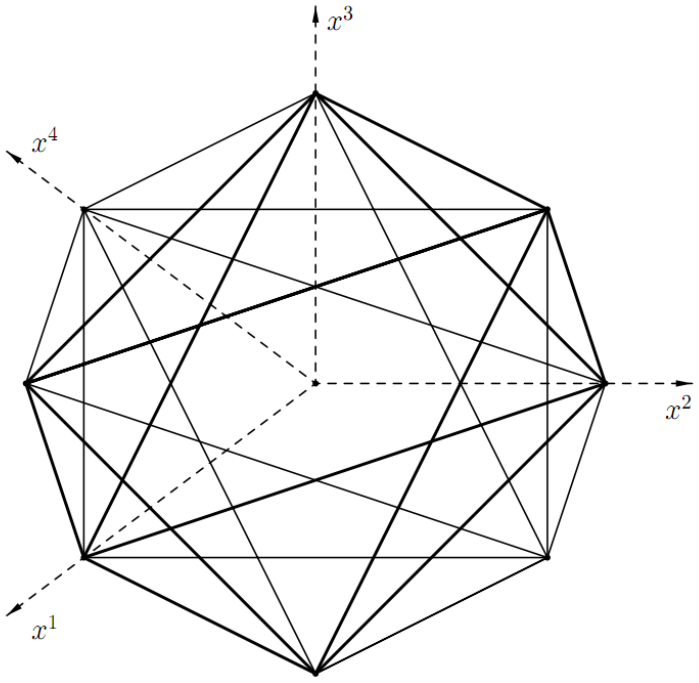


Рис. 14.

Задачи

1) Вычислить число k-граней n-кокуба.

2) Вычислить число элементов в группе кокуба, подсчитав число его флагов.

3) Построить граф, образованный 3-гранями и 2-гранями 4-кокуба, и графы, соответствующие развертками границы 4-кокуба, аналогично графам на рис. 4 и 5.

СЕЧЕНИЯ КУБА ГИПЕРПЛОСКОСТЯМИ, ОРТОГОНАЛЬНЫМИ ДИАГОНАЛИ КУБА

Прямая, соединяющая противоположные вершины куба, называется его диагональю. Одной из диагоналей куба (13) является прямая OE, проходящая через начало координат $O(0, 0, \ldots, 0)$ и противоположную точку $E(1, 1, \ldots, 1)$. Направляющий вектор диагонали $\mathbf{d} = \overrightarrow{OE}$ имеет координаты {1, 1, …, 1}, поэтому гиперплоскость, ортогональная диагонали, имеет уравнение

$$x^1+x^2+\ldots+x^n=a. \qquad (22)$$

При $a<0$ и $a>n$ куб (13) и гиперплоскость (22) не имеют общих точек, при $a=0$ и $a=n$ их общими точками являются вершины O и E соответственно. Если же $0<a<n$, то пересечение куба (13) и гиперплоскости (22) является выпуклым $(n-1)$-мерным многогранником, лежащим в гиперплоскости (22). Обозначим его $P(a)$. Какой вид может иметь многогранник $P(a)$ при различных значениях $a \in (0,n)$? Куб лежит в координатном угле, где $x^i \geq 0$, $i=1,\ldots,n$, а пересечение гиперплоскости (22) с этим углом является правильным симплексом. Поэтому многогранник $P(a)$ представляет собой правильный симплекс, от которого гранями куба (возможно) отрезаны некоторые части. При $a<1$ симплекс не может иметь общих точек с гранями $x^i=1$, $i=1,\ldots,n$, поэтому при $a<1$ многогранник $P(a)$ – симплекс. При $a=1$ вершины куба, лежащие на осях, совпадут с вершинами симплекса, и поэтому $P(1)$ – это стандартный симплекс. При $1<a<2$ грани $x^i=1$, $i=1,\ldots,n$, отрежут от симплекса углы, поэтому в этом случае $P(a)$ – симплекс с отрезанными углами.

Задача. Показать, что сечение 4-куба, заданного системой неравенств $0 \leq x^i \leq 1$, $i=1, 2, 3, 4$, 3-плоскостью $x^1+x^2+x^3+x^4=2$ представляет собой октаэдр (см. рис. 15).

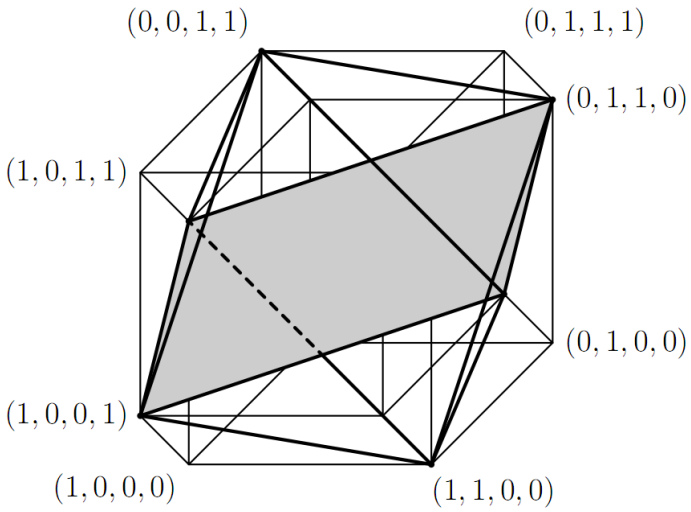


Рис. 15.

Задача. Описать сечения n-куба (13) гиперплоскостью $x^1+x^2+\ldots+x^n=m$, где m – целое число. Показать, что все такие гиперплоскости делят диагональ n-куба на n равных частей.

ЗВЕЗДА И СИМВОЛ ПРАВИЛЬНОГО МНОГОГРАННИКА

Поворотом правильного многогранника P с помощью некоторого элемента группы $G(P)$, оставляющим на месте его вершину A, можно совместить любые два ребра AB и AC, выходящие из этой вершины. Отсюда следует, что ребра, выходящие из вершины A, образуют равные углы с отрезком AO, соединяющим вершину A с центром многогранника, а все вершины многогранника, имеющие общее ребро с вершиной A, лежат в гиперплоскости π с уравнением $(\mathbf{r}-\mathbf{r}_B,\mathbf{r}_A)=0$, где центр O является началом координат, а B – некоторая вершина, соединенная с A ребром.

Задача. Доказать, что пересечение $st_A(P)=\pi\cap P$ – правильный $(n-1)$-мерный многогранник.

Многогранник $st_A(P)$ называется *звездой* многогранника P с вершиной A. Очевидно, что все звезды многогранника P эквивалентны и могут быть переведены преобразованием из группы $G(P)$ одна в другую.

Символ $(a_1, a_2,\ldots,a_{n-1})$ правильного многогранника P ([3], с. 496) представляет собой набор целых чисел, где a_1 – число сторон в двумерной грани многогранника

P, а набор $(a_2,\ldots,a_{n-1})$ – символ звезды $st_A(P)$. Правильный многогранник определяется своим символом с точностью до подобия.

Задачи.

1) Выписать символы трехмерных правильных многогранников.

2) Показать, что символы n-мерных симплекса, куба и кокуба имеют соответственно вид (3, …, 3), (4, 3, …, 3) и (3, …, 3, 4).

При $n>4$ других правильных многогранников, кроме симплекса, куба и кокуба, нет ([3], с. 498), но в размерности 4 имеются еще три многогранника с символами (3, 4, 3), (3, 3, 5) и (5, 3, 3). Ниже будет рассмотрен многогранник (3, 4, 3). У этого многогранника 24 вершины и 24 грани, одно из его названий – 24-ячейка. Двойственный к (3, 4, 3) многогранник совпадает с (3, 4, 3) с точностью до подобия. У многогранника (3, 3, 5) 600 граней и 120 вершин, а у двойственного многогранника (5, 3, 3) – 120 граней и 600 вершин ([5], с. 574).

24-ЯЧЕЙКА (МНОГОГРАННИК (3, 4, 3))

Стандартный правильный многогранник с символом (3, 4, 3) представляет собой выпуклую оболочку объединения стандартного куба $-1\le x^i\le 1$, $i=1, 2, 3, 4$, и кокуба, заданного неравенством $|x^1|+|x^2|+|x^3|+|x^4|\le 2$, или выпуклую оболочку 24 точек: 16-ти вершин куба (±1, ±1, ±1, ±1) и 8-ми вершин кокуба (±2, 0, 0, 0), (0, ±2, 0, 0), (0, 0, ±2, 0), (0, 0, 0, ±2). Покажем, что многогранник P, представляющий собой выпуклую оболочку вышеуказанных 24 точек, является правильным и имеет символ (3, 4, 3).

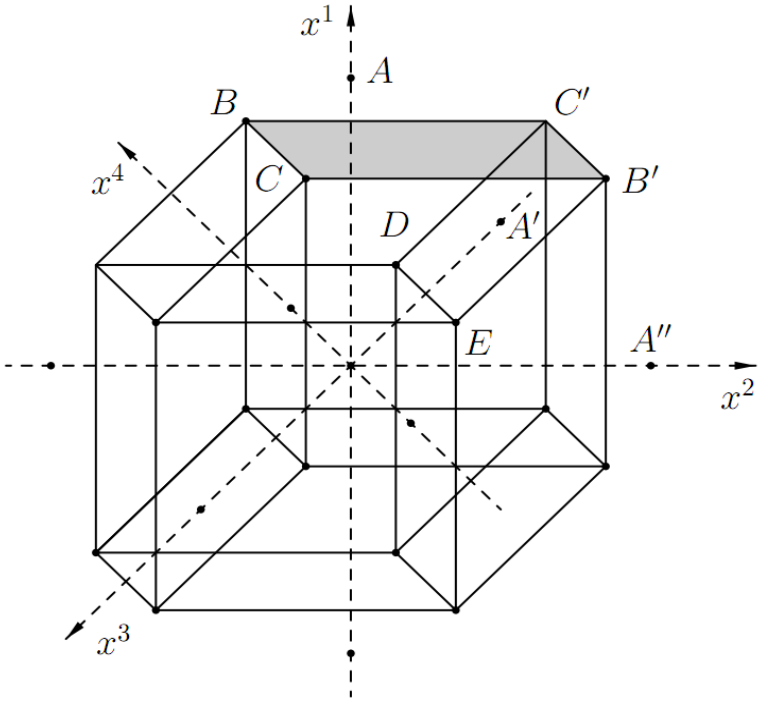


Рис. 16.

Рассмотрим вершину кокуба $A(2, 0, 0, 0)$. Ближайшими к ней вершинами многогранника P являются вершины грани $x^1=1$ куба, которая, таким образом, является звездой вершины A. Грань, содержащая вершину A (3-грань), содержит кроме нее еще некоторую 2-грань звезды, например, 2-грань $BCB'C'$ ($x^1=1$, $x^3=-1$), выделенную на рис. 16. Легко проверить, что 3-плоскость, содержащая точку A и квадрат $BCB'C'$, содержит и вершину кокуба A' с координатами (0, 0, −2, 0) и что многогранник $ABCB'C'A'$ – октаэдр. Поэтому флаги при вершине A (и при всех вершинах кокуба) – это октаэдры с их гранями. Преобразования из общей для куба и кокуба группы переводят флаги при вершинах кокуба снова в флаги при вершинах кокуба. Рассмотрим теперь ребро AD, где D – вершина куба с координатами (1, 1, 1, 1), и 3-плоскость π, проходящую через середину (3/2, 1/2, 1/2, 1/2) ребра AD перпендикулярно AD. Эта плоскость π имеет уравнение

$$-(x^1-3/2)+(x^2-1/2)+(x^3-1/2)+(x^4-1/2) \Leftrightarrow -x^1+x^2+x^3+x^4=0. \qquad (23)$$

Легко проверить, что многогранник P при симметрии σ относительно π переходит в себя. А именно, симметричными являются следующие 8 пар его вершин, в которых одна вершина является вершиной кокуба:

$$(2, 0, 0, 0) \leftrightarrow (1, 1, 1, 1), \qquad (-2, 0, 0, 0) \leftrightarrow (-1, -1, -1, -1),$$

$$(0, 2, 0, 0) \leftrightarrow (1, 1, -1, -1), \quad (0, -2, 0, 0) \leftrightarrow (-1, -1, 1, 1),$$
$$(0, 0, 2, 0) \leftrightarrow (1, -1, 1, -1), \quad (0, 0, -2, 0) \leftrightarrow (-1, 1, -1, 1),$$
$$(0, 0, 0, 2) \leftrightarrow (1, -1, -1, 1), \quad (0, 0, 0, -2) \leftrightarrow (-1, 1, 1, -1),$$

симметричны две вершины куба $(1, -1, -1, -1) \leftrightarrow (-1, 1, 1, 1)$, а остальные 6 вершин куба (1, 1, 1, −1), (1, 1, −1, 1), (1, −1, 1, 1), (−1, 1, −1, −1), (−1, −1, 1, −1) и (−1, −1, −1, 1) лежат в плоскости π. Таким образом, группа преобразований, порождаемая преобразованиями из группы куба и преобразованием σ, транзитивно действует на множестве флагов многогранника P, и многогранник P имеет символ (3, 4, 3).

Задача. Показать, что числа ребер, 2-граней и 3-граней многогранника (3, 4, 3) равны 96, 96 и 24 соответственно.

Число элементов в группе правильного многогранника можно вычислить как произведение числа его вершин на число элементов в группе звезды. Отсюда следует, что группа многогранника (3, 4, 3) состоит из 24×48=1152 элементов.

Задача. Вычислить это же число 1152 как число флагов многогранника (3, 4, 3).

Задача. Две 3-грани *ABCB'C'A'* и *ADEB'C'A''* на рис. 16 имеют общую 2-грань *AB'C'*. Показать, что векторы с началом в центре многогранника и концами в центрах 3-граней *ABCB'C'A'* и *ADEB'C'A''* имеют координаты {1,0,−1,0} и {1,1,0,0} соответственно, что позволяет вычислить двугранный угол при 2-грани *AB'C'*, равный 120^{o}.

В размерности три можно рассмотреть многогранник с 14 вершинами (±1, ±1, ±1), (±2, 0, 0), (0, ±2, 0), (0, 0, ±2), определенный аналогично (3, 4, 3) как выпуклая оболочка куба $-1 \le x^i \le 1$, i=1, 2, 3, и октаэдра $|x^1|+|x^2|+|x^3| \le 2$. Этот многогранник называется *ромбододекаэдром* ([5, с. 578]).

Задача. Проверить, что ромбододекаэдр имеет 12 граней, каждая из которых является ромбом с острым углом arccos(1/3), в вершинах куба сходится по три грани, а в вершинах октаэдра сходится по четыре грани ромбододекаэдра, ребра куба являются диагоналями граней ромбододекаэдра (см. Рис. 17, на котором изображена часть граней ромбододекаэдра).

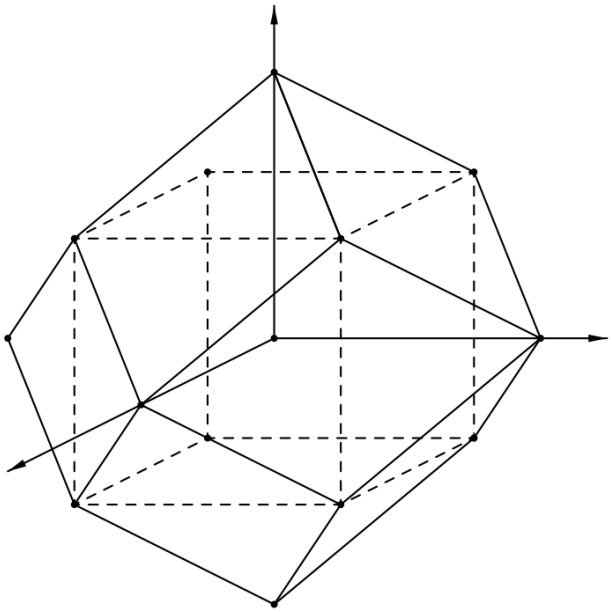


Рис. 17.

Задача. Проверить, что сечения стандартной 24-ячейки координатными 3-плоскостями являются ромбододекаэдрами.

Задача. Вычислив координаты пересечения ребер стандартной 24-ячейки с 3-плоскостью (23), показать, что сечение 24-ячейки 3-плоскостью (23) является ромбододекаэдром.

Задача. Показать, что выпуклая оболочка 24 точек с координатами

$$(\pm1, \pm1, 0, 0),\quad (\pm1, 0, \pm1, 0),\quad (\pm1, 0, 0, \pm1),$$
$$(0, \pm1, \pm1, 0),\quad (0, \pm1, 0, \pm1),\quad (0, 0, \pm1, \pm1)$$

является 24-ячейкой. *Указание:* рассмотреть систему координат, направляющими векторами координатных осей которой являются векторы

$$\mathbf{e}_1=\{1, 1, 0, 0\},\ \mathbf{e}_2=\{1, -1, 0, 0\},\ \mathbf{e}_3=\{0, 0, 1, 1\},\ \mathbf{e}_4=\{0, 0, 1, -1\}.$$

СФЕРЫ В ЕВКЛИДОВЫХ ПРОСТРАНСТВАХ

Сфера $S(M_0, a)$ радиуса a с центром в точке M_0 в E_n задается уравнением (15). В прямоугольной системе координат уравнение (15) принимает вид

$$(x^1-x_0{}^1)^2+(x^2-x_0{}^2)^2+\ldots+(x^n-x_0{}^n)^2=a^2.$$

Подмножество $\mathbf{S}^n$ в $\mathbf{R}^{n+1}$, заданное уравнением

$$(x^1)^2+(x^2)^2+\ldots+(x^{n+1})^2=1,$$

называется *стандартной n-мерной сферой*.

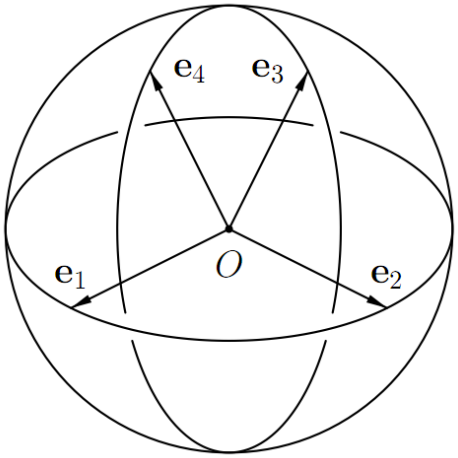


Рис. 18.

На рис. 18 изображена трехмерная сфера. Плоскости Ox^1x^2 и Ox^3x^4 имеют только одну общую точку — начало координат, большие окружности сферы, лежащие в этих плоскостях, не пересекаются.

Задача. Векторное пространство $\mathbf{R}^{2n}$ можно снабдить структурой комплексного пространства $\mathbf{C}^n$ (например, полагая $z^j=x^j+\sqrt{-1}\,x^{j+n}$). Использовав то, что n-мерное комплексное векторное пространство является объединением одномерных векторных пространств, имеющих нуль единственным общим вектором, показать, что нечетномерная сфера $\mathbf{S}^{2n-1}$ (в частности, $\mathbf{S}^3$) является объединением непересекающихся окружностей.

В уравнении (15) размерность сферы не присутствует, поэтому многие формулы для n-мерной сферы не отличаются от аналогичных формул для двумерной сферы или окружности. Например, уравнение касательной (гипер)-плоскости к сфере (15) в ее точке $M_1(\mathbf{r}_1)$ имеет вид

$$(\mathbf{r}-\mathbf{r}_1,\, \mathbf{r}_1-\mathbf{r}_0)=0 \;\Leftrightarrow\; (\mathbf{r}-\mathbf{r}_0,\, \mathbf{r}_1-\mathbf{r}_0)=a^2.$$

Задача. Вычислить радиусы сфер, описанных около рассмотренных выше стандартных правильных многогранников.

Задача. Показать, что для всякого правильного многогранника существует сфера с центром в центре многогранника, касающаяся всех его граней в центре этих граней. Эта сфера называется сферой, вписанной в правильный многогранник. Использовав формулу (9), вычислить радиусы сфер, вписанных в рассмотренные выше стандартные правильные многогранники.

СФЕРА S^3 КАК ДВА ПОЛНОТОРИЯ С ОБЩЕЙ ГРАНИЦЕЙ

Топологически, то есть с точностью до взаимно однозначного и взаимно непрерывного соответствия, n-мерная сфера $\mathbf{S}^n$ эквивалентна границе (n+1)-мерного куба и границе (n+1)-мерного симплекса (и вообще границе любого выпуклого многогранника в E_{n+1}). Взаимно однозначное и взаимно непрерывное соответствие называется гомеоморфизмом. Гомеоморфизм между кубом или симплексом и описанной сферой можно осуществить, устанавливая соответствие между точками, лежащими на одном луче, выходящем из общего центра. На рис. 19 такие гомеоморфизмы изображены для случая размерности n+1=2.

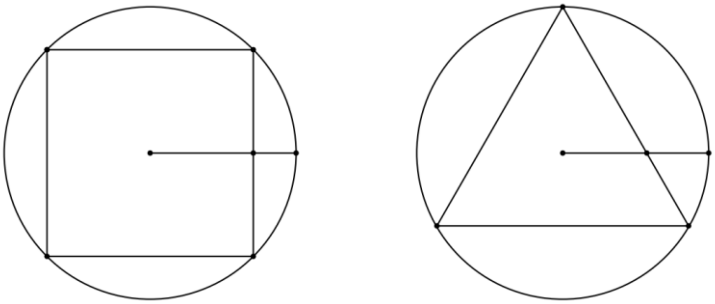


Рис. 19.

Стандартную трехмерную сферу $\mathbf{S}^3$, заданную уравнением

$$(x^1)^2+(x^2)^2+(x^3)^2+(x^4)^2=1 \tag{24}$$

в пространстве $\mathbf{R}^4$, можно представить как объединение двух подмножеств

$$P=\{(x^1, x^2, x^3, x^4)\in\mathbf{S}^3 \mid (x^1)^2+(x^2)^2\le 1/2\}, \tag{25}$$

$$Q=\{(x^1, x^2, x^3, x^4)\in\mathbf{S}^3 \mid (x^3)^2+(x^4)^2\le 1/2\}, \tag{26}$$

имеющих общую границу, определяемую системой уравнений

$$(x^1)^2+(x^2)^2=1/2, \quad (x^3)^2+(x^4)^2=1/2. \tag{27}$$

Системой уравнений (27) в $\mathbf{R}^4$ определяется двумерная поверхность T, называемая тором. Если пространство $\mathbf{R}^4$ рассматривать как произведение двух двумерных пространств $\mathbf{R}^2\times\mathbf{R}^2$, то тор T представляет собой произведение $\mathbf{S}^1\times\mathbf{S}^1$ двух окружностей. Топологически тор T эквивалентен поверхности бублика, то есть поверхности в трехмерном пространстве, получаемой вращением окружности вокруг прямой, лежащей с окружностью в одной плоскости и не пересекающей окружность. Тор T расположен на трехмерной сфере $\mathbf{S}^3$ и делит сферу $\mathbf{S}^3$ на два

подмножества *P* и *Q*, каждое из которых топологически представляет собой полноторие (заполненный тор).

Понятно, что в любом евклидовом пространстве E_4 в произвольной прямоугольной системе координат сфера единичного радиуса с центром в начале координат будет иметь уравнение (24) и будет представляться в виде объединения двух полноторий (25) и (26). Таким образом, представить трехмерную сферу в виде объединения двух полноторий с общей границей можно многими способами. Поскольку трехмерная сфера гомеоморфна границе любого четырехмерного куба, то указанные полнотория (точнее, их образы при гомеоморфизме) можно отыскать на границе куба.

Задача. Убедиться, что объединения (последовательно идущих) трехмерных граней 4-куба на рисунке 2, определяемых соответственно уравнениями $x^1=1$, $x^2=1$, $x^1=0$, $x^2=0$ и $x^4=0$, $x^3=0$, $x^4=1$, $x^3=1$, представляют собой пару полноторий с общей границей. На развертке, изображенной на рисунке 6, четыре выстроенных в линию куба принадлежат одному полноторию, а четыре, окружающие центральный куб $x^2=1$, принадлежат второму.

Задача. Найти пару взаимно дополнительных полноторий на рис. 8.

Другие темы из области многомерной геометрии, а также задачи, можно найти в приведенной ниже литературе.

СПИСОК ЛИТЕРАТУРЫ

1. *Базылев В.Т., Дуничев К.И., Иваницкая В.П.* Геометрия I. М.: Просвещение, 1974. 352 с.

2. *Базылев В.Т., Дуничев К.И.* Геометрия II. М.: Просвещение, 1975. 367.

3. *Берже М.* Геометрия. Т. 1. М.: Мир, 1984. 560 с.

4. *Берже М.* Геометрия. Т. 2. М.: Мир, 1984. 368 с.

5. *Кокстер Г.С.М.* Введение в геометрию. М.: Наука, 1966. 648 с.

6. Открытая Поволжская математическая олимпиада студентов, приуроченная ко дню рождения Н.И. Лобачевского. URL: https://lobachevolymp.kpfu.ru

7. *Проскуряков И.В.* Сборник задач по линейной алгебре. М.: Наука, 1967. 384 с.

8. *Розенфельд Б.А.* Многомерные пространства. М.: Наука, 1966. 648 с.

9. *Садовничий В.А., Григорьян А.А., Конягин С.В.* Задачи студенческих математических олимпиад. М.: Изд-во Моск. ун-та, 1987. 310 с.

10. Сборник задач по геометрии. Под ред. В.Т. Базылева. М.: Просвещение, 1980. 240 с.

11. *Шурыгин В.В., Шурыгин В.В. (мл.)* Аналитическая геометрия I. Учебное пособие к курсу «Аналитическая геометрия». Часть I. Аналитическая геометрия плоскости. Казань: КФУ, 2018. 154 с.

12. *Шурыгин В.В., Шурыгин В.В. (мл.)* Комбинирование методов евклидовой, аффинной и проективной геометрий при решении геометрических задач // Электронные библиотеки. 2021. Т. 24. № 3. С. 563-580.

13. *Шурыгин В.В., Шурыгин В.В. (мл.)* Аналитическая геометрия III. Учебное пособие к курсу «Аналитическая геометрия». Часть III. Многомерные пространства. Гиперповерхности второго порядка. Казань: КФУ, 2014. 160 с.

14. Энциклопедия элементарной математики. Кн. V — Геометрия. М.: Физматгиз, 1966. 624 с.

MULTIDIMENSIONAL GEOMETRY IN ELECTIVE COURSES FOR SECONDARY SCHOOL AND FIRST YEAR UNIVERSITY STUDENTS

V. V. Shurygin[1][0000-0002-4325-214X], V. V. Shurygin, jr.[2][0000-0001-9771-1447]

Kazan Federal University, Kazan

[1]Vadim.Shurygin@kpfu.ru, [2]1Vadim.Shurygin@kpfu.ru

Abstract

In the paper, we develop some approaches to teaching multidimensional geometry in elective courses the aim of which is to help students to develop multidimensional geometric intuition. Special attention is given to the use of transformation groups in the study of geometry of regular polyhedrons.

Keywords*: Four-dimensional cube, four-dimensional geometry, four-dimensional regular polyhedron, multidimensional geometry, three-dimensional sphere, 24-cell.*

REFERENCES

1. *Bazylev V.T., Dunichev K.I., Ivanitskaya V.P.* Geometriya I. M.: Prosveshchenie, 1974. 352 p.

2. *Bazylev V.T., Dunichev K.I.* Geometriya II. M.: Prosveshchenie, 1975. 367 p.

3. *Berger M.* Geometry. V. 1. Springer, 1987. 432 p. (Russian translation)

4. *Berger M.* Geometry. V. 2. Springer, 1987. 406 p. (Russian translation)

5. *Coxeter H.S.M.* Introduction to Geometry. NY.: John Wiley & Sons, 1989. 496 p. (Russian translation)

6. Lobachevskii Mathematical Competition for University Students. URL: https://lobachevolymp.kpfu.ru

7. *Proskuryakov I.V.* Sbornik zadach po lineynoy algebre. M.: Nauka, 1967. 384 p.

8. *Rosenfeld B.A.* Mnogomernye prostranstva. M.: Nauka, 1966. 648 p.

9. *Sadovnichii V.A., Grigoryan A.A., Konyagin S.V.* Zadachi studencheskikh matematicheskikh olimpiad. M.: Izd-vo Mosk. un-ta, 1987. 310 p.

10. Sbornik zadach po geometrii. Pod red. V.T. Bazyleva. M.: Prosveshchenie, 1980. 240 p.

11. *Shurygin V.V., Shurygin V.V., jr.* Analiticheskaya geometriya I. Uchebnoe posobie k kursu «Analiticheskaya geometriya». Chast’ I. Analiticheskaya geometriya ploskosti. Kazan: KFU, 2018. 154 p.

12. *Shurygin V.V., Shurygin V.V., jr.* Combining Methods of Euclidean, Affine, and Projective Geometries in Solving Geometric Problems // Russian Digital Libraries Journal. 2021. V. 24. № 3. P. 563-580.

13. *Shurygin V.V., Shurygin V.V., jr.* Analiticheskaya geometriya III. Uchebnoe posobie k kursu «Analiticheskaya geometriya». Chast’ III. Mnogomernye prostranstva. Giperpoverkhnosti vtorogo poryadka. Kazan: KFU, 2014. 160 p.

14. Entsiklopediya elementarnoy matematiki. Kn. V — Geometriya. M.: Fizmatgiz, 1966. 624 p.

СВЕДЕНИЯ ОБ АВТОРАХ

ШУРЫГИН Вадим Васильевич – профессор, кафедра геометрии, Казанский федеральный университет, Казань.

Vadim Vasilievich SHURYGIN – Professor of the Chair of Geometry, Kazan Federal University, Kazan.

email: Vadim.Shurygin@kpfu.ru

URL: https://kpfu.ru/Vadim.Shurygin

ORCID: 0000-0002-4325-214X.

ШУРЫГИН Вадим Вадимович – доцент, кафедра геометрии, Казанский федеральный университет, Казань.

Vadim Vadimovich SHURYGIN –Associate Professor of the Chair of Geometry, Kazan Federal University, Kazan.

email: 1Vadim.Shurygin@kpfu.ru

URL: https://kpfu.ru/1Vadim.Shurygin

ORCID: 0000-0001-9771-1447.

Материал поступил в редакцию 11 мая 2024 года