МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

Национальный исследовательский ядерный университет «МИФИ»

Факультет Кибернетики и Информационной безопасности

КАФЕДРА КИБЕРНЕТИКИ (№ 22)

Направление подготовки 09.04.04 Программная инженерия

Пояснительная записка

к научно-исследовательской работе студента на тему:

Сбор, анализ, разметка данных и создание среды обучения для виртуального актора на нейросетевой основе в контексте юмористических сюжетов

Группа _____ М20-504

Студент _____ Клычков М. Д.

Руководитель _____ Самсонович А. В.

Научный консультант _____

Оценка руководителя _____ Оценка комиссии _____

Члены комиссии

_____ _____

_____ _____

_____ _____

_____ _____

Москва 2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

Национальный исследовательский ядерный университет «МИФИ»

Факультет Кибернетики и Информационной безопасности

КАФЕДРА КИБЕРНЕТИКИ (№ 22)

Задание на НИР

Студенту гр. М20-504 Клычкову Матвею Дмитриевичу

ТЕМА НИР

Сбор, анализ, разметка данных и создание среды обучения для виртуального актора на нейросетевой основе в контексте юмористических сюжетов

ЗАДАНИЕ

№ п/п	Содержание работы	Форма отчетности	Срок исполнения	Отметка о выполнении Дата, подпись
1.	Аналитическая часть			
1.1.	Изучение и анализ классических математических моделей Земли и способов нахождения кратчайших расстояний на них	Пункт ПЗ		
1.2.	Изучить материалы описывающие движение воздушного судна	Пункт ПЗ		
1.3.	Изучение и анализ теории компьютерной графики применительно к построению полигональных сеток	Пункт ПЗ		
1.4.	Анализ методов компьютерной графики применительно к задачам построения заданных кривых в пространстве	Пункт ПЗ		
1.5.	Оформление расширенного содержания пояснительной записки (РСПЗ)	Текст РСПЗ	10.10.2021	
2.	Теоретическая часть			
2.1.	Аффинное преобразование			
2.2.	Описание математической модели Земли	Модели		
2.3.	Разработка алгоритмов построения полигональной сетки на основе таблицы высот земной поверхности	Алгоритмы		
2.4.	Разработка алгоритмов построения траектории движения воздушного судна	Алгоритмы		
2.5.	Модификация методов аппроксимации компьютерной графики для построения траектории движения воздушного судна	Алгоритмы		
3.	Инженерная часть			
3.1.	Проектирование методов и алгоритмов компьютерной графики, адаптированных под задачу построения полигональной модели Земли	Макеты		

3.2.	Проектирование классов и функций, для расчёта траектории движения воздушного судна	Макеты		
3.3.	Проектирование классов и функций, для нахождения пересечения кривых и полигональной сетки	Макеты		
3.4.	Проектирование юнит-тестов для проверки корректности работы статической библиотеки	Макеты		
3.5.	Результаты проектирования оформить с помощью UML диаграмм	UML диаграммы		
4.	Технологическая и практическая часть			
4.1.	Реализация статической библиотеки адаптированных методов и алгоритмов компьютерной графики, для построения полигональной сетки	Исполняемые файлы, исходный текст		
4.2.	Реализация статической библиотеки адаптированных методов и алгоритмов компьютерной графики, для построения траектории воздушного судна	Исполняемые файлы, исходный текст		
4.3.	Реализация статической библиотеки адаптированных методов и алгоритмов компьютерной графики, для нахождения пересечения траектории воздушного судна и земной поверхности	Исполняемые файлы, исходный текст		
4.4.	Тестирование статической библиотеки адаптированных методов и алгоритмов компьютерной графики	Исполняемые файлы, исходные тексты тестов и тестовых примеров		
5.	Оформление пояснительной записки (ПЗ) и иллюстративного материала для доклада.	Текст ПЗ, презентация	01.01.2022	

ЛИТЕРАТУРА

- Никулин Е. А. Компьютерная геометрия и алгоритмы машинной графики. — СПб: БХВ-2. Петербург, 2003
- Эдвард Энджел. Интерактивная компьютерная графика. Вводный курс на базе OpenGL = Interactive Computer Graphics. A Top-Down Approach with Open GL. — 2-е изд. — : «Вильямс», 2001
- Ефремов А.В., Захарченко В.Ф., Овчаренко В.Н., Суханов В.Л. Динамика полета: учебник для студентов высших учебных заведений – М. : Машиностроение, 2011
- Bjarne Stroustrup The C++ Programming Language Special Edition. - М.: Издательство «БИНОМ» 2012
- Andrew Koenig, Barbara E. Moo Accelerated C++: Practical Programming by Example.

Дата выдачи задания:	Руководитель	_____	Клычков М. Д.
10.10.2021	Студент	_____	Самсонович А. В.

Реферат

Пояснительная записка содержит 46 страниц, 19 рисунков, – 32 источников литературы.

Ключевые слова: UNITY3D, EBICA, СОЦИАЛЬНО-ЭМОЦИОНАЛЬНЫЙ ИНТЕЛЛЕКТ, ВИРТУАЛЬНЫЙ АКТОР, МАШИННОЕ ОБУЧЕНИЕ, РЕККУРЕНТНЫЕ СЕТИ

Объектом исследования являются экспертные системы.

Предмет исследования - модель когнитивной архитектуры для создания Виртуального Актора.

Целью данной научно-исследовательской работы является создание прототипа Виртуального Актора, модель интеллекта которого основана на когнитивной архитектуре eBICA. Существует два глобальных подхода к созданию социально-эмоционального интеллекта. Один основанный на нейросетях, другой на когнитивных архитектурах. В ходе работы над НИР был разработан и протестирован с участием испытуемых прототип Виртуального Актора, обладающий социально-эмоциональным интеллектом и помещенный в виртуальное окружение, которое создано при помощи графического движка Unity3d, была реплизована сиетема для анализа речи с выявлением эмоций соответствующим речевым признакам, так же были в дополнении к вышеуказанной системе, были спроектированы и реализованы методы воздействия на виртуального агента, основыющиеся на семантичекой составляющей речевого контекста. Данная работа является актуальной поскольку на данный момент эта область находится на начальных этапах развития и активной интеграции в различные индустрии. Созданная и протестированная модель интеллекта затем может быть интегрирована в другие проекты с Виртуальным Актором: виртуальный слушатель, виртуальный клоун, виртуальный танцор.

Содержание

Введение

В последнее время все большую и большую популярность набирают технологии предоставляющие возможность участвовать человеку в виртуальном мире либо технические средства, позволяющие представление виртуальную реальность в реальном мире. Виртуальные Акторы способны в будущем заменить докладчика на конференциях различного характера и представлениях.

Целью исследования является создание общей вычислительной модели механизмов, лежащих в основе человеческих эмоций. Осуществляется это путем создания Виртуального агента, подкрепленного когнитивной архитектурой и помещенного в виртуальное окружение. В данной парадигме человек (испытуемый, проводящий сеанс игры) может взаимодействовать с Виртуальным Актором, воплощенного в виде аватара в игре с трехмерной графикой, и оценивать его по различным параметрам. Такая модель может интерпретировать человеческое поведение и на основе экспериментов можно делать выводы о ее социальной приемлемости и точности имитирования человеческих эмоций.

В первом разделе обозначены когнитивные архитектуры, выявляются их преимущества и недостатки по сравнению с когнитивной архитектурой eBICA. Опрделяеются типы распознавания и синтеза речи. Ставятся цели и задачи научно-исследовательской работы.

Во втором производится анализ когнитивных архитектур и производится анализ методов распознавания человеской речи, а так же выявление в ней эмоциональных составляющих. Так же приводятся теоретические выкладки описания модели социально-эмоционального интеллекта.

В третьем разделе приводится подробное описание работы алгоритма, реализующего когнитивную модель социально-эмоционального интеллекта с учетом интегрированной системы распознавания речи. Строятся блок-схемы для кодовой реализации модели интеллекта.

В четвертом разделе приводятся реализация программного продукта представляющего собой когнитивно эмоциональный интелект, способный взаимодействовать на эмоциональной основе.

1. Исследование существующих когнитивных архитектур и анализ их недостатков

Проводится анализ по выявлению существующих недоработок прототипа. Выявляются недостатки и преимущества по сравнению с другими моделями искусственного интеллекта.

1.1 Изучение и анализ существующих когнитивных архитектур

Одной из наиболее известных когнитивных архитектур является архитектура, составленная Jonathan Gratch и Stacy Marsella, что описано в работе [1]. Цель их исследования - создать общую вычислительную модель механизмов, лежащих в основе человеческих эмоций, которая сможет всецело их описать. Хотя такая модель может давать объяснение человеческого поведения, они рассматривают разработку вычислительных моделей эмоций как ключевой объект исследований для искусственного интеллекта, который будет способствовать развитию большого количества вычислительных систем, которые моделируют, интерпретируют или влияют на человеческое поведение. На рисунке (Рис. 1.1) демонстрируется Когнитивно-мотивационно-эмоциональная система.

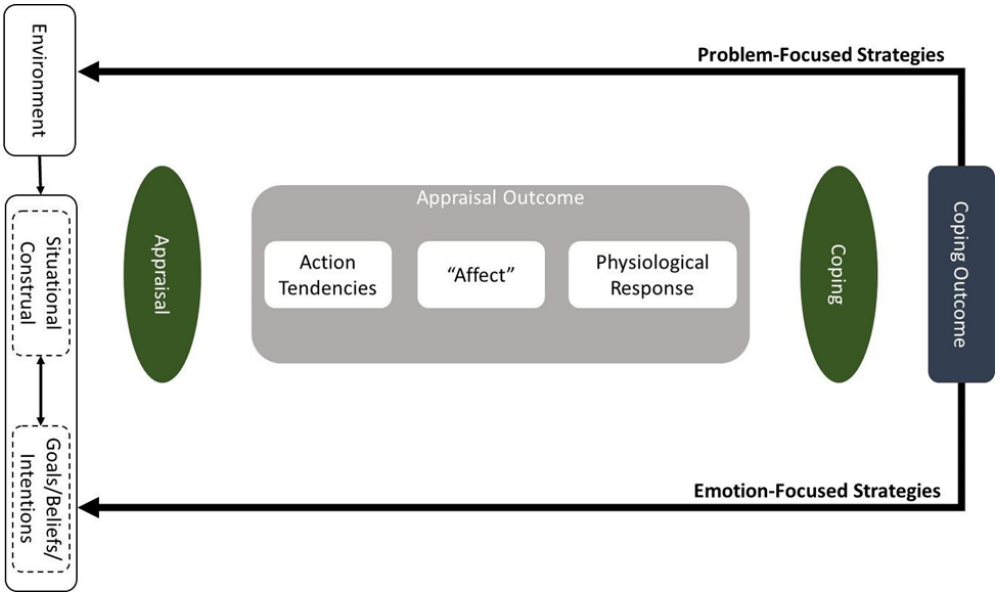

Рис. 1.1 – Когнитивно-мотивационно-эмоциональная система по материалам Smith and Lazarus.

Теория оценки служит концептуальной основой их работы, но эта психологическая теория недостаточно точна, чтобы служить спецификацией вычислительной модели. Для этого они

переделывают теорию с точки зрения методов и представлений искусственного интеллекта. Когнитивно-мотивационно-эмоциональная система Craig Smith и Richard Lazarus, показанная на рисунке 1, является представителем современных теорий оценки. Эмоция концептуализируется как двухступенчатая система контроля. Оценка характеризует отношения между человеком и его физическим и социальным окружением, называемые отношениями человека и окружающей среды, копирование поведения для восстановления или поддержания этих отношений. Поведение возникает в результате тесной связи познания, эмоций и реакций совладения: когнитивные процессы служат для построения индивидуальной интерпретации того, как внешние события соотносятся с его целями и желаниями (отношения человека и окружающей среды). Система использует эти характеристики для изменения отношений между человеком и окружающей средой, мотивируя действия, которые изменяют среду (копирование, ориентированное на проблему), или мотивируя изменения в интерпретации этих отношений (копирование, ориентированное на эмоции).

Модель PAD была разработана Albert Mehrabian и James A. Russell в 1974 году для описания и измерения эмоциональных состояний, как говорится в Работе [2]. В данной модели используются три числовых измерения для представления всех эмоций:

- A — arousal (возбуждение);
- P — pleasure (удовольствие);
- D — dominance (доминирование).

Модель PAD первоначально использовалась в теории психологии окружающей среды, а основное идеей модели было предположение о том, что физическая среда влияет на людей через их эмоциональное воздействие. На основе данной модели были построены физиологическая теория эмоций и теория эмоциональных эпизодов. Также модель использовалась для изучения невербального общения, в потребительском маркетинге и при создании анимированных персонажей, которые выражают эмоции.

В модели PAD используются трехмерные шкалы, которые в теории могут иметь любые числовые значения:

- шкала удовольствия-неудовольствия показывает, насколько приятно или, наоборот, неприятно человек себя чувствует по отношению к чему-то. Например, радость это — приятная эмоция; гнев и страх — неприятные эмоции;
- шкала возбуждения-неактивности измеряет, насколько человек чувствует возбуждение или его отсутствие. В данном случае оценивается именно возбуждение, а не интенсивность эмоций. Например, горе или депрессия характеризуются слабым возбуждением, но

сильной интенсивностью; а гнев или ярость имеют и высокую интенсивность, и высокое состояние возбуждения;

- шкала доминирования-покорности описывает чувство контроля и доминирования по сравнению со смирением и подчиненностью. Например, гнев — это доминирующая эмоция, а страх
- эмоция покорности, хотя обе они имеют неприятный характер.
- эмоция покорности, хотя обе они имеют неприятный характер.

Еще одна интересная когнитивная архитектура описана в статье трех научных деятелей Ron Sun, Nick Wilson, Michael Lynch. Статья имеет название: “Emotion: A Unified Mechanistic Interpretation from a Cognitive Architecture”. В этой статье рассматривается проект, который пытается интерпретировать эмоции - сложное и многогранное явление с механистической точки зрения, чему способствует существующая комплексная вычислительная когнитивная архитектура - CLARION. Эта когнитивная архитектура состоит из ряда подсистем: подсистем, ориентированных на действие, не ориентированных надействия, мотивационной и метакогнитивной подсистем. С этой точки зрения эмоции в первую очередь основаны на мотивации. Основываясь на этих функциональных возможностях, мы механистически (вычислительно) соединяем части вместе в рамках CLARION и фиксируем множество важных аспектов эмоций, как описано в литературе. На (Рис. 1.2) демонстрируются подсистемы когнитивной архитектуры CLARION.

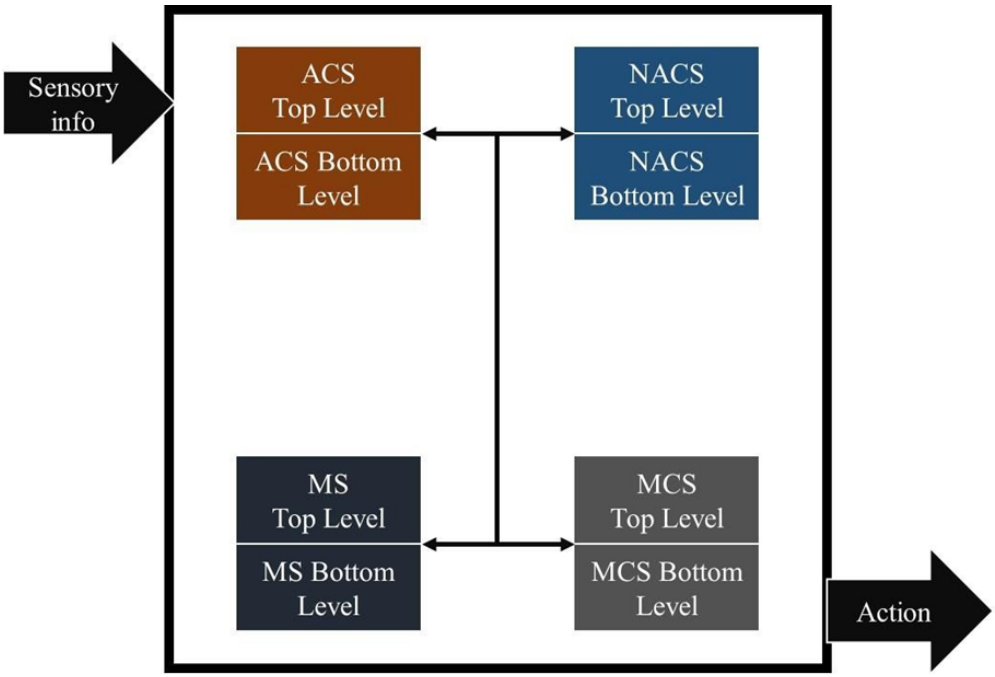

Рис. 1.2 – Подсистемы когнитивной архитектуры CLARION

Основные информационные потоки показаны стрелками. ACS означает подсистему, ори-

ентированную на действия. NACS означает подсистему, не ориентированную на действия. MC — это мотивационная подсистема. MCS означает метакогнитивную подсистему.

Получившая наибольшее распространение из всех формальных моделей представления эмоций является модель OCC (Ortony, Clore, & Collins), которая упоминается в работет [3], предложенная в 1988 году учеными Кембриджского университета. Иерархия содержит три ветви, а именно: эмоции, касающиеся последствий событий (например, радость и жалость), действия агентов (например, гордость и упрек) и аспекты объектов (например, любовь и ненависть). Кроме того, некоторые ветви объединяются в группу сложных эмоций, а именно эмоций относительно последствий событий, вызванных действиями агентов (например, благодарность и гнев). На рисунке (Рис. 1.3) демонстрируется оригинальная модель OOC.

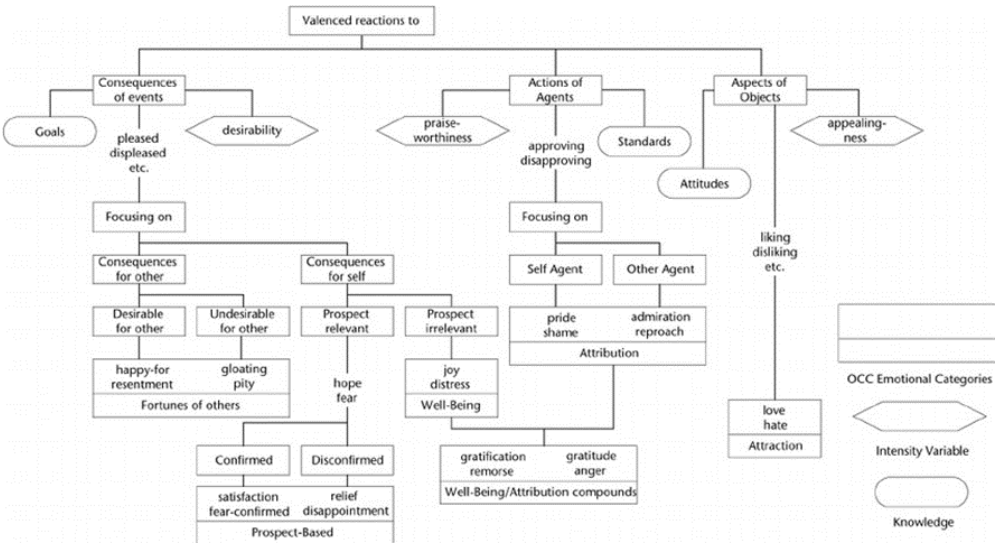

Рис. 1.3 – Оригинальная модель OCC

В основе правил динамики данной модели лежит реакция валентности (Valenced reaction). Под «валентностью» в психологии понимают внутреннюю привлекательность – «хорошую» (положительную валентность) или отвратительность – «плохую» (отрицательную валентность) события, объекта или ситуации. Эмоции формируются под воздействием трех основных факторов — последствий событий (Consequences of events), действий движущих сил (Actions of agents) и аспектов событий.

Рассмотрим левую «ветку» эмоциональной реакции. Последствия событий могут быть приносящими удовольствие (pleased) или доставляющими неудовольствие (displeased). Проведя предварительную оценку, человек фокусируется (Focusing on) на разделении последствий событий для себя (Consequences for self) и для других (Consequences for self), которые, в свою очередь могут оказаться для последних желательными (Desirable for other) или нежелательными (Undesirable for other). По поводу судеб других (Fortunes of others) в зависимости от личного

отношения — положительного или отрицательного — человек может испытывать следующие эмоции: радость за другого (Happy for), обида (Resentment), злорадство (Gloating) или жалость (Pity).

У этой модели есть свои ограничения, заключающиеся как в ее требовании упрощения человеческих эмоций, так и в ее сложном подходе к тому, как надлежит выводить эмоциональные состояния конечных пользователей посредством интерпретации поведения человека через знаки и сигналы, транслируемые людьми. Использование этой модели в ее оригинальном описании затруднено отсутствием математического аппарата, в следствии чего многие исследователи в своих Виртуальных Акторах используют упрощённые версии данной модели.

Также большой интерес представляет когнитивная архитектура, реализованная в физическом роботе, под названием - интегрированное когнитивное универсальное тело (iCub). Это когнитивная архитектура, дизайн которой основан на существующих знаниях в области робототехники, вычислений, нейробиологии и психологии, целью которой является копирование некоторых когнитивных процессов человека для их включения в человекоподобных роботов.

Эта архитектура реализована в человекоподобном роботе. Он был разработан для исследования сообществом когнитивных систем. Кроме того, он имеет лицензию «Стандартная общественная лицензия GNU (GPL)», так что любой человек может свободно использовать все наработки по данному проекту. Данная архитектура реализована в человекоподобном роботе, который имеет 53 степени свободы. По размеру он похож на ребенка трех-четырех лет и ребенка в возрасте 2,5 лет по когнитивным способностям. Кроме того, он может ползать и сидеть. Некоторые особенности, которые описаны в работе [4]

- Не хватает семантической памяти, чтобы помочь ему обобщать события;
- Невозможно сформировать привычки;
- Он учится путем подражания, проб и ошибок;
- Обнаруживает, распознает и отслеживает человеческое лицо, наблюдая за его действиями;
- Действия основаны на жестах рук, таких как встряхивание и манипулирование объектами,например, толкание, подъем и опускание.
- Действия, наблюдаемые роботом, изучаются и сохраняются в базе данных в процессе обучения.

На рисунке (Рис. 1.4) представлена схема работы iCub.

Вспомогательным инструментом при создании актора, наделенного социально- эмоциональным интеллектом, может являться - имитация моторного обучения (IML). IML начинает

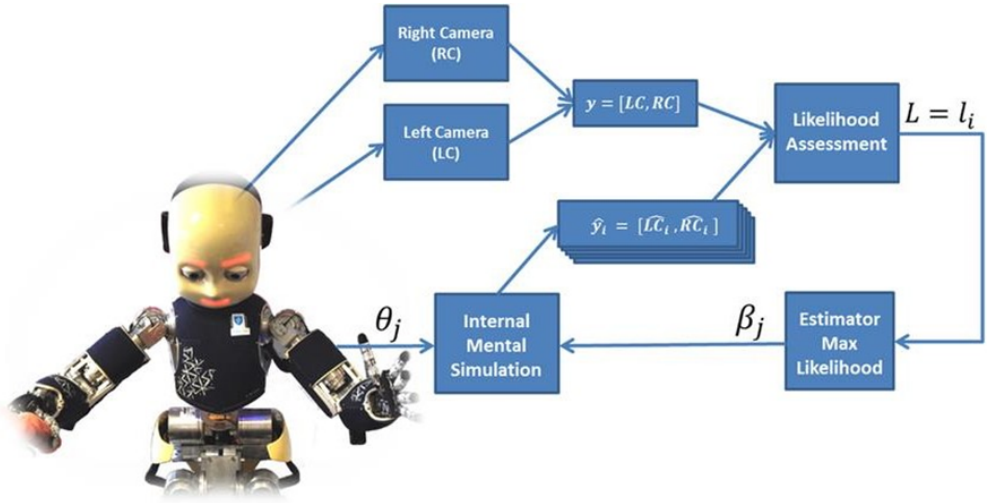

Рис. 1.4 – Схема работы iCub

наблюдать за другим актором, осуществляющим некоторую цепочку действий, затем категоризирует действия (определяет какую цель преследуют данные действия) одновременно отслеживая изменения точки обзора, окружающей среды, положения и типов объектов. Другими словами, когда Виртуальный агент неоднократно наблюдает за определенной новой последовательностью действий, каждый из знакомых элементов действия активирует соответствующее моторное представление через существующие ассоциации. Данное наблюдение формирует связи между элементарными моторными представлениями. Эта связь представлений составляет моторное обучение и улучшает имитационное движение. Способность моторной системы интегрировать разные части организма позволила бы создать обширный репертуар моторного поведения путем смешивания выходных сигналов разных частей организма, чтобы конечный результат отражал относительный и взвешенный вклад каждого в достижении цельной имитации движения. Поскольку невозможно воспроизвести функционирование мозга, были созданы модели, которые пытаются имитировать различные функции и поведение.

1.2 Изучение и анализ когнитивной архитектуры eBICA

Архитектура состоит из семи компонентов: интерфейсный буфер, рабочая, процедурная, семантическая и эпизодическая системы памяти, система ценностей и система когнитивных карт. Три основных строительных блока для этих компонентов — это ментальные состояния, схемы и семантические карты. Семантическая память — это коллекция определений схем. Буфер интерфейса заполняется схемами. Рабочая память включает активные психические состояния. Эпизодическая память хранит неактивные психические состояния, сгруппированные в эпизоды - предыдущее содержимое рабочей памяти. Следовательно, эпизодическая память состоит из структур, аналогичных тем, которые обнаруживаются в рабочей памяти, но которые

«заморожены» в долговременной памяти [5]. Процедурная память включает в себя примитивы. Система ценностей включает в себя шкалы, представляющие основные значения. Система когнитивных карт включает, в частности, семантические карты эмоциональных ценностей. Семантическая карта использует абстрактное метрическое пространство (семантическое пространство) для представления семантических отношений между ментальными состояниями, схемами и их 13 экземплярами, а также для присвоения значений их оценкам. На (Рис.1.5) демонстрируется семантическая карта [5].

Рис. 1.5 – Семантическая карта

Для когнитивного семантического отображения может использоваться слабое когнитивное семантическое картирование. Идея заключается в том, чтобы расположить представления на основе очень немногих основных семантических измерениях. Эти измерения могут возникать автоматически, если стратегия состоит в том, чтобы объединить синонимы и антонимы друг от друга. Карта, часть которой показана на рисунке 6 является результатом этого процесса. Эта карта не очень хорошо отделяет различные значения друг от друга: например, основные и сложные чувства. Однако она классифицирует значения в соответствии с их семантикой. Рисунок (Рис. 1.6) демонстрирует примеры простейших эмоциональных элементов в рамках eBICA [6].

(A) Схема имеет оценку в качестве своего атрибута. Это также атрибут головного узла. Значение этого атрибута - «доминантный», что означает, что действие воспринимается как проявление доминирования, или агент воспринимается как «доминантный по отношению ко мне» и т. Д. (B) Психическое состояние имеет оценку атрибут, который представляет собой эмоциональное состояние и самооценку агента в данный момент в данной ментальной перспективе.

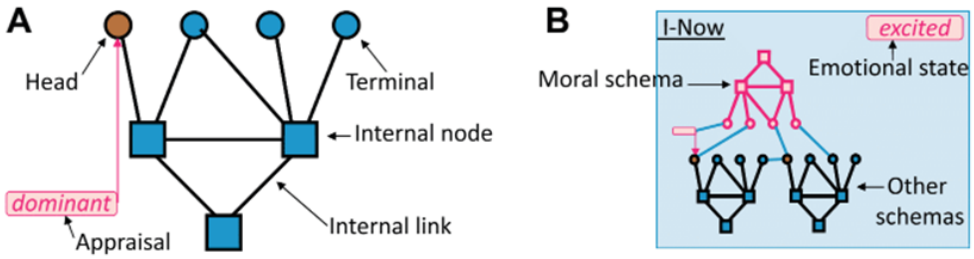

Рис. 1.6 – примеры эмоциональных элементов в рамках eBICA

Показанная ценность этой оценки «взволнована», что означает, что агент находится в возбужденном эмоциональном состоянии. Моральная схема, показанная в B, связывается с частью содержания психического состояния (включая определенный образец оценок) и представляет оценку выбранного образца, например, образец взаимодействий и взаимных оценок двух агентов, упомянутых в ментальном состоянии.

1.3 Нейронные сети и их типы

Нейро́нная сеть (также искусственная нейронная сеть, ИНС) — математическая модель, а также её программное или аппаратное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма. Это понятие возникло при изучении процессов, протекающих в мозге, и при попытке смоделировать эти процессы. Первой такой попыткой были нейронные сети У. Маккалока и У. Питтса. После разработки алгоритмов обучения получаемые модели стали использовать в практических целях: в задачах прогнозирования, для распознавания образов, в задачах управления и др.

Разделяют несколько основных разновидностей Нейронных сетей, согласно работе [7], а именно:

- Нейронные сети прямого распространения
- Сети радиально-базисных функций
- Нейронная сеть Хопфилда (Hopfield network, HN)
- Цепи Маркова (Markov chains, MC или discrete time Markov Chains, DTMC)
- Машина Больцмана (Boltzmann machine, BM)
- Ограниченная машина Больцмана (restricted Boltzmann machine, RBM)
- Автокодировщик (autoencoder, AE)
- Разреженный автокодировщик (sparse autoencoder, SAE)
- Вариационные автокодировщики (variational autoencoder, VAE)
- Шумоподавляющие автокодировщики (denoising autoencoder, DAE)

– Сеть типа «deep belief» (deep belief networks, DBN)
– Свёрточные нейронные сети (convolutional neural networks, CNN)
– Развёртывающие нейронные сети (deconvolutional networks, DN)

С точки зрения машинного обучения, нейронная сеть представляет собой частный случай методов распознавания образов, дискриминантного анализа. Рекуррентные нейронные сети (РНС, англ. Recurrent neural network; RNN) — вид нейронных сетей, где связи между элементами образуют направленную последовательность [8]. Благодаря этому появляется возможность обрабатывать серии событий во времени или последовательные пространственные цепочки. В отличие от многослойных перцептронов, рекуррентные сети могут использовать свою внутреннюю память для обработки последовательностей произвольной длины. Поэтому сети RNN применимы в таких задачах, где нечто целостное разбито на части, например: распознавание рукописного текста или распознавание речи. Было предложено много различных архитектурных решений для рекуррентных сетей от простых до сложных. В последнее время наибольшее распространение получили сеть с долговременной и кратковременной памятью (LSTM) и управляемый рекуррентный блок (GRU).

В последнее время наибольшую популярность для решения задач тематической классификации, применяемой для выделения семантического смысла текста, приобрели глубокие нейронные сети, так как они позволяют достичь наивысшей точности среди всех известных моделей машинного обучения. В частности, сверточные нейронные сети совершили прорыв в классификации изображений. В настоящее время они успешно справляются и с некоторыми задачами автоматической обработки текстов. Более того, как утверждается в некоторых исследованиях сверточные сети подходят для этого даже лучше рекуррентных нейронных сетей, которые чаще всего используются для анализа текстовых последовательностей. С другой стороны, использование сверточных сетей для классификации текстов мало исследовано. Поэтому исследование применения сверточных нейронных сетей для задачи классификации текстов в качестве альтернативы рекуррентным нейронным сетям представляет практический интерес, что описано в [9].

Для решения поставленной задачи требуется получить способ представления данных в виде, пригодном для обработки сверточной нейронной сетью. Например, в виде матрицы вещественных чисел. Наиболее распространенным является способ отображения каждого слова в многомерное векторное пространство. В рамках данной работы векторные представления слов строились на основе модели word2vec [10]. sa

1.4 Синтез и распознавание речи

Синтез речи - широком смысле — восстановление формы речевого сигнала по его параметрам; в узком смысле — формирование речевого сигнала по печатному тексту. Часть искусственного интеллекта. Синтезом речи — прежде всего называется всё, что связано с искусственным производством человеческой речи.

Первой технологией воспроизводящей синтез речи служила механическая говорящая машина Фабера (1845). Воздушный мех,приводимый в движение ножной педалью служил «лёгкими», а вытесняемый из меха воздух при помощи ряда клавиш направлялся в различные по объёму трубки - разные положения ”голосовой щели” и ”полости рта” (Рис. 1.7).

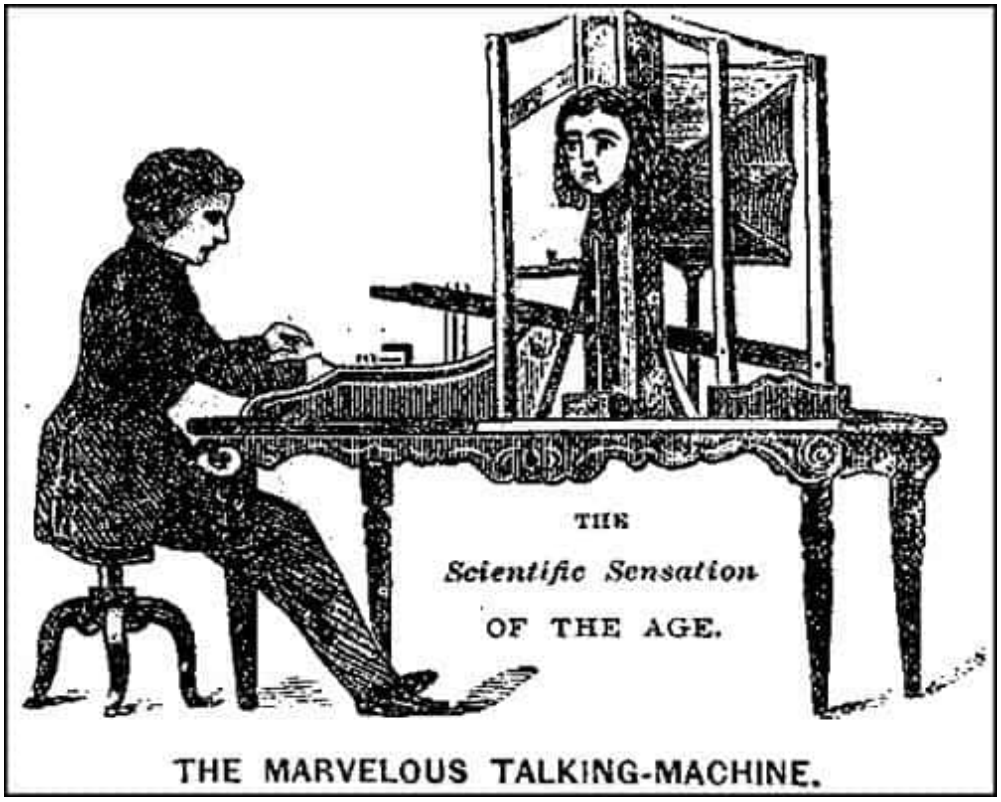

Рис. 1.7 – Машина Фабера 1845

Еще одной технологией служил так называемый ”вокодер”. Вокодер – синтезатор речи на основе произвольного сигнала с богатым спектром. Изначально вокодеры были разработаны в целях экономии частотных ресурсов радиолинии системы связи при передаче речевых сообщений. Вместо собственно речевого сигнала передают только значения его определённых параметров, которые на приёмной стороне управляют синтезатором речи (Рис. 1.8).

Основу синтезатора речи составляют три элемента:

- Генератор тонального сигнала для формирования гласных звуков;
- Генератор шума для формирования согласных;

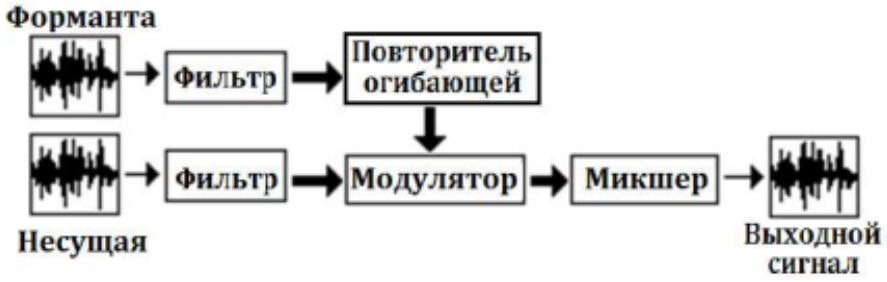

Рис. 1.8 – Вокодер

– Система формантных фильтров для воссоздания индивидуальных особенностей голоса.

Устная речь являет собой практически непрерывный звуковой поток, представляющий определенные сложности для сегментирования. В большинстве случаев для выделения фонемы необходимо знание языка. Выделяемость фонемы некоторым образом сопряжена со смыслом и со значением, хотя сама по себе она не является значащей единицей.

На текущий момент синтез речи не предствляет собой сложную задачу, так как в связи с темпами развития технологий, компьютеры позволяют хранить требуемую для синтеза информацию и соответственно синтезировать передаваемую фразу по посылаемой машиной строкой текста, а так же передавать нужную эмоциональную окраску в зависимости от семантики синтезируемого речевого фрагмента. Однако с темпами развития технологий, стали появляться задачи, когда машина реагирует не на скриптованное выполнение команд, а должна произвести речевой анализ получаемого текста и в ответ синтезировать ответ, для такого рода задач применяется распознавание речи.

Распознавание речи — автоматический процесс преобразования речевого сигнала в цифровую информацию. Обратной задачей является синтез речи. Распознавание речи — одна из самых интересных и сложных задач искусственного интеллекта. Здесь задействованы достижения весьма различных областей: от компьютерной лингвистики до цифровой обработки сигналов.

Основными задачами распознавания речи является:

1. Системы Interactive Voice Response (IVR) в колл-центрах:
   - Биометрическая идентификация
   - Автоматическая маршрутизация звонка
   - Аналитика речи, поиск пауз, тормозов в интерфейсе, аналитика эмоционального состояния
   - Сбор оценок операторов колл-центра

2. Персональные помощники:
    - Распознавание поисковых запросов
    - Распознавание команд управления

Существует ряд технологий предоставляющих возможность осуществлять распознавание речи:

- Алиса (Yandex)
- iOS Siri (Apple/Nuance)
- Google Assistant
- Amazon Alexa

В данный момент для распознавания речи используется так называемая акустическая модель. По сути это функция, принимающая на вход небольшой участок акустического сигнала (фрейм) и выдающая распределение вероятностей различных фонем на этом фрейме. Таким образом, акустическая модель дает возможность по звуку восстановить, что было произнесено — с той или иной степенью уверенности.

Еще один важный аспект акустики — вероятность перехода между различными фонемами. Из опыта известно, что одни сочетания фонем произносятся легко и встречаются часто, другие сложнее для произношения и на практике используются реже. Можно обобщить эту информацию и учитывать ее при оценке «правдоподобности» той или иной последовательности фонем.

Однако акустическая модель — это всего лишь одна из составляющих современных систем. Словари на текущий момент состоят из миллионов, а то и триллионов слов, многие из них совпадают по своему звучанию и могут даже совпадать. Вместе с тем, при наличии контекста роль акустики падает: невнятно произнесенные, зашумленные или неоднозначные слова можно восстановить «по смыслу». Для учета контекста используются так же вероятностные модели, такой тип языковых моделей называется n-gram language models.

1.5 Классификации и определение эмоций

Многообразие эмоций, их качественных и количественных проявлений исключают возможность простой и единой классификации. Каждая из характеристик эмоций может выступать в качестве самостоятельного критерия, основания для их классификации (таб. 1.1).

Таблица 1.1 – характеристики эмоции как основания для их классификации

Знак	Пололожительные, отрицательные, амбивалентные
Модальность	Радость, гнев, страх и др.
Влияение на поведение и деятельность	Осознаваемые, неосознаваемые
Предметность	Предметные, беспредментные
Степень произвольности	Произвольные, непроизвоольные
Происхождение и развлечения	Врожденные, приобретенные, первичные, вторичные
Уровень	Высшие, низшие
Длительность	Кратковременные, длительные
Интенсивность	Слабые, сильные

По знаку эмоциональные переживания можно разделить:

1. на положительные
2. отрицательные
3. амбивалентные

Основной функцией положительных эмоций является поддержание контакта с позитивным событием, поэтому им присуща реакция приближения к полезному, необходимому стимулу. Кроме того, по мнению П.В. Симонова, они побуждают нарушать достигнутое равновесие с окружающей средой и искать новую стимуляцию.

Для отрицательных эмоций характерной является реакция удаления, прерывания контакта с вредным или опасным стимулом. Считается, что они играют более важную биологическую роль, поскольку обеспечивают выживание индивида.

Амбивалентными эмоциями являются противоречивые эмоциональные переживания, связанные с двойственным отношением к чему-либо или кому-либо (одновременное принятие и отвержение).

1.6 Выводы

были рассмотрены основные бла бла бла

2. Описание моделей, отвечающих за генерацию поведения виртуального актора

В данном разделе приводится теоретическое описание модели.

2.1 Постановка задачи

В рамках научно-исследовательской работы был расширен подход решения поставленной задачи задачи, который выражается в использовании машинного обчучения. Данный подход используется в совокупности когнитивной архитектурой eBICA. eBICA – “emotional biologically inspired cognitive architecture” – “эмоциональная биологически вдохновленная когнитивная архитектура”.

В этой архитектуре эмоциональные элементы добавлены практически ко всем процессам за счет модификации основных строительных блоков архитектуры. Ключевым моментом этой когнитивной архитектуры являются оценки, которые связаны со схемами и психическими состояниями как их атрибуты, моральные схемы, которые контролируют модели оценок и представляют социальные эмоции, а также семантические пространства, которые дают значения этих оценок.

Как видно из (Рис. 2.1), архитектура представляет собой конгломерат компонентов: интерфейсный буфер, рабочая, процедурная, семантическая и эпизодическая системы памяти, система ценностей и система когнитивных карт [6]. Три основных строительных блока для этих компонентов - это ментальные состояния, схемы и семантические карты. Семантическая память - это коллекция определений схем. Буфер интерфейса заполняется схемами.

Рабочая память включает активные психические состояния. Эпизодическая память хранит неактивные психические состояния, сгруппированные в эпизоды - предыдущее содержимое рабочей памяти. Следовательно, эпизодическая память состоит из структур, аналогичных тем, которые обнаруживаются в рабочей памяти, но которые «заморожены» в долговременной памяти. Процедурная память включает в себя примитивы. Система ценностей включает в себя шкалы, представляющие основные значения. Система когнитивных карт включает, в частности, семантические карты эмоциональных ценностей. Семантическая карта использует абстрактное метрическое пространство (семантическое пространство) для представления семантических отношений между ментальными состояниями, схемами и их экземплярами, а также для присвоения значений их оценкам.

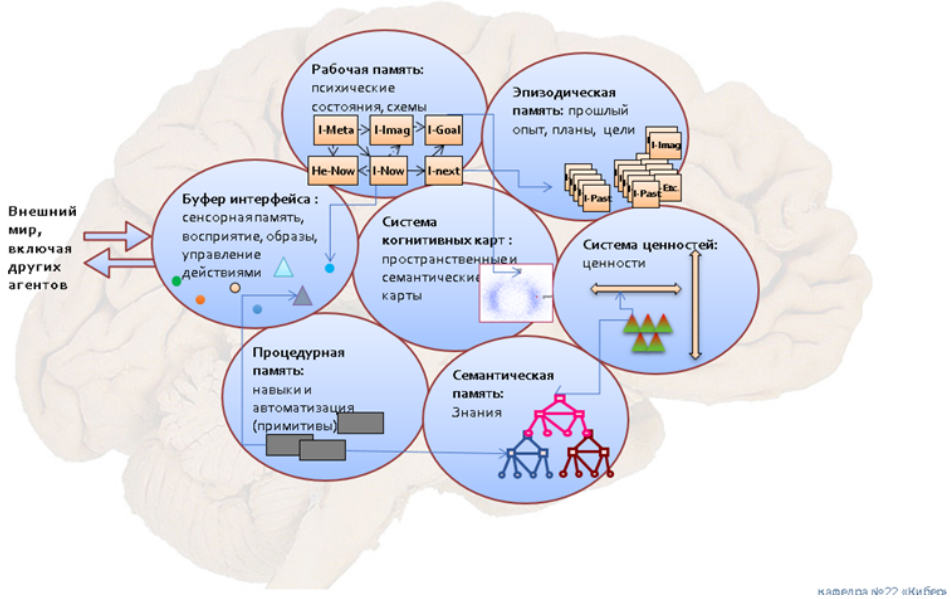

Рис. 2.1 – Структура когнитивной архитектуры eBICA

2.2 Описание работы модели актора

На момент начала выполнения работы уже была реализована система работы виртуальных агентов, и она состоит в том, что сперва считываются действия и объекты с заданными для них параметрами и значениями из Excel файла, а также инициализируются значения. Затем выбор действий происходит в следующем порядке: в радиусе вокруг оценок виртуального актора выбираются действия, которые попадают в этот радиус, а также проверяются различные условия, необходимые для выполнения действия. Если условия не выполняются, то действие не может быть выбрано. Затем, после того как в список добавлены все действия, которые могут быть выполнены, рассчитываются вероятности на основе оценок и рассчитанных констант. Также, если действие повторное, его вероятность несколько занижается. После расчета вероятностей выбирается действие, которое влияет на оценки виртуального актора.

Помимо этого происходит замена состояний объектов и виртуальных акторов. После этого происходит перерасчет оценок Appraisals и Feelings [6].

Основная модель eBICA определяет поведение виртуального актора исходя из следующих факторов:

- соматический;
- рациональный;
- когнитивный.

Нравственный фактор регулирует отношения первого актора со вторым на основе системы ценностей (представленной семантической картой) и моральных схем. Под когнитивным фак-

тором понимается учет соображений нравственности, этики и морали, общей системы ценностей, понятий о добре и зле, о собственном достоинстве, эмпатии, соображений эстетики, стремлений к простоте и элегантности, и т.д. Учет этих соображений возможен на основе когнитивных оценок (appraisals) всех релевантных агентов, событий, их возможных действий и последствий этих действий, фактов, свойств, отношений, и т.д. Возможен вариант модели, в которой ответное действие может выбираться лишь из двух вариантов: положительная реакция на действие человека и отрицательная. Данная версия модели весьма неплохо работает даже с таким ограничением. Но невозможно придерживаться данной парадигмы при увеличении количества возможных вариантов для взаимодействия между акторами. В данной модели необходимо учесть пересчет оценок Appraisals и Feelings. Для пересчета оценок Appraisals используется следующая формула 2.1:

$$Appraisals = (1 - r) * Appraisals + r * Action \tag{2.1}$$

где Appraisals - оценка, r - эмпирически вычисленная константа экспоненциального затухания, Action - оценка совершаемого действия на семантической карте.

Одновременно с Appraisals пересчитываются так называемые “чувства” Feelings согласно режиму работы моральной схемы. Аффективное пространство VAD – это трехмерное векторное пространство, точки которого соответствуют определенным эмоциональным состояниям, или аффектам, представленным триплетами значений (Valence, Arousal, Dominance). Существуют и сходные модели: PAD (Pleasure, Arousal, Dominance), EPA (Evaluation, Potency, Arousal) и другие. Здесь мы используем модель VAD. Соответственно, под «семантической картой» здесь часто понимается ее конкретная разновидность: аффективная карта (или когнитивная семантическая карта).

Шкалы имеют следующие значения:

- dominance – варьируется при значении от 0 (покорность) до +1 (доминантность) и описывает соответствующие чувства;
- valense – при значениях от -1 до 0 показывает уровень негатива или радости соответственно;
- arousal – значения от -1 до 1 показывают уровень возбуждения (заинтересованности), к примеру, гнев по уровню возбуждения сильнее раздражительности, но слабее ярости.

Оценки представлены в виде векторов на трехмерной семантической карте [5], 1.5. Моральная схема определяет общую установку на оценку поведения акторов, согласно их ролям и типу ситуации. Ее целью (как агента) является достижение и поддержание «нормального» по-

ложения дел, определенного набором Feelings. Вообще говоря, моральная схема состоит из двух частей: части, распознающей тип ситуации и осуществляющей привязку (binding), и части, реализующей динамику схемы. В случае парадигмы актора можно считать, что моральная схема одна, уже привязана, и потому первая часть ее не актуальна.

Субъективные оценки (Feelings) генерируются по определенным правилам на основании истории объективных оценок и состояний системы. Грубо говоря, Feelings – это субъективное представление о том, каким оцениваемый актор является «на самом деле», и, следовательно, какого поведения от него нужно ожидать и на какое место его нужно ставить своим поведением. Следовательно, выбор поведения актора должен осуществляться так, чтобы приблизить Appraisals к Feelings.

Значение Feelings определяет моральная схема, которая может работать в одном из трех режимов. Первый режим основывается на формуле 2.2:

$$Feelings = beta * Appraisals \tag{2.2}$$

где beta – эмпирически вычисленная константа.

В данном режиме схема говорит, что если актор ведет себя хорошо, то к нему нужно относиться как к хорошему, и т.д.

Цель данного процесса – распознать и классифицировать актора, выработать отношение к нему и приписать ему определенную роль во взаимоотношениях.

В данном режиме моральная схема работает пока разница между квадратами норм Feeling и Appraisals не станет меньше некоторого значения.

Суть второго режима заключается в том, что значение Feeling фиксировано и экстремально по абсолютной величине, т.е. находится на сфере, ограничивающей семантическую карту (предположим, что есть такая сфера). Направленность вектора Feeling может быть либо произвольной, определенной предысторией, либо дискретной – вдоль одной из осей.

Третий режим состоит в том, что значения Feelings меняются 2.3, подстраиваясь под текущие значения Appraisals (здесь r1 может быть отличным от r):

$$Feelings = (1 - r_1) * Feelings + r1 * (Appraisal - Feelings) \tag{2.3}$$

Соответственно значения Appraisals и Feelings как говорится в работе [**Samsonovich05**] пересчитываются после каждого действия первого актора, направленного на второго актора. Также пересчет оценок происходит после определения и совершения одним из акторов ответного или самостоятельного действия. В данном контексте под термином “самостоятельное дей-

ствие" имеется в виду действие, основанное лишь на текущем состоянии мира и значений векторов Appraisals и Feelings акторов, отобрежнные на (Рис. 2.2).

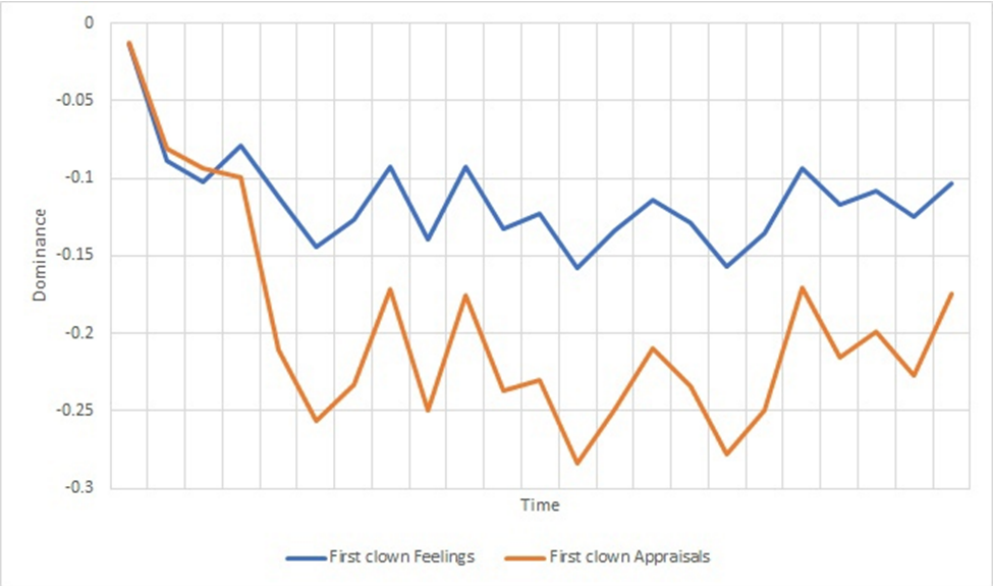

Рис. 2.2 – Корреляция значений Appraisals (оранжевая) и Feelings (синяя) для показателя доминантности на протяжении времени/действий с шагом в 5 секунд

Согласно (Рис. 2.2) мы видим, что работа модели сводится к выбору действия, которое будет максимально приближать Appraisals к Feelings и вектор соматического состояния к начальному положению.

2.3 Распознавание речи

Как было упомянуто ранее, для распознавания речи используются акустические модели, однако этого может быть недосточно, так как словари на текущий момент состоят из миллионов, а то и триллионов слов, многие из них совпадают по своему звучанию и могут даже совпадать и в написании и в зучании.

За основной параметр в использовании распозвавания речи при испоьзовании классической акустической модели - берется фонема, однако в слове она может находиться в трех различных состояниях: начало, середина и конец. фонемы являются позиционно-зависимыми и контекстно-зависимыми: формально «одна и та же» фонема звучит существенно по-разному в зависимости от того, в какой части слова она находится и с какими фонемами соседствует. Вместе с тем, простое перечисление всех возможных вариантов контекстно-зависимых фонем вернет очень большое число сочетаний, многие из которых никогда не встречаются в реальной жизни; чтобы сделать количество рассматриваемых акустических событий разумным, близкие контекстно-зависимые фонемы объединяются на ранних этапах тренировки и рассматриваются вместе.

Тренировка акустической модели — сложный и многоэтапный процесс. Для тренировки используются алгоритмы семейства Expectation-Maximization, такие, как алгоритм Баума-Велша. Суть алгоритмов такого рода — в чередовании двух шагов: на шаге Expectation имеющаяся модель используется для вычисления матожидания функции правдоподобия, на шаге Maximization параметры модели изменяются таким образом, чтобы максимизировать эту оценку (Рис. 2.3).

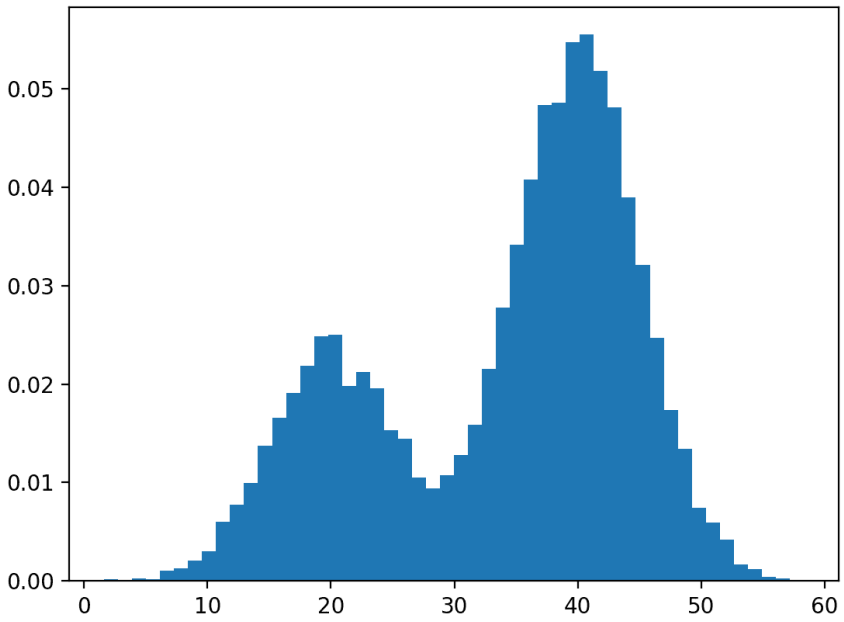

Рис. 2.3 – Гистограмма набора данных, построенная из двух разных гауссовских процессов

На ранних этапах тренировки используются простые акустические модели: на вход даются простые MFCC features (Рис. 2.4), фонемы рассматриваются вне контекстной зависимости, для моделирования вероятности эмиссии в HMM используется смесь гауссиан с диагональными матрицами ковариаций (Diagonal GMMs — Gaussian Mixture Models).

Результаты каждой предыдущей акустической модели являются стартовой точкой для тренировки более сложной модели, с более сложным входом, выходом или функцией распределения вероятности эмиссии. Существует множество способов улучшения акустической модели, однако наиболее значительный эффект имеет переход от GMM-модели к DNN (Deep Neural Network), что повышает качество распознавания практически в два раза. Нейронные сети лишены многих ограничений, характерных для гауссовых смесей, и обладают лучшей обобщающей способностью. Кроме того, акустические модели на нейронных сетях более устойчивы к шуму и обладают лучшим быстродействием.

Нейронная сеть для акустического моделирования тренируется в несколько этапов. Для

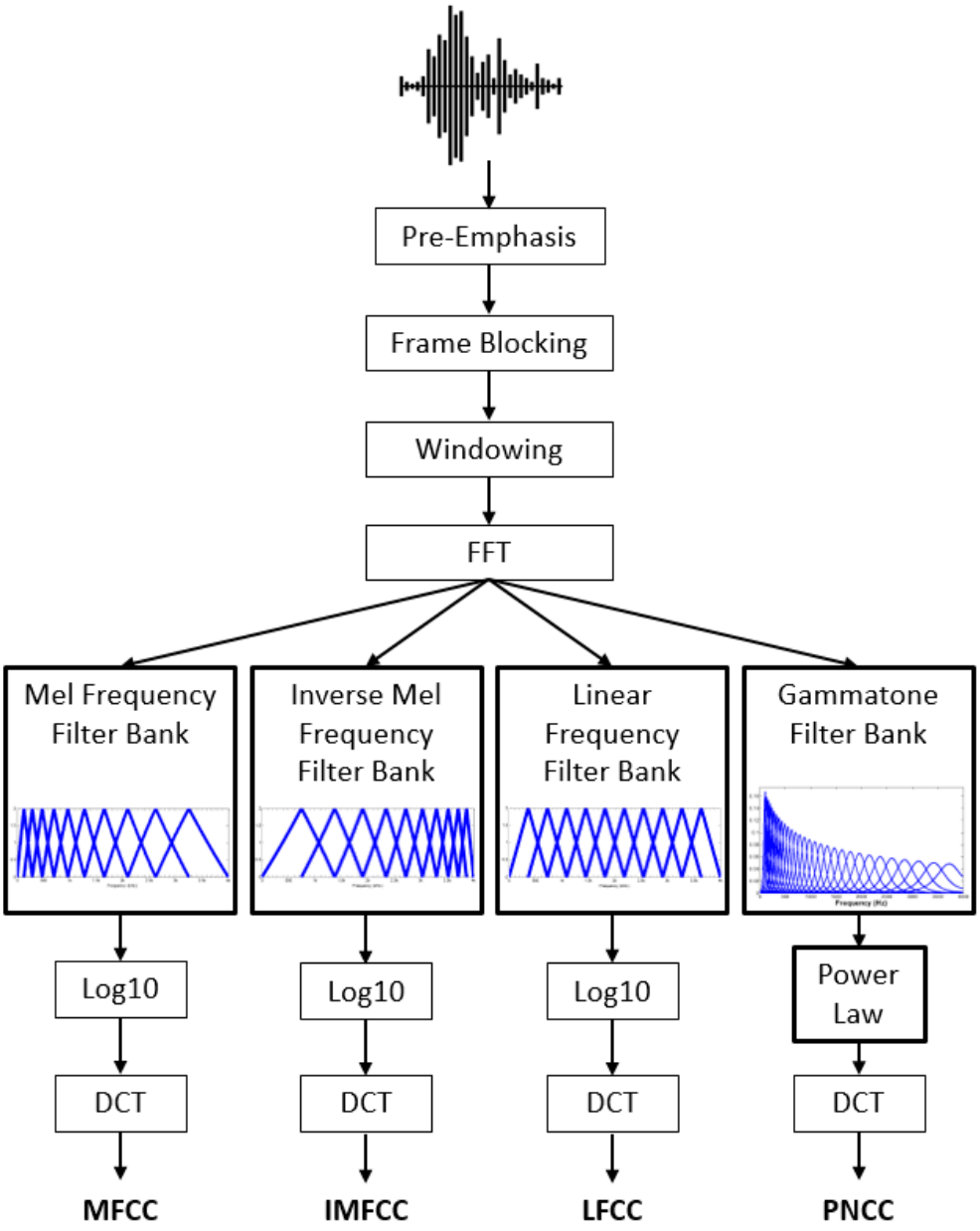

Рис. 2.4 – Четыре метода выделения признаков MFCC, IMFCC, LFCC и PNCC

инициализации нейросети используется стек из ограниченных машин Больцмана (Restricted Boltzmann Machines, RBM). RBM — это стохастическая нейросеть, которая тренируется без учителя. Хотя выученные ей веса нельзя напрямую использовать для различения между классами акустических событий, они детально отражают структуру речи. Можно относиться к RBM как к механизму извлечения признаков (feature extractor) — полученная генеративная модель оказывается отличной стартовой точкой для построения дискриминативной модели. Дискриминативная модель тренируется с использованием классического алгоритма обратного распространения ошибки, при этом применяется ряд технических приемов, улучшающих сходимость и предотвращающих переобучение (overfitting). В итоге на входе нейросети — несколько фреймов MFCC-features (центральный фрейм подлежит классификации, остальные образуют

контекст), на выходе — около 4000 нейронов, соответствующих различным сенонам. Эта нейросеть используется как акустическая модель в production-системе.

2.4 Рекуррентные нейронные сети

LSTM — это класс возвратных нейронных сетей. Поэтому, прежде чем мы сможем перейти к LSTM, важно понять нейронные сети и рекуррентные нейронные сети [8].

Нейронные сети - Искусственная нейронная сеть представляет собой слоистую структуру из связанных нейронов, вдохновленную биологическими нейронными сетями. Это не один алгоритм, а комбинация различных алгоритмов, которая позволяет нам выполнять сложные операции с данными. Рекуррентные нейронные сети - это класс нейронных сетей, предназначенных для работы с временными данными. Нейроны RNN имеют состояние / память ячейки, и ввод обрабатывается в соответствии с этим внутренним состоянием, которое достигается с помощью петель в нейронной сети. В RNN существуют повторяющиеся модули «tanh» слоев, которые позволяют им сохранять информацию. Однако ненадолго, поэтому нам нужны модели LSTM.

LSTM - Это особый вид рекуррентной нейронной сети, способной изучать долгосрочные зависимости в данных. Это достигается за счет того, что повторяющийся модуль модели имеет комбинацию четырех слоев, взаимодействующих друг с другом [8].

На (Рис. 2.5) отображены структуры вышеупомянутые виды рекуррентных сетей:

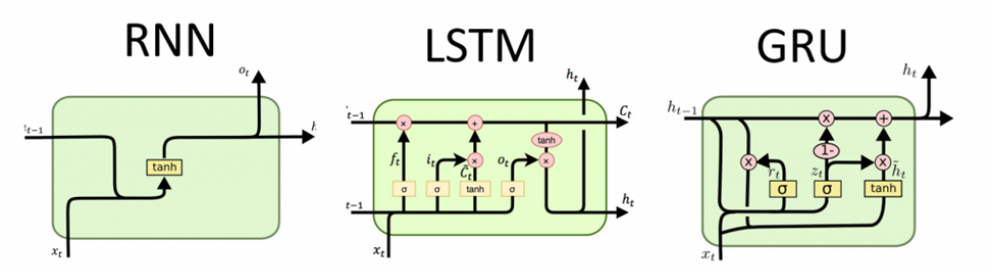

Рис. 2.5 – Структуры рекуррентных нейронных сетей

2.5 Выводы

В данном разделе была сформулирована постановка задачи, рассмотрены возможные ее решения с помощью архитектуры eBICA и подхода с использованием машинного обучения. Выделен подход для распозвавания речи.

3. Проектирование модели поведения виртуального агента

В этом разделе описывается и обосновывается выбор инструментария для проектирования и программного воплощения модели поведения актора в заданной парадигме. Описываются ключевые моменты проектирования и программной реализации модели поведения актора.

3.1 Проектирование модифицированного прототипа Виртуального Актора

В данном разделе описывается реализация когнитивной модели в виде функций, интерпретируемых в форме псевдокода. Здесь приведены основные функции, вызываемые при работе алгоритма по выбору действия для Виртуального Агента. В начале будет описан алгоритм по выбору ответного действия Виртуального Актора на действие человека.

На вход поступает действие человека, целью которого является влияние на Виртуального Актора. В зависимости от характеристики действия пересчитываются оценки Appraisals человека и Виртуального Актора по формуле 3. Соответственно уже на данном этапе меняется характеристика взаимодействующих акторов, что может повлиять на выбор ответного действия пингвина. Следующим шагом в соответствии с режимом работы моральной схемы производится вычисление новых значений векторов Feelings для взаимодействующих акторов.

Работа моральной схемы, следующая - по умолчанию она выключена и значения Feelings вычисляются по формуле 4. В этот момент времени моральная схема не оказывает никакого воздействия на принятие решения Виртуальным Актором. При отклонении абсолютного значения вектора Feelings от начального положения больше, чем на заранее заданную константу StartMoralSchema - моральная схема начинает свою работу. Ее функционирование происходит теперь в двух режимах:

1. Feelings вычисляется по формуле номер 5 в случае, если (расхождение между Feelings и Appraisals больше заранее определенной константы). Данный режим работы моральной схемы называется конфликтным. В таком ключе продолжается работа пока верна формула, описанная выше.
2. Feelings константна и экстремальна по своим параметрам. Это означает, что Виртуальный Актор понимает каким образом нужно относиться к человеку: как к другу или врагу, как к подчиненному или начальнику и т.д. В данном режиме схема работает до тех пор, пока расхождения между Feelings и Appraisals не станет критическим. В этом случае снова включается режим номер 1.

Возможен также вариант, когда отклонение Feelings от начального положения крайне мало. В таком случае моральная схема снова выключается и Feelings вычисляется по формуле 4.

Первая функция описывает работу моральной схемы. В данной функции определяется режим работы моральной схемы и каким образом будет меняться субъективная оценка человека в отношении Виртуального актора.

На данном этапе получается следующая картина - обработано действия человека, направленное на пингвина, и пересчитаны “оценки” и “чувства” взаимодействующих акторов. Далее на основе соматического критерия и когнитивного фактора вычисляются вероятности для всех возможных действий в данной ситуации. Набор всех действий для Виртуального Актора заранее описан в *. json файле и у каждого действия есть параметр, который определяет в каком контексте оно применимо. Соответственно после определения действий применимых в данной ситуации, для каждого из них вычисляется вероятность на основе когнитивного фактора по формуле 6, на основе соматического по формуле 7 и итоговая вероятность по формуле 8.

Следующим шагом необходимо определить ответное действия для пингвина из возможных с использованием ранее посчитанных для них вероятностей. Все действия сортируются по возрастанию численного значения их вероятностей. При помощи функции рандомной генерации чисел, основанной на равномерном распределении, генерируется дробное число от 0 до 1. Далее по формуле 9 определяется ответное действие.

Где - вероятность i-го действия, k - число, пробегающее от 1 до n, где n - общее количество рассматриваемых действия для Виртуального Агента. Минимальное k, при котором выполняется неравенство, описанное в формуле 9, соответствует номеру действия в списке действий, отсортированном по возрастанию вероятностей. Это действие и будет совершено Виртуальным Актором.

Предпоследним этапом работы алгоритма является пересчет значений Appraisals и Feelings по описанной в начале данного пункта схеме.

В конце действие, которое должен выполнить пингвин возвращается в виде строкового значение в функцию, отвечающую за перемещение и действия пингвина в виртуальном окружении и выбранное действие визуализируется Виртуальным Актором. После этого человек может снова совершить воздействие на пингвина, и вся данная процедура снова повторится.

Далее будет описан алгоритм выбора самостоятельного действия Виртуальным Актором. Под самостоятельным действием понимается такой действия, которое предпринимает пингвин, основываясь на состоянии виртуального окружения и отношении с человеком. Алгоритм выбора самостоятельного действия схож с алгоритмом выбора ответного действия на действие со стороны человека, направленное на Виртуального актора. Исполнение начинается с шага

вычисления вероятностей для возможных в данной ситуации действий на основе когнитивного и соматического фактора. В данной парадигме возможно проявление одного из следующих взаимодействий со стороны пингвина:

- Подойти к корзине со снежками с желанием поиграть
- Подойти к корзине с рыбой и попросить поесть
- Подойти к человеку с целью взаимодействия с ним
- Пойти спать
- Посмотреть в сторону человека
- Поприветствовать человека

Здесь может быть проблема с бесконечной очередью вызовов действий без временного интервала между ними. Тогда взаимодействие с Виртуальным агентом может стать проблематичным. Возможны два пути решения данной проблемы. Первое решение — это создание некоторого минимального промежутка времени после выполнения самостоятельного действия пингвина, в течение которого, программно будет запрещено совершение самостоятельного действия для Виртуального Актора. Такой подход в определенной мере решает проблему бесконечной непрерывной очереди действий пингвина, однако он довольно искусственный и слабо согласуется с моделью социально-эмоционального интеллекта на основе когнитивной архитектуры eBICA. Второй подход заключается в создании действий “заглушек”, которые практически не влияют на соматические и когнитивные оценки акторов. Данными действиями могут быть, например:

- Пингвин стоит на месте
- Пингвин перемещается по виртуальному окружению в какую-либо его точку

При совершении этих действий оценки Виртуального Актора и человека практически не будут меняться и будет создан временной интервал между социальными действиями пингвина.

3.2 Модель тембора

В связи с задачей эмоционального взаимодействия ?пользователя с вирутальным актором ставится задача распознования эмоциональной составляющей из человеческой речи. А также классификация эмоционального воздействия.

Так как в данной работе для анализа человеческой речи используется нейросеть, а это значит, что требуется датасет для обучения данной нейросети. В большинстве работ по классификации эмоций присутсвующих в речи человека используются открытые данные RAVDESS. Этот датасет включает в себя 7356 аудио записей с эмоционнальным наполнением.

Область распознавания речи и ее эмоцианальной составляющей довольно обширна. И решение данной задачи с нуля является слишком трудозатратной. Поэтому в работе будет использована предобученная модель. Данная модель позволит преобразовывать аудиоинформацию в векторное пространство.

Вектор полученный из записи человеческой речи будет использован для обучения нейронной сети, которая по признакам вектора определит эмоциональную составляющую.

Для того, чтобы получить решение данной задачи будут использованы такие модели (Рис. 3.1):

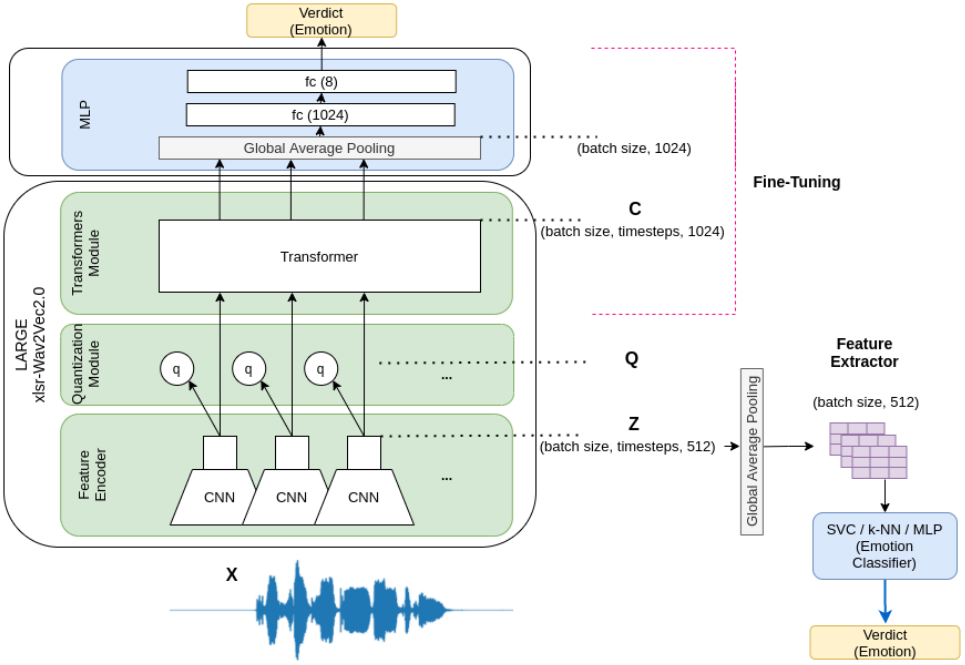

Рис. 3.1 – Предлагаемые конвейеры для распознавания речевых эмоций.

SVM с ядром «RBF» - является одной из наиболее популярных методологий обучения по прецедентам, предложенной В. Н. Вапником и известной в англоязычной литературе под названием SVM (Support Vector Machine).

k-ближайших соседей (kNN) с большинством голосов для выбора класса - расшифровывается как k Nearest Neighbor или k Ближайших Соседей — это один из самых простых алгоритмов классификации, также иногда используемый в задачах регрессии.

Многослойный персептрон (MLP) - это класс искусственных нейронных сетей прямого распространения, состоящих как минимум из трех слоёв: входного, скрытого и выходного. За исключением входных, все нейроны использует нелинейную функцию активации.

3.3 Модель семантики

Текст человеческой речи позволяет передать не только непосредственно смысл, который закладывает в нее говорящий, но и эмоциональную составляющую. Конечно текстовый формат представления не позволяет передать интонацию, но даже по тексту можно выявить эмоцию говорящего.

В данной работе требуется в дополнение к эмоциональной оценке по аудиозаписи получить эмоциональную оценку по тексту. Что позволит наиболее точно установить эмоциональное взаимодействие с виртуальным актором. Так же такой анализ позволит выявить случаи сарказма в речи ?пользователя.

Для упрощения будем классифицировать текста шкале: негативная оценка - нейтральная оценка - позитивная оценка. Сегодня уже существуют достаточно точные решения, определяющие эмоциональную составляющую текста. Но большинство таких решений являются нейросетями обученными на полноценных текстах редко включающих в себя диалоговую составляющую. Тогда как в работе требуется только анализ дилогов. Поэтому будет взята модель анализа тональности текста и дообучена на диалоговых данных. Данные будут собраны посредством фильтрации диалогов из публично доступных датасетов с сохранением разметки.

3.4 Инструменты для анализа текста

После того как речь была распознана и конвертирована в текстовый форомат ставится задача определить семнатический смысл предложения.

Возможность идентификации семантической близости между словами сделала модель word2vec широко используемой в NLP-задачах, которые подробно описываются в [13]. Идея word2vec основана на контекстной близости слов. Каждое слово может быть представлено в виде вектора, близкие координаты векторов могут быть интерпретированы как близкие по смыслу слова [14].

Таким образом, извлечение семантических отношений (отношение синонимии, родовидовые отношения и другие) может быть автоматизировано. Установление семантических отношений вручную считается трудоемкой и необъективной задачей, требующей большого количества времени и привлечения экспертов. Но среди ассоциативных слов, сформированных с использованием модели word2vec, встречаются слова, не представляющие никаких отношений с главным словом, для которого был представлен ассоциативный ряд [15].

В работе рассматриваются дополнительные критерии, которые могут быть применимы для решения данной проблемы. Наблюдения и проведенные эксперименты с общеизвестными характеристиками, такими как частота слов, позиция в ассоциативном ряду, могут быть использованы для улучшения результатов при работе с векторным представлением слов в части

определения семантических отношений для русского языка.

Представление слов в виде векторов позволяет применять математические операции. В большинстве примеров можно встретить вычитание векторов, когда результат вычисления vec('Madrid') - vec('Spain') + vec('France') будет ближе к vec('Paris'), чем к другим векторам из распределения. Таким образом, разница векторов может быть использована для поиска семантических отношений между словами [16].

Word2vec не возвращает напрямую семантические отношения между словами. В ассоциативном ряду, который может быть возвращен в качестве близких слов к запрашиваемому (главному) слову, отражаются слова, которые часто употребляются рядом в контексте. Бесспорно, в ассоциативном ряду встречаются синонимы, антонимы, гипонимы, гиперонимы, холонимы, меронимы, ассоциации и другие типы, которые могут быть определены как семантические отношения.

Для реализации используются такие пакеты языка программирования python как: ufal.udpipe и wget. Как было сказано ранее для нахождения близости слов мы используем готовую модель, которую взяли с ресурса https://rusvectores.org/static/models с помощью пакета wget, принцип работы описан в [22]. Данный пакет позволяет скачивать данные с веб ресурсов посредством GET запроса.

Для того чтобы модель word2vec, описанная в [23] могла определить расстояние между словами, ей следует передать слово, ставится задача определить его часть речи, для того, чтобы автоматизировать такой процесс по определению части речи слова из текста, мы должны с помощью системы word2vec понять по слову какими категориями частей речи оно обладает, самые распространённые варианты – это существительные и глагол, далее после определения, мы передаем их на анализ дистанции. Принцип работы модели представлен на (Рис. 4.10):

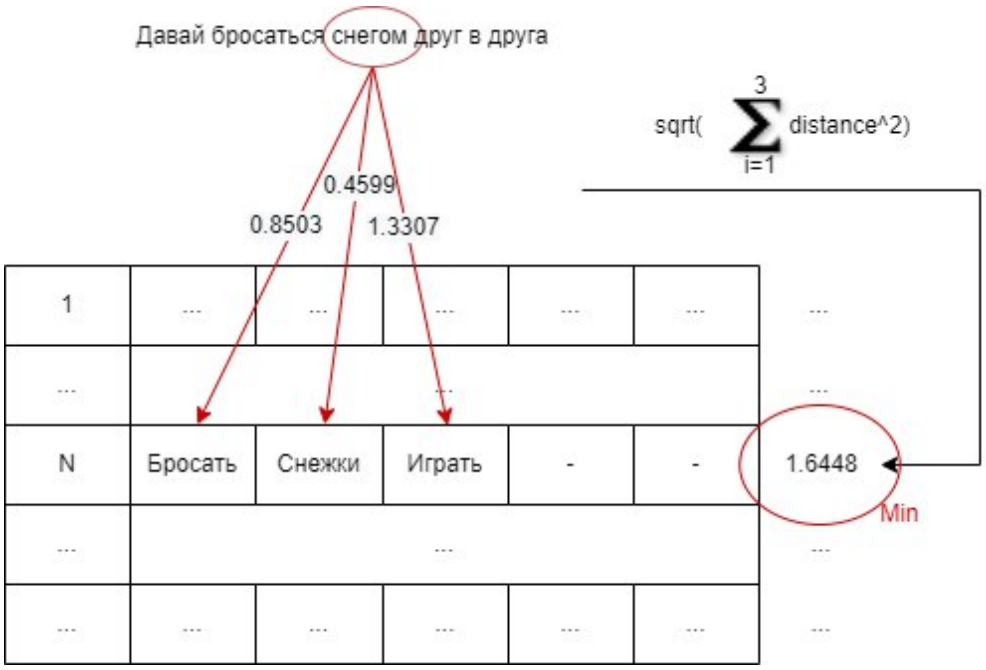

Рис. 3.2 – принцип работы word2vec

В рамках данной работы используется модель ruwikiruscorpora_tokens_elmo_1024_2019 - модель UDPipe для нахождения синонимов. Данная модель позволяет узнать семантическую близость слов.

Чтобы начать применять указанную выше нейронную сеть непосредственно, следует подготовить действия виртуальных акторов так, чтобы ими могла оперировать частично или полностью сама нейронная сеть. Для этого возьмем разобьем описание каждого действия на ключевые слова так, что каждое действие будет ассоциировано с набором слов. При этом создан класс отражающий действие актора как сущность - VAAction(Virtual Actor Action), принцип которой описан в [24].

Данный класс содержит в себе описание действия и ключевые слова, описывающие данное действие. Среди таких слов отсутствуют предлоги, а сами слова представлены в нормальной форме (для существительных - именительный падеж единственное число).

Для построения сценариев берутся текста, полученные в разделе выше. Для каждого текста выделяются ключевые слова, которые наиболее близки к ключевым словам, описывающим действия акторов. Что происходит в процессе итерации через все слова текста так, что для каждого слова применяется считается значение близости слова к действию виртуального актора. Для работы с текстом создан класс WText (Web Text), который является ответственным за итерацию через все слова текста. С помощью методов класса задается функция, которая будет применяться к каждому слову. В качестве такой функции берется функция, которая находит близость слова с ключевыми словами экземпляра класса VAAction.

Также класс WText формирует набор определяющих текст слов, эти слова выбираются так, что значение семантической близости больше, чем заданный заранее порог, данной работе порог равен 0.08. После того как все тексты были переведены в экземпляры класса WText, была произведена оценка кол-ва текстов с одинаковым кол-вом определяющих слов.

3.5 Построение блок-схем и UML диаграмм

В данном разделе строятся блок-схемы реализованного алгоритма (Построение такой диаграммы рекомендуется в работе [17]). На (Рис. 3.3) представлена упрощенная блок-схема работы алгоритма по выбору ответного действия для актора. Данный алгоритм срабатывает каждый раз после совершения действия человеком, которое направлено на пингвина. Также данный алгоритм может приостанавливать любое продолжительное действие самого пингвина.

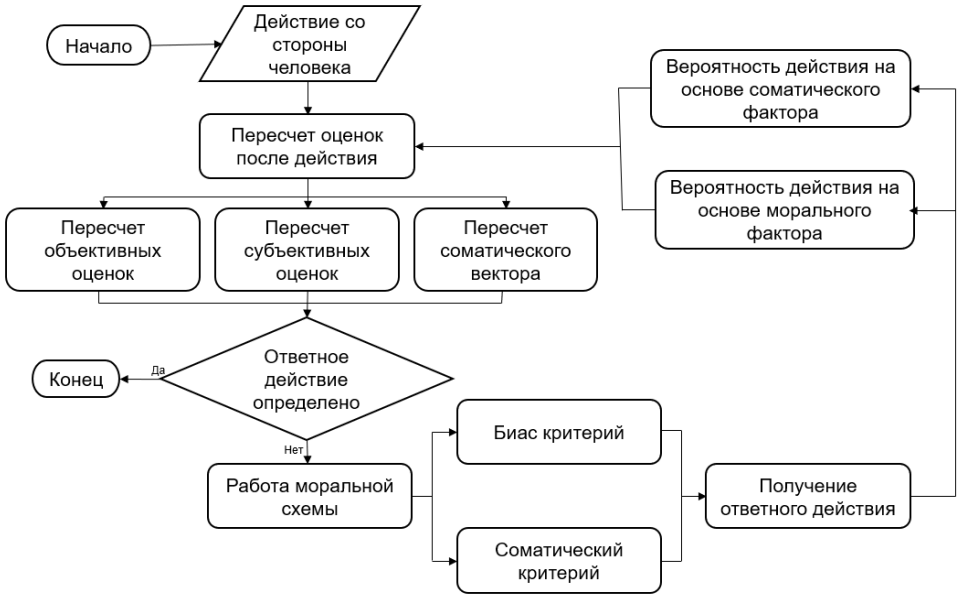

Рис. 3.3 – Блок-схема общей работы алгоритма

На вход поступает действие человека, направленное на пингвина. На основе оценок данного действия пересчитывается Appraisals и Feelings человека и пингвина, а также вектор соматического состояния. Затем последовательно срабатывают две функции: Критерий Биас и Соматический критерий. В них определяется вероятность для каждого действия на основе когнитивного и соматического состояния соотвственно. Далее на основе данных вероятностей определяется итоговое действие, которое будет выполнено пингвином. Данное действие определяется наибольшей вероятностью.

ассмотрим более подробно алгоритм пересчета Feelings (субъективных оценок) для акторов. На (Рис. 3.4) представлена блок-схема данного алгоритма. На вход поступают Appraisals и Feelings актора. Затем, если до этого никогда данному актору не присваивалась константная

оценка Feelings, то Feelings присваивается значение согласно уравнению, когда моральная схема выключена и не оказывает никакого воздействия на вектор Feelings. Если Feelings актора уже принимал заданное константное экстремальное значение, то Feelings присваивается значение в зависимости от разницы норм Feelings и Appraisals.

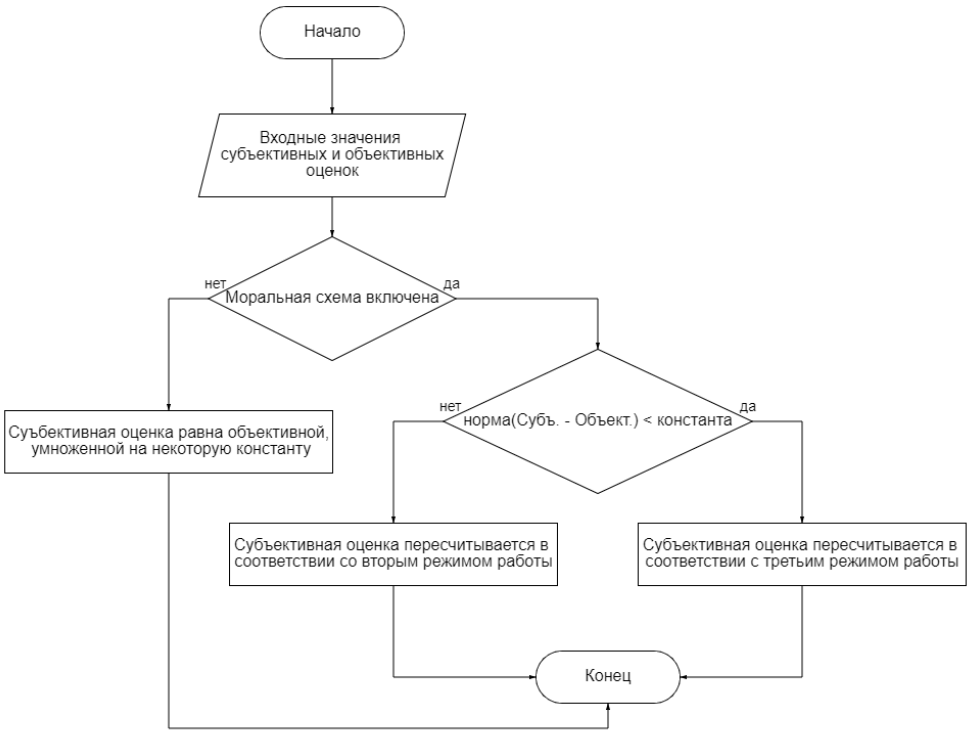

Рис. 3.4 – Блок-схема алгоритма пересчета значений Feelings акторов

На (Рис. 3.5) показана диаграмма классов, описывающая процедуру взаимодействия Виртуального Агента и человека. Диаграмма наглядно демонстрирует взаимосвязь человека и Виртуального Актора с методами логирования и выбора действий для пингвина с пересчетом “оценок” и “чувств”. При использовании данной схемы алгоритм поведения Виртуального Актора, основанного на когнитивной архитектуре eBICA, можно легко перенести и адаптировать под другую парадигму и виртуальное окружение.

Класс Human описывает характеристики человека, проводящего игровую сессию с пингвином. Класс Penguin представляет собой Виртуального Агента, реализованного в виде пингвина в виртуальном окружении. Human и Penguin наследуются от общего класса Actor и имеют характеристики: Appraisals - “оценки”, Feelings - “чувства”, CoordinateX, CoordinateY, Azimuth - координаты местонахождения в виртуальном окружении и угол поворота относительно севера (север представляет собой сильно удаленную точку от площадки взаимодействия человека и пингвина и определен заранее). Также у классов Penguin и Human есть уникальные поля. У Penguin поле Somatic, которое представляет собой вектор соматического состояния, представ-

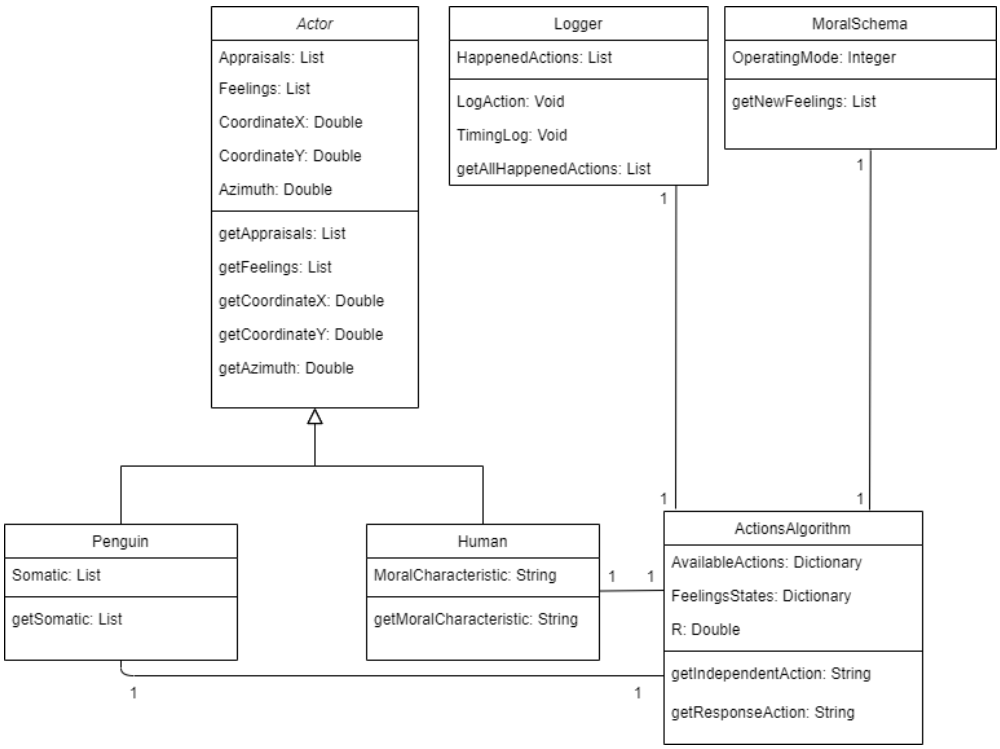

Рис. 3.5 – Диаграмма классов, описывающая взаимодействие Виртуального агента и человека

ленного структурой List. У Human поле MoralCharacteristic - представляет собой строковое значение, которое показывает, как пингвин относится к человеку с точки зрения моральной схемы. Penguin связан ассоциацией с классом ActionsAlgorithm. Данный класс отвечает за выбор действий для Виртуального Актора и обрабатывает действия человека. ActionsAlgorithm связан с классом MoralSchema, который представляет собой моральную схему. Принцип ее работы был описан в разделе 3.1. MoralSchema связана ассоциацией с Logger. Этот класс отвечает за логирование всех действий со стороны человека и пингвина. Также срабатывает функция TimingLog каждые 4 секунды для сохранения в файл текущего состояния виртуального окружения.

В дополнение к классам приложения до модернизации проектируются классы для работы с человеческой речью (Рис. 3.6):

Для работы с человеческой речью непосредственно используется класс Voice, который отвечает за ф ильтрацию речи и разбиение аудиодорожки на фреймы. Фреймы передаются в классы EmotionFromVoice и TextFromVoice. Первый извлекает эмоциональную составляющую из фрейма. Второй распознает речь и возвращает текст, который используется в классах EmotionFromText и CommandFromText. EmotionFromText как и EmotionFromVoice извлекает эмоциональную составляющую, но уже из текста. CommandFromText итерируется по полученному тексту и сопостовляет ключевые слова с действиями пи нгвина по алгоритму из рисунка 4.10 . Полученные команды передаются в ActionAlgorithm для того, чтобы увеличить вероятность выполнения

распознанных действий. Все эмо циональные оценки передаются в Penguin и меняют его состояние соответсвующе.

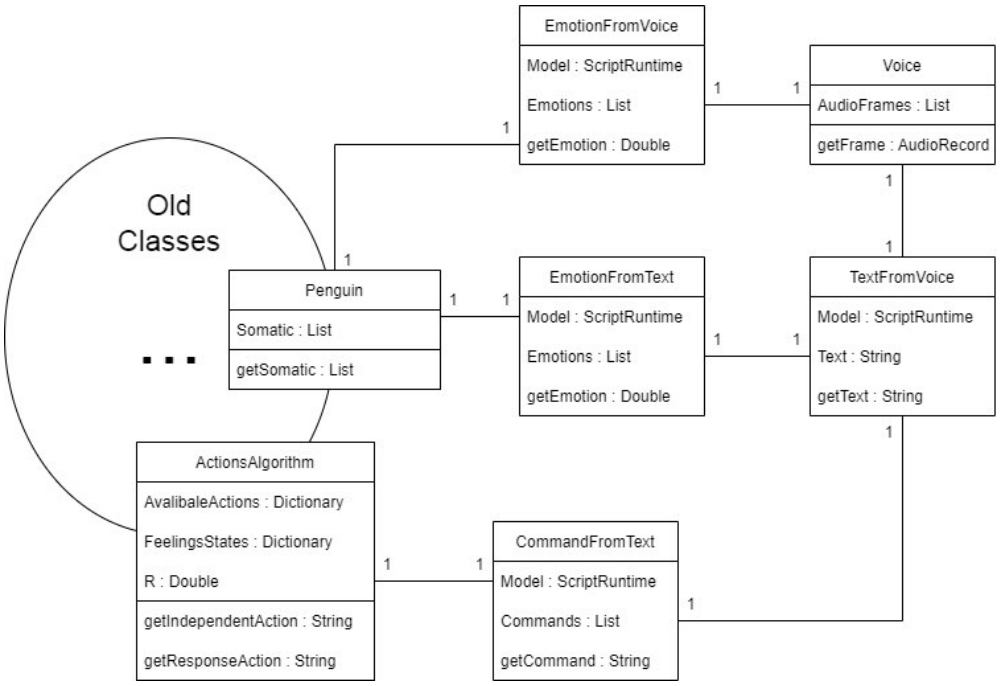

Рис. 3.6 – Расширенная диаграмма классов с участием голосового распозвавания

3.6 Выводы

бла бла бла

4. Реализация программного продукта

В этой главе описаны практические методы и их частичное представлении реализации, которые мы использовали для достижения поставленной цели.

4.1 Использование парсера данных

Так как для обучения нейронной сети статическим контентом, а именно: сайты с анекдотами и сайты со сценариями юмористических сценок. Для того чтобы собрать все эти данные потребуется решить задачу парсинга нужны большие массивы данных, возникает задача сбора данных относящихся к юмору(тории юмора). Источников таких данных не так много, основынми будут веб ресуры с юморивеб сайтов.

Поэтому опишем алгоритм решающий такую задачу. В качестве вводных возьмем url ссылки на веб ресурсы. Так как веб сайты могут иметь разную архитектуру возникает задача построения унификационного решения данного вопроса.

Как было описано ранее для полного парсинга страницы необходимо использовать рекурсивный метод, принцип которого подробно описан в parser02, который выражен в том, что мы проходим все ссылки на выбранном ресурсе. Основная функциональность заключается в следующих строчках :

```
class WSManager:
    def __init__(self, base_url):
        self.base_url = base_url
    def get_urls(self, level):
        urls = []
        try:
            links_1 = []
            start_link = self.base_url
            links_1.append(start_link)
            for i in links_1:
                response = requests.get(i)
                soup = BeautifulSoup(response.content, "html.parser",
parse_only=SoupStrainer(['a', 'img']))
                full_list = [link['href'] for link in soup if link.get('href')] +
[img['src'] for img in soup if img.get('src')]
                full_list = list(set(full_list))
                for url in full_list:
                    if not url.startswith('https:/'):
                        if url.startswith('/'):
                            if url.find('.org') == -1:
                                url = start_link + url[1:]
                                full_list.append(url)
                            elif url.find('.org'):
                                url = 'https:' + url
                                full_list.append(url)
                        elif url.startswith('//'):
                            url = start_link + url[2:]
                            full_list.append(url)
                        else:
                            pass
                    elif url.startswith('https:/'):
                        full_list.append(url)
                        urls.append(full_list)
                self.get_urls(level - 1)
                links_1 = full_list
                links_1 = list(set(links_1))
                return links_1
        except MemoryError as e:
            print(e)

        return urls
```

Рис. 4.1 – Класс WSManager - Основная функциональность рекурсивного парсинга

Имея ссылку на страницу, мы можем полуть html код страницы. По выше указанному коду найдем все ссылки на ресурсы данного веб сервиса предоставленные на этой странице. Осмотрим все полученные ссылки таким же образом. В итоге мы получим html коды практически всех страницы доступных на принятом в рассмотрение ресурсе.

URL ссылки на веб сервисы предоставляющие, данные о юмористических текстах, агрегируются в экземплярах класса WSManager (Web Site Manager). Далее каждый экземпляр класса создает и запускает экземпляры класса RManager (Request Manager). Которые занимаются асинхронным получением данных с сайтов.

Для классического получения данных при наличии API, код будет выглядеть следующим образом:

```
url = self.base_url
data = {"content": message}
header = {"authorization": token}
r = requests.post(url, data=data, headers=header)
```

Рис. 4.2 – Получения данных при наличии API

При получении положительного ответа 200, запрос отработал успешно. В данной работе класс RManager использует корутины, что позволяет существенно сэкономить время ожидания всех ответов веб сервисов. Указанная экономия целиком и полностью объясняется тем, что пока класс ожидает ответ на первые запросы, он не перестает слать последующие. А те запросы, ответ на которые уже был получен, передаются в экземпляры класса WHParser так, что на каждую полученную html страницу создается один экземпляр такого класса.

```
def t(self):
        if self.token is None or self.expiration is None or self.expiration <
datetime.now():
            loop = asyncio.get_event_loop()
            loop.run_until_complete(self. __get_smth_for_smth__())
        if self.expiration > datetime.now():
            return self.token

async def __get_smth_for_smth__(self):
        loop = asyncio.get_event_loop()
        future = loop.run_in_executor(None, functools.partial(requests
self.base_url))
        resp = await future
```

Рис. 4.3 – функция получения данных HTML класса WHParser

Предположим, что на сайте еще стоит ограничение по запросам, поэтому предварительно, мы имеем лист проверенных и подготовленных рабочих прокси. При этом статусы готовности, которых получены в отдельных скриптах языка python3 с использованием корутин из программного модуля asyncio.

```
new_proxies = [i for i in [proxies_list[i:i+2] for i in
range(0,len(proxies_list),2)]]
       proxies = [{'IP': new_proxies[i][0], 'Port': new_proxies[i][1], 'Status':
'unknown'} for i in range(len(new_proxies))]
```

Рис. 4.4 – Обработка proxy

{'IP': '138.201.120.214', 'Port': '1080', 'Status': 'Alive'}
{'IP': '51.222.40.26', 'Port': '9090', 'Status': 'Alive'}
{'IP': '92.42.109.189', 'Port': '1080', 'Status': 'Alive'}
{'IP': '54.37.160.92', 'Port': '1080', 'Status': 'Alive'}
{'IP': '20.105.253.176', 'Port': '8080', 'Status': 'Alive'}
{'IP': '103.73.194.2', 'Port': '80', 'Status': 'Alive'}
{'IP': '54.37.160.90', 'Port': '1080', 'Status': 'Alive'}
{'IP': '134.119.206.106', 'Port': '1080', 'Status': 'Alive'}
{'IP': '176.241.89.244', 'Port': '53583', 'Status': 'Alive'}

Рис. 4.5 – Проверка статуса прокси серверов

Далее помещаем наши прокси в ассинхронные запросы так, что при неудовлетворительном ответе с сервера, класс RManager меняет используемую прокси.

```
some_req = Request('https://www.example/.org/', proxies =
proxies[random.randint(len(proxies))], timeout=5,verify=False)
some_req.add_header('User-Agent', ua.random)
some_doc = urlopen(some_req).read().decode('utf8')
```

Рис. 4.6 – пример работы RManager

WHParser, получив код веб страницы, обрабатывает его с помощью модуля BaeautifulSoup. Выделяет текста на основании тегов, определенных экспертами для каждого веб сервиса. soup = BeautifulSoup(some_doc, 'html.parser')

Для вытаскивания данных из таблиц используем обращения к классам HTML. Атрибут class указывает одно или несколько классов для элемента. Атрибут class используется в основном для того, чтобы указывать на класс в таблице стилей. Однако он также может использоваться JavaScript (через HTML DOM) для внесения изменений в элементы HTML с указанным классом. Так же надо учесть то, что текстовые элементы традиционно заключают в элементы с теговым названием “p”, но полный семантически связанный текст может быть разбит множества предложений, таким образом, что отдельные элементы одного и того же уровня вложенности будут представлять эти разбитые множества. В связи с этим возникает задача восстановления семантически связанного текста из элементов одинаковой вложенности и одинакового типа, что следует учитывать при дальнейшей разработке.

```
for table in soup.find_all('table'):
        try:
            if 'table-striped' in table['class']:
                parsed = [[i] for i in [i.text for i in table.find_all('p')]]
        except KeyError:
            pass
```

Рис. 4.7 – Обращение к атрибутам HTML

Получаем данные в формате:

```
for table in soup.find_all('table'):
        try:
            if 'table-striped' in table['class']:
                parsed = [[i] for i in [i.text for i in table.find_all('p')]]
        except KeyError:
            pass
```

Рис. 4.8 – Полученные данные

4.2 Анализ и отбор полученных данных

После получения текстовых данных с выбранных ресурсов, требуется провести просеивание массива на данные, излишне затронутые в процессе парсинга, так как на сайте слишком много текста и достаточно большой процент содержания лишнего текста по типу маркировок, контекстной рекламы и прочего, а наша задача получить интересующие нас юмористические текстовые данные в совокупности html страниц.

Значимый текст выбирается на основании анализа его принадлежности к заранее определенным классам текста (или тематикам). Как правило, методы автоматической классификации основаны на методе машинного обучения: сначала получают обученную с помощью какого-либо алгоритма модель, качество которой определяет точность классификации. Таким образом, процесс обучения зависит от выбранного алгоритма и «чистоты» обучающей выборки. Для решения такой проблемы выделим тэги и классы элементов, содержащих юмористические данные. Чтобы понять какие данные нам подходят, а какие нет.

Следует учесть, что большое количество классов (десятки и сотни) приводит к увеличению трудоемкости обучения и понижению точности классификации, которая описана в [18]. Тематики в юморе, близки по своей сути (например, экономика и бизнес), что приводит к тому, что классы в обучающей модели начинают пересекаться, приводя к снижению точности.

В таких случаях, классификация проходит по принципу объединения классов в один, а затем используют под классификацию или повторную классификацию документов внутри класса.

В данной системе автоматической классификации используется популярный метод опорных векторов (или SVM – Support Vector Machine) с мерой TFiDF [19]. Написано консольное приложение на языке программирования C#, целью которого ставится использование библиотеки dll - LingvoNET, которая представляет собой модель специально обученную на нескольких классах, определенных заранее, принцип проектирования описан в [20]. Для использования данного ресурса в проекте реализуется класс написанные на языке программирования Python, который взаимодействует с консольным приложением через стандартный вход и стандартный выход.

Классы содержащиеся в библиотеке LingvoNET – представлены ниже:

- Экономика и бизнес
- Шоу-бизнес и развлечения
- Семья
- Мода
- Компьютерные игры
- Здоровье и медицина
- Политика
- Наука и технологи
- Спорт
- Туризм, путешествия
- Недвижимость
- Авто

Согласно этим классам, происходит классификация каждого входящего текста с учетом его меры близости к тому или иному классу, так же как и в юморе чаще всего есть семантическая принадлежность к тому или иному классу представленному выше, однако такие классы как “Игры”, “Спорт”, “Мода”, “Авто” и “Недвижимость”, чаще всего относятся к контекстной рекламе. Если документ близок к двум тематикам, то он попадает в соответствующие два класса. Если текст похож сразу на несколько тематик, то, скорее всего, это шум и не семантически не подходит.

Качество классификации чаще всего оценивается по двум критериям: точностью и полнотой классификации согласно работе [21].

Точность - показывает, насколько точно тексты с ресурсов попадают в определенный класс.

Полнота - определяется соотношением текстов, релевантных данному классу, к общему количеству релевантных текстов. Точность можно повышать, задавая порог прохода текста в тот

или иной класс, при этом полнота классификации будет уменьшаться. Как правило, стараются найти оптимальное соотношение этих критериев.

Представим пример полученного текста с ресурса:

– Мам, помоги с сочинением. Нужно не меньше 350 слов.
– А что за тема?
– Некрасов: ”Кому на Руси жить хорошо”.
– Напиши — ”депутатам”.
– Но надо 350 слов.
– Пиши поименно!

В данном случае получим следующие результаты анализа представленных выше данных:

Таблица 4.1 – Результат семантического анализа подходящего текста

Семья	46.81%
Политика	19.17%
Другие	менее 10%

Текст будет отнесен к классу “Семья” и семантически подходит нам в качестве искомого вида юмористического текста.

Иной же случай, это пример контекстной рекламы: “Горячие предложения контрафактного алкоголя на каждой подворотне! Тема, которая волнует репортерское агентство в этом городе. Бодяжная сивуха убивает людей, о чем думает администрация???”

Таблица 4.2 – Результат семантического анализа не подходящего текста

Недвижимость	14.18%
Здоровье и медицина	12.25%
Политика	11.55%
Семья	10.91%
Экономика и бизнес	10.45%
Шоу-бизнес и развлечения	10.02%
Другие	менее 10%

В данном случае текст будет определяться как шум, так как не проходит ни один порог принадлежности к тому или иному заранее заданному классу. В процессе анализа текста по принадлежности, одновременно запоминаются пути на html странице, а именно, названия атрибутов к которым относится текст на странице html, если в процессе анализа были найдены тексты с одинаковыми путями и при этом все они проходят порог классификации как юмористические, то текстовые данные под такими путями и атрибутами считаются юмористическими и подлежат дальнейшей работе без семантического анализа. Таким образом, оптимизирован поиск требуемых данных на ресурсах и с помощью выбора вышеперечисленных инструментов определяется востребованность данных и отсеивается лишний, для данного проекта, контент.

В результате получаем следующий набор данных, представленный на (Рис. 4.9):

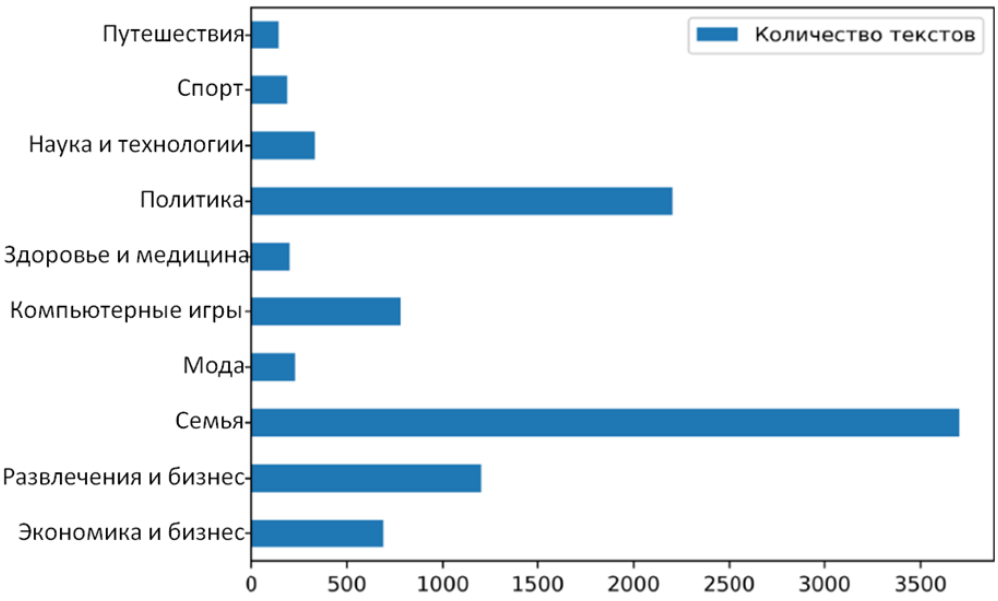

Рис. 4.9 – Диаграмма распределения по количеству, собранных текстов с сайтов

4.3 Разметка отобранных данных

Для реализации моделей из третьего раздела, используются такие пакеты языка программирования python как: ufal.udpipe и wget. Как было сказано ранее для нахождения близости слов мы используем готовую модель, которую взяли с ресурса https://rusvectores.org/static/models с помощью пакета wget, принцип работы описан в [22]. Данный пакет позволяет скачивать данные с веб ресурсов посредством GET запроса.

Для того чтобы модель word2vec, описанная в [23] могла определить расстояние между словами, ей следует передать слово, ставится задача определить его часть речи, для того, чтобы автоматизировать такой процесс по определению части речи слова из текста, мы должны с помощью системы word2vec понять по слову какими категориями частей речи оно обладает,

самые распространённые варианты – это существительные и глагол, далее после определения, мы передаем их на анализ дистанции. Принцип работы модели представлен на (Рис. 4.10):

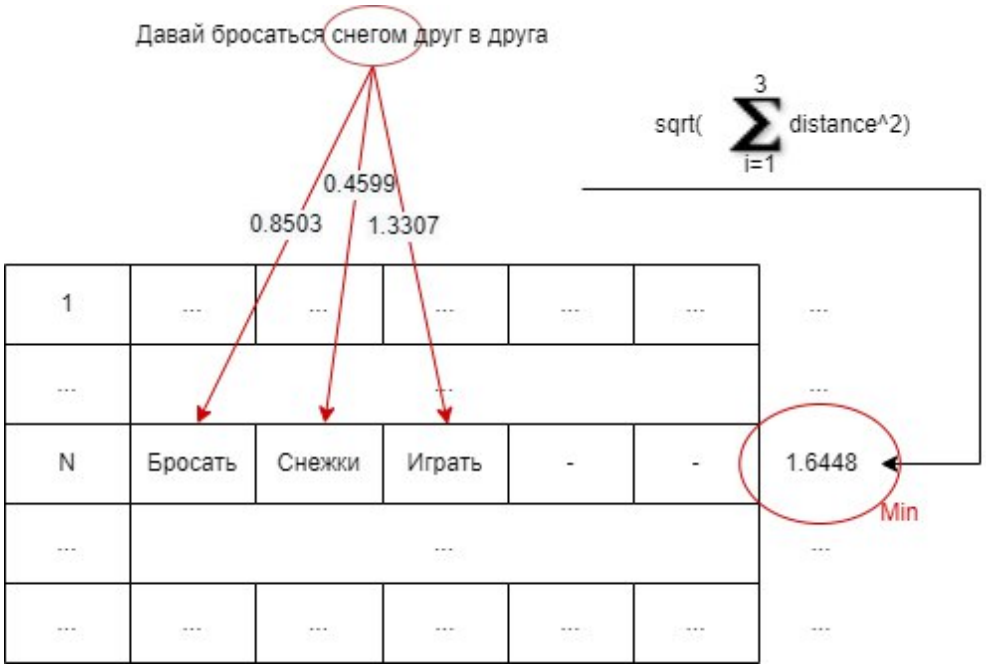

Рис. 4.10 – принцип работы word2vec

В рамках данной работы используется модель ruwikiruscorpora_tokens_elmo_1024_2019 - модель UDPipe для нахождения синонимов. Данная модель позволяет узнать семантическую близость слов.

Чтобы начать применять указанную выше нейронную сеть непосредственно, следует подготовить действия виртуальных акторов так, чтобы ими могла оперировать частично или полностью сама нейронная сеть. Для этого возьмем разобьем описание каждого действия на ключевые слова так, что каждое действие будет ассоциировано с набором слов. При этом создан класс отражающий действие актора как сущность - VAAction(Virtual Actor Action), принцип которой описан в [24].

Данный класс содержит в себе описание действия и ключевые слова, описывающие данное действие. Среди таких слов отсутствуют предлоги, а сами слова представлены в нормальной форме (для существительных - именительный падеж единственное число).

Для построения сценариев берутся текста, полученные в разделе выше. Для каждого текста выделяются ключевые слова, которые наиболее близки к ключевым словам, описывающим действия акторов. Что происходит в процессе итерации через все слова текста так, что для каждого слова применяется считается значение близости слова к действию виртуального актора. Для работы с текстом создан класс WText (Web Text), который является ответственным за итерацию через все слова текста. С помощью методов класса задается функция, которая будет приме-

няться к каждому слову. В качестве такой функции берется функция, которая находит близость слова с ключевыми словами экземпляра класса VAAction.

Также класс WText формирует набор определяющих текст слов, эти слова выбираются так, что значение семантической близости больше, чем заданный заранее порог, данной работе порог равен 0.08. После того как все тексты были переведены в экземпляры класса WText, была произведена оценка кол-ва текстов с одинаковым кол-вом определяющих слов. Такое распределение приведено на (Рис. 4.11):

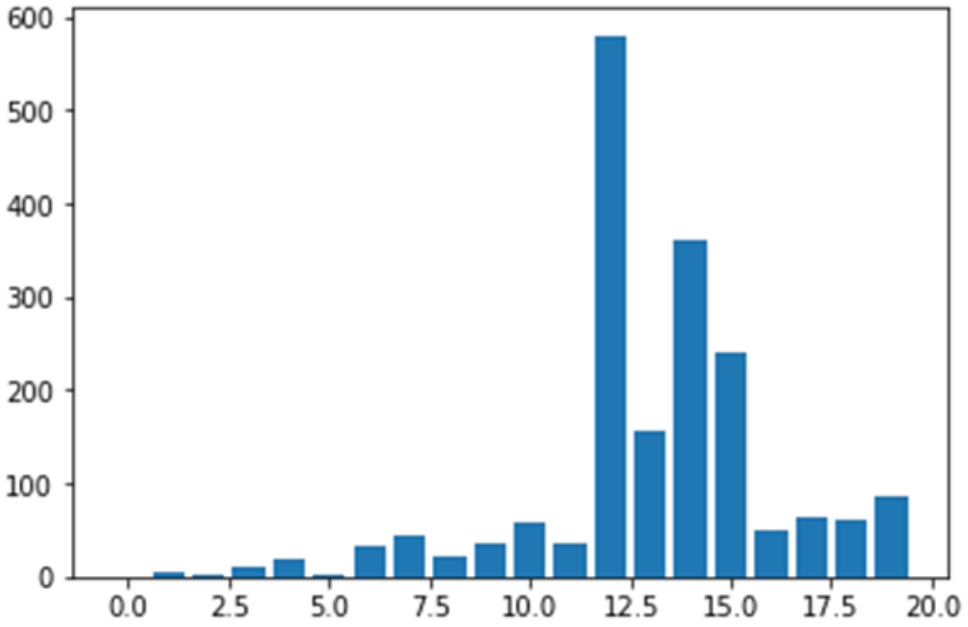

Рис. 4.11 – Распределение количества ключевых слов на текст

Исходя из картинки видно, что подавляющее большинство текстов имеют более 10 определяющих слов. Каждому определяющему слову ставится в соответствие экземпляр класса VAAction, таким образом из текстов набираются сценарии из 10 последовательных действий. Чтобы было удобнее формировать данные для обучения модели, определяющей поведение виртуальных акторов, реализуется метод получения последовательности состояний виртуальных агентов так, что переход в каждое последующее состояние.

4.4 Реализация модели поведения виртуального актора

По полученным размеченным данным на основании семантической близости слов выделенных и сопоставленных с заранее описанными действиями для акторов были получены следующие результаты. Суть первого эксперимента, состояла в предсказании 50 действий производимых акторами, результаты эксперимента представлены на (Рис. 4.12), (Рис. 4.13):

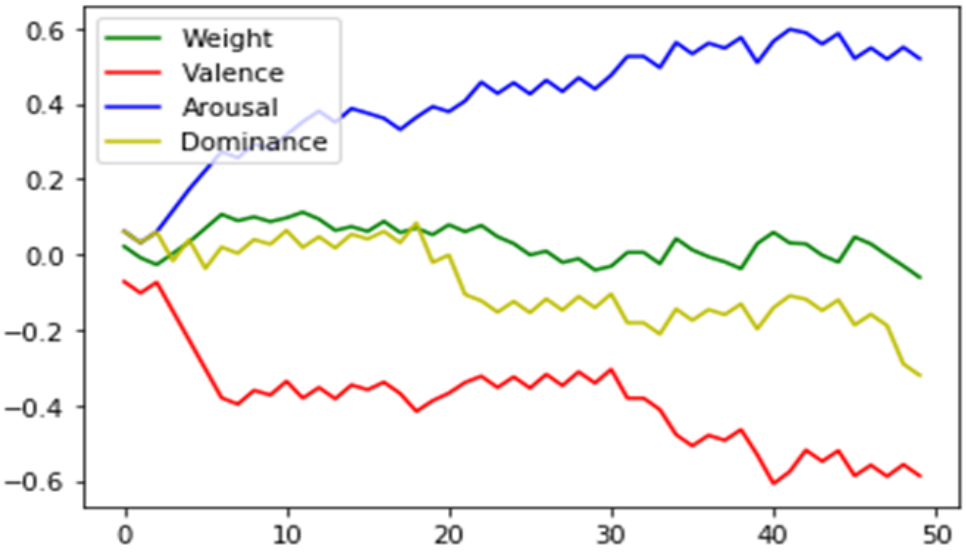

Рис. 4.12 – Оценки виртуального актора1

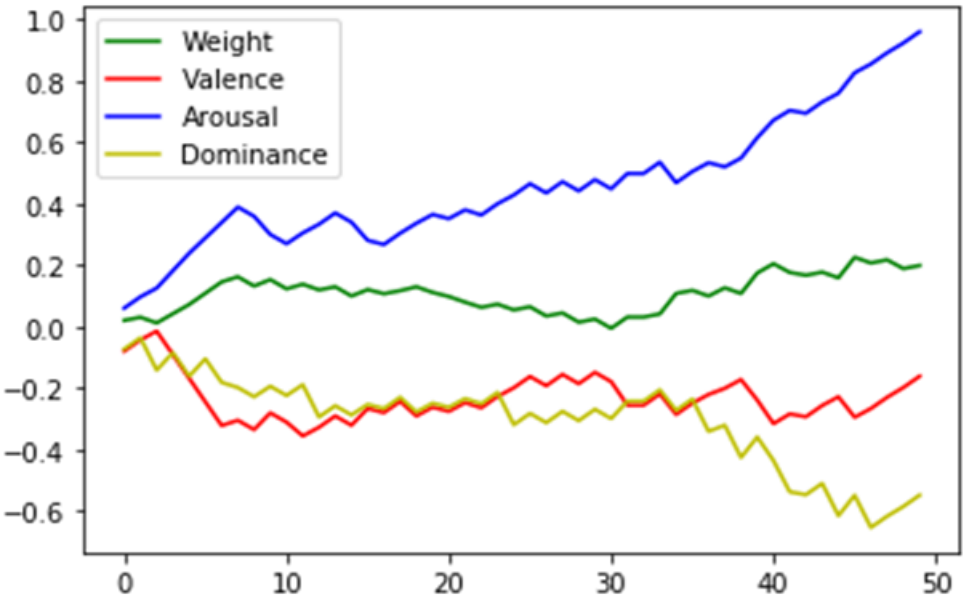

Рис. 4.13 – Оценки виртуального актора2

В следующем эксперименте было проведено 10000 действий и была построена гистограмма распределения частоты действий выполняемых первым виртуальным актором., результаты представлены на (Рис. 4.14):

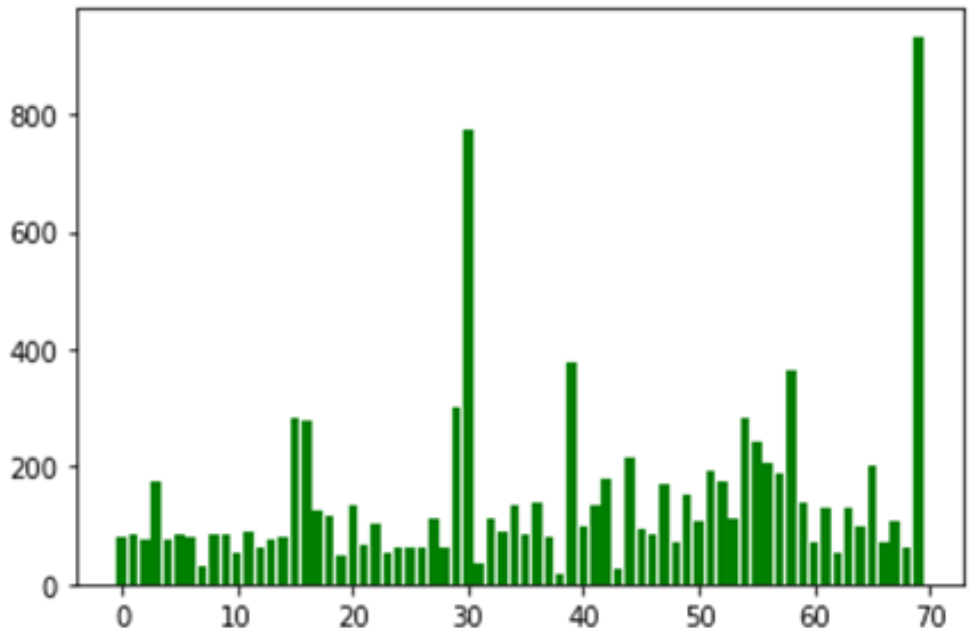

Рис. 4.14 – Частоты действий акторов

4.5 Выводы

В данном разделе осуществляется описание реализации получения и разметки данных с ресурсов содержащих юмористические сюжеты. А так же последующим обучением модели, генерирующей поведение виртуального агента.

Была разработано программное обеспечение, позволяющее определять семантическую принадлежность текста. А так же последующее определение семантической близости слов с заранее описанными действиями для виртуальных акторов. Группы действий которых впоследствии используются для обучения нейронной сети для последующей генерации юмористических сюжетов.

Заключение

В рамках представленной работы были изучены проблемы парсинга сайтов и классические методы их решения, были выделены основные нейросетевые подходы для анализа текстов, выделены основные проблемы связанные с семантитеческой классификацией, а так же изучены библиотеки и плагины, которые применяются для реализации данных подходов. Был применен метод опорных векторов (или SVM – Support Vector Machine) для определения семантитеческой принадлежности текста к юмористическимо сюжету, а так же, оптимизирован алгоритм поиска требуемых данных на юмористических ресурсах.

По заврешению получения юмористических данных в процессе парсинга, были получены данные, которые в последствии были размечены по принципу определения близости слов с заранее описанными действиями для виртуальных акторов.

Размеченные данные были использованы в последствии для обучения модели задача которой генерировать юмористические сюжеты, что подтверждается результатами обучения.

1. Были определены подходы, согласно котором разрабатывалась альтернативная линия сюжетов и их воплощения Виртуальным Актором.
2. Был осуществлен сбор и осуществлена разметкуа данных для обучения нейронной сети;
3. Был осуществлен глубокий интеллектуальный анализ данных по размеченным массивам данных.
4. Был разработан, алгоритм передающей результаты анализа виртуальному Агенту.
5. Было Реализован визуальный агент и сцена, используя межплатформенную среду разработки компьютерных игр Unity3d.

Список литературы

1. *V. S. A.* Comparative analysis of implemented cognitive architectures //Biologically Inspired Cognitive Architectures 2011. – IOS Press. — 2011.
2. *V. S. A.* Emotional biologically inspired cognitive architecture //Biologically Inspired Cognitive Architectures. — 2013.
3. *P. I. E.* Emotions and feelings. — 2007.
4. *V. S. A.* Socially emotional brain-inspired cognitive architecture framework for artificial intelligence //Cognitive Systems Research. — 2020.
5. *Langley P. Laird J. E. R. S.* Cognitive architectures: Research issues and challenges //Cognitive. — 2009.
6. *V. S. A.* Emotional biologically inspired cognitive architecture //Biologically Inspired Cognitive. — 2013.
7. *Bai S. Kolter J.Z. K. V.* An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. — 2018.
8. *ресурс] W. [* Рекуррентная нейронная сеть. — 2020.
9. *H. Y. W. K. K. Y. M. S.* Comparative study of CNN and RNN for natural language processing. 2017. arXiv preprint.
10. *И.А. Батраева А.Д. Нарцев А. Л.* ИСПОЛЬЗОВАНИЕ АНАЛИЗА СЕМАНТИЧЕСКОЙ БЛИЗОСТИ СЛОВ ПРИ РЕШЕНИИ ЗАДАЧИ ОПРЕДЕЛЕНИЯ ЖАНРОВОЙ ПРИНАДЛЕЖНОСТИ ТЕКСТОВ МЕТОДАМИ ГЛУБОКОГО ОБУЧЕНИЯ. — 2020.
11. *Standard L.* HTML Standard - whatwg. —.
12. *П. Ф.* Машинное обучение. Наука и искусство построения алгоритмов, который извлекают знания из данных. — 2015.
13. *Y. K.* Convolutional neural networks for sentence classification // Proc. of the 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP 2014). — 2014.
14. *Селезнев К., Владимиров А.* Лингвистика и обработка текстов // Открытые системы. — М., 2013.

15. *Mirkin B.* Core Concepts in Data Analysis: Summarization, Correlation and Visualisation, DOI. — 2011.

16. *Manning C., Schuetze H.* Foundations of Statistical Natural Processing. MIT. — M., 1999.

17. *Weisfeld. M.* The Object-Oriented Thought Process. — Fourth Edition. — Addison-Wesley Professional. — 2013.

18. *Константин Симаков И. К.* Особенности очистки адресных данных // Открытые системы. СУБД. — 2013.

19. *Ильвовский Д., Черняк Е.* Системы автоматической обработки текстов // Открытые системы. СУБД. — М., 2014. — URL: https://www.osp.ru/os/2014/01/13039687.

20. *H. Y. W. K. K. Y. M. S.* Comparative study of CNN and RNN for natural language processing. 2017. arXiv preprint. — 2010.

21. *Н. К. В.* Элементы теории ассоциативной семантики // Управление большими системами. — 2012.

22. *М. Л. Ю.* Люди и знаки. — 2010.

23. *MyStem. Я. технология.* MyStem. —. — URL: https://tech.yandex.ru/mystem.

24. *Xin. R.* Word2vec parameter learning explained. 2014. arXiv preprint arXiv: 1411.2738. — 2010.