



**Факультет Кибернетики и Информационной безопасности  
КАФЕДРА КИБЕРНЕТИКИ (№ 22)**

Направление подготовки 09.04.04 Программная инженерия

**Пояснительная записка**

к научно-исследовательской работе студента на тему:

**Модернизация модели виртуального Агента с целью добавления  
эмоционального взаимодействия**

Группа \_\_\_\_\_ М20-504

Студент \_\_\_\_\_ Клычков М. Д.

Руководитель \_\_\_\_\_ Самсонович А. В.

Научный консультант \_\_\_\_\_

Оценка руководителя \_\_\_\_\_ Оценка комиссии \_\_\_\_\_

Члены комиссии

---

---

---

---

**Москва 2022**



**Факультет Кибернетики и Информационной безопасности  
КАФЕДРА КИБЕРНЕТИКИ (№ 22)**

**Задание на НИР**

Студенту гр. М20-504 Клычкову Матвею Дмитриевичу

**ТЕМА НИР**

**Модернизация модели виртуального Агента с целью добавления  
эмоционального взаимодействия**

**ЗАДАНИЕ**

№ п/п	Содержание работы	Форма отчетности	Срок исполнения	Отметка о выполнении Дата, подпись
1.	Аналитическая часть			
1.1.	Изучение и анализ существующих когнитивных архитектур	Пункт ПЗ		
1.2.	Изучить методы машинного обучения	Пункт ПЗ		
1.3.	Изучение и анализ проблемной области распознавания речи	Пункт ПЗ		
1.4.	Классификация и определение эмоций	Пункт ПЗ		
1.5.	Оформление расширенного содержания пояснительной записи (РСПЗ)	Текст РСПЗ	10.04.2022	
2.	Теоретическая часть			
2.1.	Описать модель работы виртуального актора			
2.2.	Разработка алгоритмов распознавания речи	Алгоритмы		
2.3.	Модификация модификация методов машинного обучения применительно к задаче распознавания речи	Алгоритмы		
3.	Инженерная часть			
3.1.	Построить модель тембора голоса	Модели		
3.2.	Построить семантическую модель распознавания голоса	Модели		
3.3.	Проектирование инструментов для анализа текста	Макеты		
3.4.	Проектирование приложения	Макеты		
3.5.	Результаты проектирования оформить с помощью UML диаграмм	UML диаграммы		
4.	Технологическая и практическая часть			
4.1.	Реализация модели виртуального актора	Исполняемые файлы, исходный текст		

4.2.	Реализация программного приложения	Исполняемые файлы, исходный текст		
4.3.	Реализация и дообучение моделей машинного обучения	Исполняемые файлы, исходный текст		
5.	Оформление пояснительной записи (ПЗ) и иллюстративного материала для доклада.	Текст ПЗ, презентация	06.05.2022	

## ЛИТЕРАТУРА

- Tikhomirova, D. V., Chubarov, A. A., Samsonovich, A. V. (2019). Empirical and modeling study of emotional state dynamics in social videogame paradigms. Cognitive Systems Research.
- Samsonovich A.V. Comparative Analysis of Implemented Cognitive Architectures// Biologically Inspired Cognitive Architectures. 2011. Vol. 233. P. 469-479. doi: 10.3233/978-1-60750-959-2-469
- Larue, O., West, R., Rosenbloom, P. S., Dancy, C. L., Samsonovich, A. V., Petters, D., Juvina, I. (2018). Emotion in the common model of 74 A.V. Samsonovich / Cognitive Systems Research 60 (2020) 57–76 cognition. Procedia Computer Science, 145, 740–746.
- Madl, T., Franklin, S., Chen, K., Trappl, R. (2018). A computational cognitive framework of spatial memory in brains and robots. Cognitive Systems Research, 47, 147– 172.
- Laird, J. E., Lebiere, C., Rosenbloom, P. S. (2017). A standard model of the mind: Toward a common computational framework across artificial intelligence, cognitive science, neuroscience, and robotics. AI Magazine, 38(4), 13–26./

Дата выдачи задания:

10.10.2021

Студент

Руководитель

\_\_\_\_\_

Клычков М. Д.

\_\_\_\_\_

Самсонович А. В.

# Реферат

Пояснительная записка содержит - страниц, - рисунков, - 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 с многомодальным интерфейсом, 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 ВКР 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 реализована система 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, основывающиеся на семантической 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

<b>Введение</b>	<b>4</b>
<b>1 Исследование существующих когнитивных архитектур и анализ их недостатков</b>	<b>5</b>
1.1 Изучение и анализ существующих когнитивных архитектур . . . . .	5
1.2 Изучение и анализ когнитивной архитектуры eBICA . . . . .	10
1.3 Нейронные сети и их типы . . . . .	12
1.4 Методы определения эмоций в речи . . . . .	14
1.5 Классификации и определение эмоций . . . . .	15
1.6 Выводы . . . . .	16
<b>2 Описание моделей, отвечающих за генерацию поведения виртуального актора</b>	<b>17</b>
2.1 Постановка задачи . . . . .	17
2.2 Описание работы модели актора . . . . .	18
2.3 Способы распознавания речи . . . . .	21
2.4 Семантический анализ текста и его виды . . . . .	32
2.5 Wav2vec . . . . .	34
2.6 kNN . . . . .	36
2.7 SVM . . . . .	39
2.8 MLP . . . . .	40
2.9 Выводы . . . . .	41
<b>3 Проектирование модели поведения виртуального агента</b>	<b>42</b>
3.1 Проектирование модифицированного прототипа Виртуального Актора . . . . .	42
3.2 Инструменты извлечения речевых признаков из речи . . . . .	44
3.3 Модель определения эмотивной составляющей в тексте . . . . .	46
3.4 Инструменты для семантического анализа текста . . . . .	46
3.5 Построение блок-схем и UML диаграмм . . . . .	49
3.6 Выводы . . . . .	52
<b>4 Реализация программного продукта</b>	<b>53</b>
4.1 Текущая версия реализованной модели . . . . .	53

4.2	Интеграция моделей из python в C# . . . . .	54
4.3	Реализация и дообучение модели . . . . .	55
4.4	Выводы . . . . .	56
<b>Заключение</b>		<b>57</b>
<b>Список литературы</b>		<b>58</b>
Список литературы . . . . .		60

## **Введение**

В последнее время все большую и большую популярность набирают технологии предоставляющие возможность участвовать человеку в виртуальном мире либо технические средства, позволяющие представление виртуальной реальность в реальном мире. Виртуальные Акторы способны в будущем заменить докладчика на конференциях различного характера и представлениях, быть интегрированы в процесс обучения студентов, быть использованы как развивающие игрушки для детей. Однако, до сих пор не существует реализованной модели, которая представляет из себя полноценный эмоциональный интеллект.

Целью исследования является создание общей вычислительной модели механизмов, лежащих в основе человеческих эмоций, дающих с возможность модели верно интерпретировать эмоции и давать на них обратную реакцию. Осуществляется это путем создания виртуального питомца, 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. Определяются типы распознавания и синтеза речи. Ставятся цели и задачи научно-исследовательской работы.

Во втором производится анализ когнитивных архитектур и производится анализ методов распознавания человеческой 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 интеллект, 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.

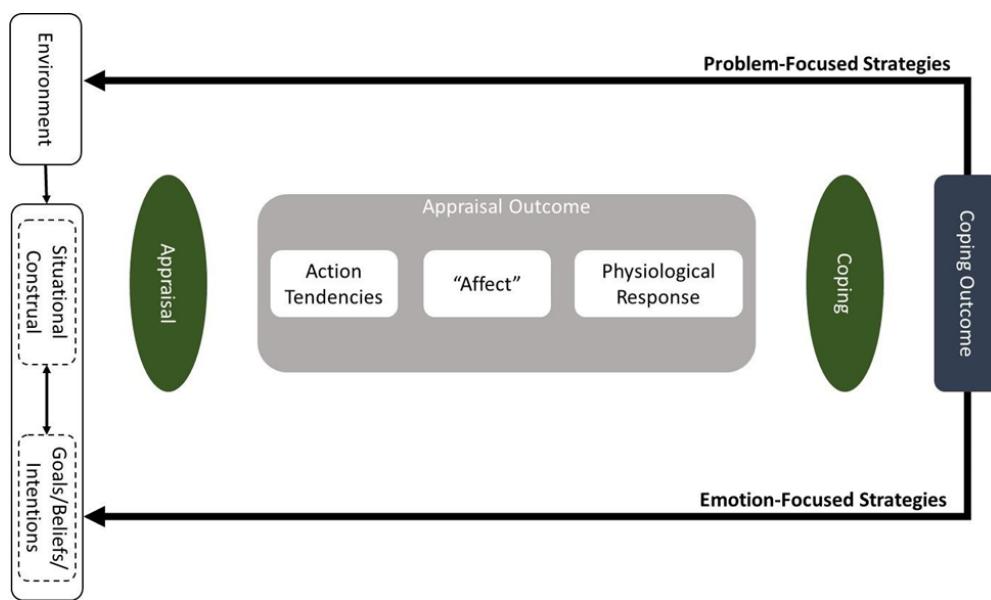


Рис. 1.1 – Когнитивно-мотивационно-эмоциональная система по материалам Smith and Lazarus.

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 совпадения: 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 неверbalного 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 на действие, 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.

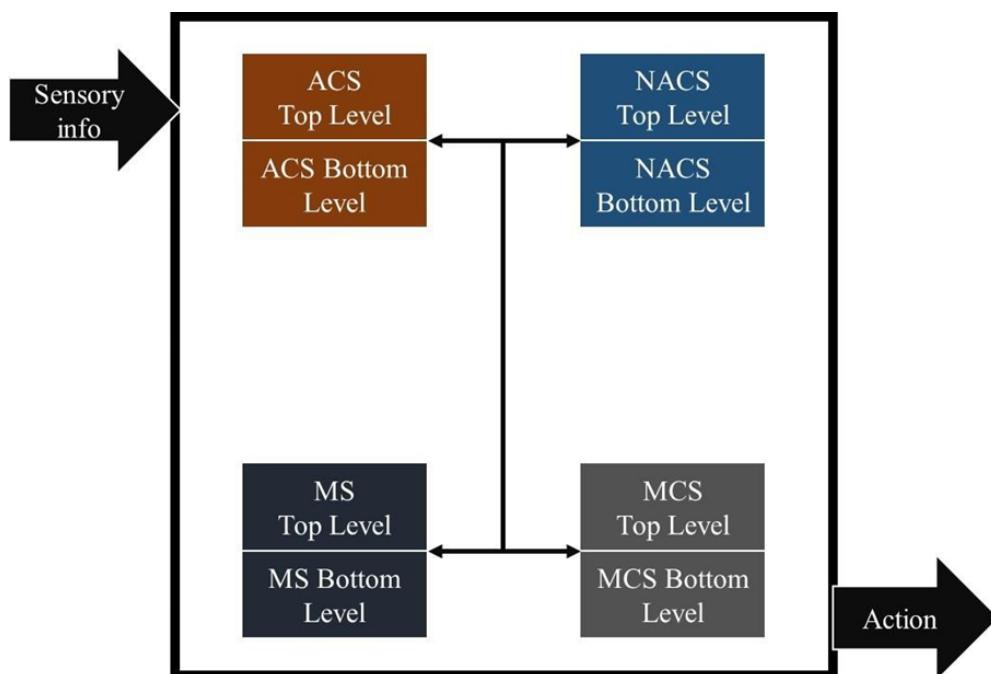


Рис. 1.2 – Подсистемы когнитивной архитектуры CLARION

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 работе [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 OCC.

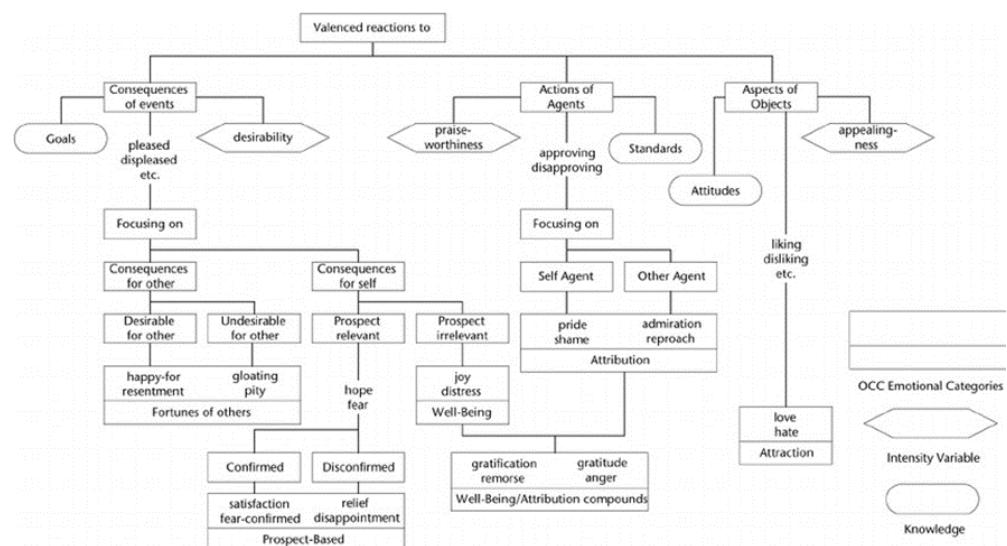


Рис. 1.3 – Оригинальная модель OCC

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 other), 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

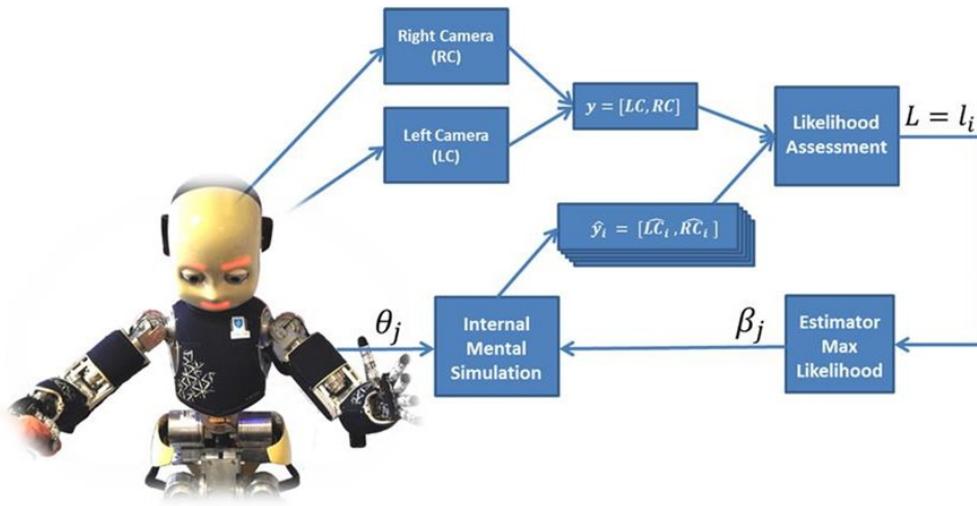


Рис. 1.4 – Схема работы iCub

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 измерений. 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].

(А) 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. (Б) Психическое состояние имеет оценку атрибут, который представляет собой эмоциональное состояние и самооценку агента в данный момент в данной ментальной перспективе.

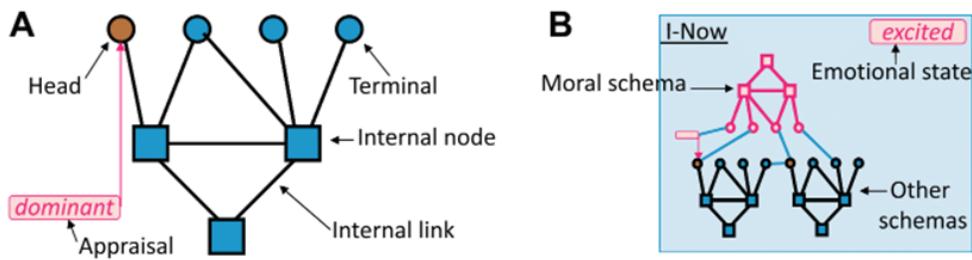


Рис. 1.6 – примеры эмоциональных элементов в рамках eBICA

Показанная ценность этой оценки «взволнована», что означает, что агент находится в возбужденном эмоциональном состоянии. Моральная схема, показанная в В, 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

Нейронная 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)
- Ограниченнная 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 Методы определения эмоций в речи

Для распознавания эмоций в речи используют следующие подходы:

- Анализ эмотивной сопровождающей текста в речи
- Анализ тональности речи

На текущий момент существуют следующие методы определения тональности текста:

1. Анализ текста методами векторного анализа (часто с применением n-граммных моделей), сравнение с ранее размеченным эталонным корпусом по выбранной мере близости и отнесение (классификация) текста к негативу или позитиву на основании полученного результата сравнения.
2. Поиск эмотивной лексики (лексической тональности) в тексте по заранее составленным тональным словарям (спискам паттернов) с применением лингвистического анализа. По совокупности найденной эмотивной лексики текст может быть оценен по шкале, отражающей количество негативной и позитивной лексики. Этот метод может использовать как списки паттернов, подставляемые в регулярные выражения, так и правила соединения тональной лексики внутри предложения
3. Смешанный метод (комбинация первого и второго подходов).

Первый метод (см., например, [Pang & al., 2002; Pang & al., 2005; Gamon, 2004]) работает достаточно быстро, но требует наличия предварительно размеченного эталонного корпуса, на основе которого происходит обучение алгоритма сравнения. Существенными недостатками такого подхода оказываются увеличение трудоемкости и ограничение разнородности корпуса (т. е. неполнота лексического покрытия), что приводит к потере точности. К тому же данный метод не позволяет провести глубокий анализ текста, то есть выявить и показать эмотивность на уровне предложения.

Второй метод [Nasukawa, 2003; Yi, 2003] не менее трудоемок в составлении тональных словарей (или получения списка тональных паттернов). Метод определения эмоций в текстах на русском языке 513 но в сочетании с синтаксическим и морфологическим анализом более гибок: он позволяет не только показать цепочки тональной лексики, но и получить синтаксически корректные эмоциональные выражения. При хорошем наполнении тональных словарных списков этот метод позволяет достичь хорошей полноты (покрытия эмотивной лексики). Недостаток этого метода в том, что с помощью него сложно дать количественную оценку негативности-позитивности текста. Чтобы избежать недостатков первого и второго метода, используют смешанный подход [Prabowo & al., 2009; Konig, 2006], частично включающий в себя два первых.

Следующий метод определения эмоций в речи это их распознавание, посредством акустического анализа.

Индивидуальность голоса обеспечивается сочетанием поведенческих и физиологических признаков. К поведенческим относят семантику, дикцию, произношение, ритм, интонации и др. Они обусловлены социальными фактами и могут быть довольно изменчивыми в зависимости от ситуации. Более надежными являются анатомические особенности речевого тракта, поэтому для работы автоматического распознавания наиболее адаптированы алгоритмы измерения акустических характеристик.

Акустическая теория речи рассматривает речевую волну как результат работы источника звука и фильтров. Подробное изложение о физиологических процессах речеобразования и моделях речевого тракта можно найти в книгах [2–4]. Здесь же кратко приведены только те параметры, которые участвуют в автоматическом распознавании дикторов. Характерные черты голоса конкретного человека в цифровой обработке сигналов получают через спектральный анализ речевой волны.

Частота первой гармоники спектра является частотой основного тона (основной частотой голоса). Частота основного тона  $F_0$  – обратная величина длительности  $T_0$  одного цикла работы голосовых связок:  $F_0 = 1/T_0$ . Основная частота определяет высоту голоса – ощущение, связанное с воздействием тона на слуховую систему человека. Индивидуальность данного параметра объясняется тем, что длительность  $T_0$  зависит от массы и упругости голосовых связок, а также от перепада давления над и под связками. Поэтому пол и возраст диктора оказывают влияние на значения основной частоты. Каждый человек имеет свой диапазон изменений частоты основного тона. Как правило, для взрослого он составляет от полутора до двух октав. В задаче распознавания личности по голосу необходимо определять базовую основную частоту, т. е. привычный и удобный для идентифицируемого человека режим работы голосовых связок.

## 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	Положительные, отрицательные, амбивалентные
Модальность	Радость, гнев, страх и др.
Влияние на поведение и деятельность	Осознаваемые, неосознаваемые
Предметность	Предметные, беспредметные
Степень произвольности	Произвольные, непроизвольные
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 изучены и проанализированы основные когнитивные архитектуры, особое внимание уделялось когнитивной архитектуре eBICA. Была рассмотрена проблема синтеза и распознавание речи. Были изучены материалы описывающие классификацию и определение эмоций.

## **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 обучения. 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.

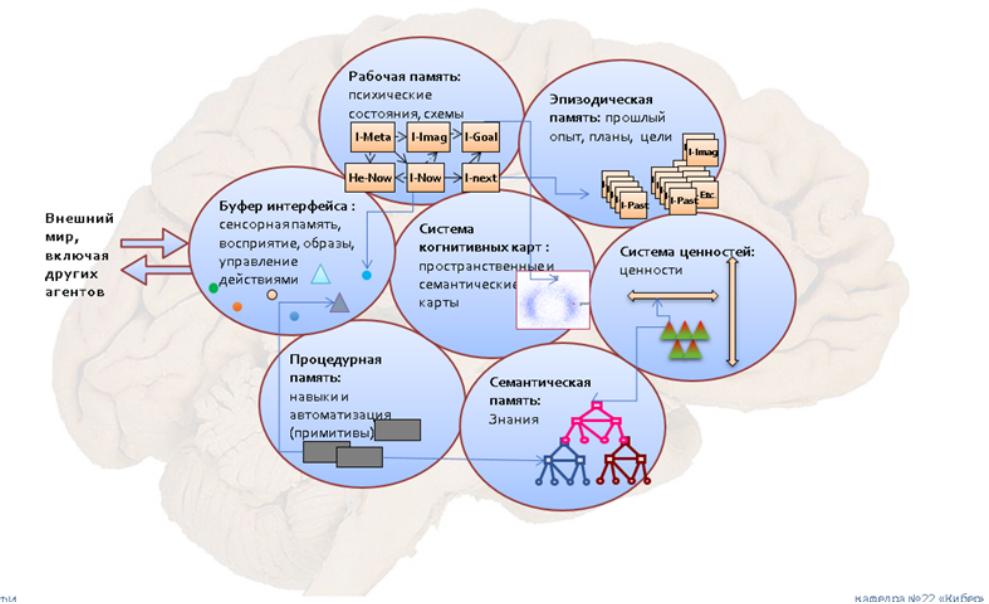


Рис. 2.1 – Структура когнитивной архитектуры eBICA

## 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 актера 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 актера.

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

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

- соматический;
- рациональный;
- когнитивный.

Нравственный фактор регулирует отношения первого актера 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:

$$Appraisals = (1 - r) * Appraisals + r * Action \quad (2.1)$$

где Appraisals - оценка, r - эмпирически вычисленная константа экспоненциального затухания, Action - оценка совершающего 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;
- valence – 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-

ложении 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:

$$Feelings = \beta * Appraisals \quad (2.2)$$

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  $r_1$  может быть отличным от  $r$ ):

$$Feelings = (1 - r_1) * Feelings + r_1 * (Appraisal - Feelings) \quad (2.3)$$

Соответственно значения Appraisals и Feelings как говорится в работе [11] 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 действие” имеется

в виде действие, основанное лишь на текущем состоянии мира и значений векторов Appraisals и Feelings акторов, отображенные 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 Способы распознавания речи

Сейчас существует несколько методов решения задачи распознавания речи в аудиозаписи. Успешность и скорость их работы во многом зависит от дикции автора, качества записи аудиофайла, поданного на вход, фонового шума, а также от качества работы спроектированной системы распознавания, и, как следствие, от выбранного метода.

Разнообразие особенностей речи, а также реализуемых подходов к решению задачи распознавания ведут к многообразию существующих решений. Все они борются за лидерство по качеству работы, но все еще имеют свои индивидуальные достоинства и недостатки.

Самыми популярными методами распознавания речи являются:

- Скрытое Марковское моделирование (HMM)
- Использование N-грамм
- Подход искусственного интеллекта
- Рекуррентные сети

До сих пор наиболее успешный [12] и часто используемый метод для распознавания речи

- это математическая модель, полученная на основе модели Маркова (Рис. 2.3). Скрытая Марковская модель представляет собой статистическую модель, моделирующую процесс Маркова с неизвестными параметрами. Пусть имеет  $N$  состояний модели. Каждое из состояний имеет свою вероятность наступления. Обозначим вероятности перехода между состояниями как матрицу  $A=\{a_{ij}\}$ , где  $a_{ij}$  – вероятность перехода из  $i$ -го в  $j$ -е состояние. В каждом из своих состояний система принимает одно из  $M$  значений какого-то своего параметра.

Обозначим вероятность выпадения каждого из  $M$  значений параметра в каждом из  $N$  состояний системы через  $B=\{b_j(k)\}$ , где  $b_j(k)$  – вероятность выпадения  $k$ -го значения параметра в  $j$ -м состоянии. Также введем вектор вероятности наступления начального состояния через  $\pi=\{\pi_i\}$ , где  $\pi_i$  – вероятность того, что в начальный момент система окажется в  $i$ -м состоянии. На рисунке 2.3 представлена схема строения скрытой Марковской модели. Состояния представлены в виде голубых кружков ( $N$  штук), значения скрытого параметра представлены с помощью желтых квадратов ( $M$  штук). Каждая стрелка символизирует возможный переход и имеет свой вес (значение вероятности). Таким образом, скрытой Марковской моделью называется тройка  $\lambda = \{A, B, \pi\}$  [13].

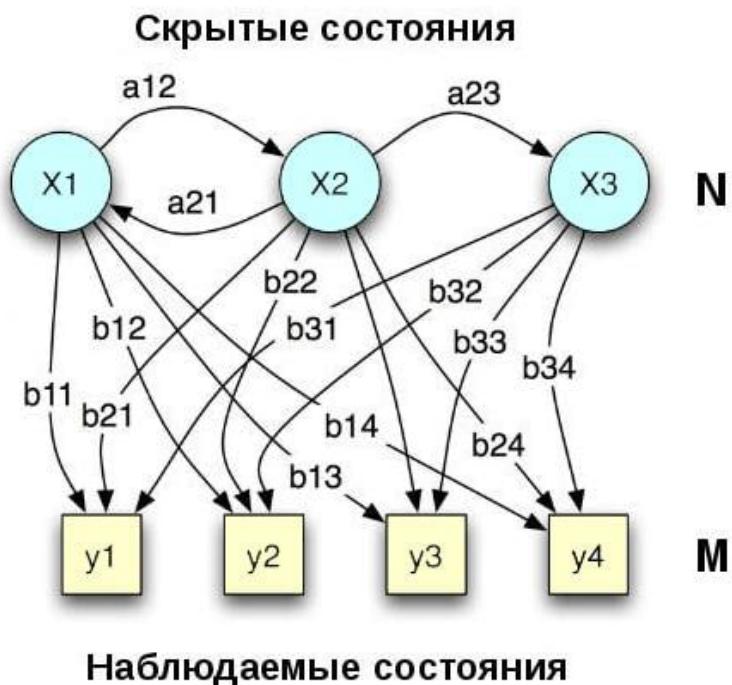


Рис. 2.3 – Общая схема устройства скрытой Марковской модели. [14]

Использование скрытых Марковских моделей для распознавания речи основано на трех допущениях [15]:

1. речь можно разбить на фрагменты (соответствующие состояниям в СММ) так, что внутри

каждого фрагмента речевой сигнал можно рассматривать как стационарный. Параметры речи в его пределах постоянны;

2. переход между состояниями осуществляется мгновенно;
3. вероятность каждого фрагмента зависит только от текущего состояния модели и не зависит от предыдущих состояний.

N-грамма – это модель представления данных, которая также может использоваться для распознавания речи [16]. N-граммная модель рассчитывает вероятность последнего элемента N-граммы при условии, что известны все предыдущие. В задаче распознавания речи элементами могут выступать фонемы, если задача решается на более низком уровне, или же целые слова.

При использовании этого подхода предполагается, что появление каждого слова (фонемы) зависит только от предыдущих слов (фонем), причем значение N указывает, от скольких предыдущих слов оно зависит.

Вероятность фразы можно вычислить как произведение вероятностей каждого из слов этой фразы:

$$P = P() \times P(|) \times P(|) \times P(|) \times P(|) \quad (2.4)$$

Здесь  $P(|)$  обозначает условную вероятность возникновения «самых» при условии появления «мой дядя». Чтобы определить  $P()$ , нужно посчитать, сколько раз это слово встретилось в тексте, и поделить это значение на общее число слов. Рассчитать вероятность  $P(|)$  уже сложнее. Чтобы упростить эту задачу, примем, что вероятность слова в тексте зависит только от предыдущего слова. Именно здесь проявляет себя выбранное значение N. Тогда формула для расчета фразы примет вид:

$$P = P() \times P(|) \times P(|) \times P(|) \times P(|) \quad (2.5)$$

Рассчитать условную вероятность  $P(|)$  несложно. Для этого считаем количество пар 'мой дядя', и делим на количество в тексте слова 'мой'. В результате, если мы посчитаем все пары слов в некотором тексте, мы сможем вычислить вероятность произвольной фразы. Этот набор рассчитанных вероятностей и будет биграммной моделью. Степень модели указывает на количество слов, учитываемых при расчете вероятностей. Рисунок 2.4 показывает несколько N-грамм для небольших значений N. Униграмма имеет степень 1, биграмма – степень 2, триграмма – степень 3.

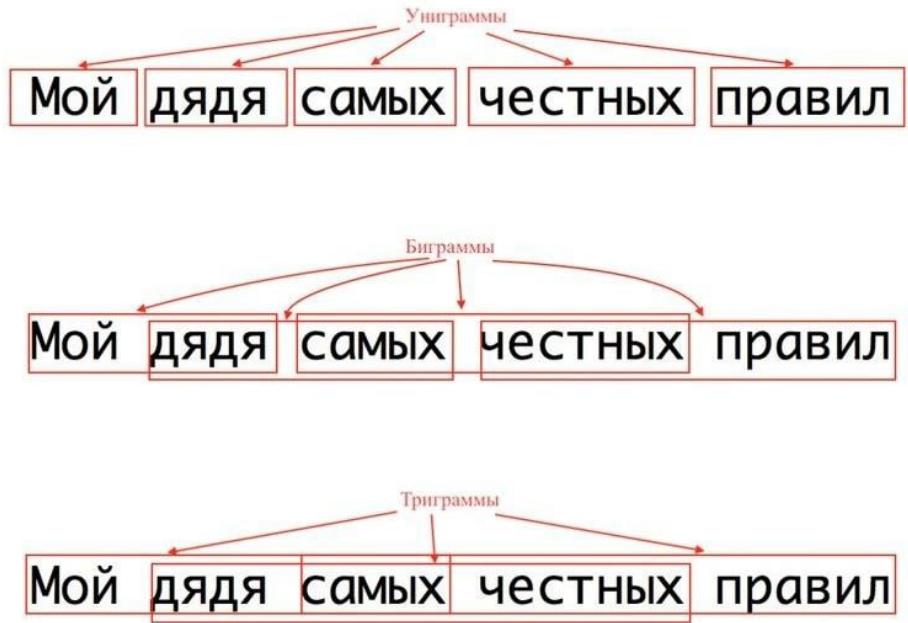


Рис. 2.4 – N-грамма для N=1,2,3

Представим фразу  $w$  как последовательность слов  $w = (w_1, w_2, w_3, \dots, w_k)$ . Тогда более формально N-граммная модель рассчитывает вероятности по формуле:

$$p(w) = \prod_{i=1}^{k+1} p(w_i | w_{i-1}) \quad (2.6)$$

$$w_{i-n+1}^{i-1} = (w_{i-n+1}, \dots, w_{i-1}) \quad (2.7)$$

Тогда формула вычисления вероятностей биграммы представляется в виде:

$$w_{i-n+1}^{i-1} = (w_{i-n+1}, \dots, w_{i-1}) \quad (2.8)$$

Подход с использованием методов машинного обучения один из самых часто используемых подходов. Для этого используют:

- Рекуррентные сети
- Двунаправленные рекуррентные сети
- Свёрточная нейронная сеть

Рекуррентные нейронные сети (RNN) - это класс нейронных сетей, обладающих искусственной внутренней памятью. Такие сети используются в случаях, когда важна сама последовательность данных и та временная динамика, которая соединяет данные. Последовательные данные – это, в основном, просто упорядоченные данные, в которых связанные объекты следуют друг за другом. Примерами могут служить тексты, финансовые данные, последовательность ДНК,

или данные временных рядов. Различные архитектуры рекуррентных сетей позволяют работать с разной необходимой глубиной памяти, в следствие чего спектр задач, в которых такие сети могут применяться, достаточно велик. Возможность учитывать порядок входных данных делает рекуррентные сети идеально подходящими для задач машинного обучения, связанных с обработкой текстовых данных.

В RNN информация проходит циклически и с каждым циклом может затухать – запоминать более новую информацию, обновляя память. Когда принимается решение, сеть учитывает текущие входные данные, а также то, что она узнала на предыдущих слоях. Рисунок 2.5 иллюстрирует разницу в потоке информации между RNN и обычной нейронной сетью.

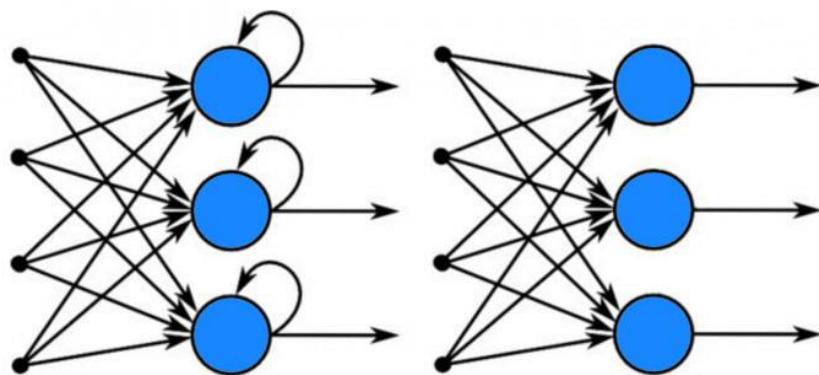


Рис. 2.5 – Сравнение структуры рекуррентной нейронной сети (справа) и сети прямого распространения (слева). [17]

Нейроны сети производят вывод, а также копируют этот вывод и зацикливают его обратно на свой вход. Так они получают информацию не только от предыдущего слоя, но и от самих себя предыдущего прохода. Это означает, что порядок, в котором подаются данные и обучается сеть, становится важным.

Большой проблемой при работе сетей RNN является проблема исчезающего градиента, которая заключается в быстрой потере информации с течением времени. Сети способны хранить только самую недавнюю информацию, что не всегда является достаточным для данной задачи. Допустим, мы хотим предсказать последнее слово в тексте “Я вырос в России... Мой родной язык русский”. Ближайший контекст предполагает, что последним словом будет называние языка, но, чтобы установить, какого именно языка, нам нужен контекст «России» из более отдаленного прошлого. Таким образом, разрыв между актуальной информацией и точкой ее применения может стать очень большим, что не всегда допустимо. В таких задачах необходимо уметь сохранять более глубокую память.

Сети с долгой краткосрочной памятью (Long Short Term Memory, LSTM) стараются решить

вышеупомянутую проблему RNN потери информации. Для этого в сети используются фильтры и клетки памяти.

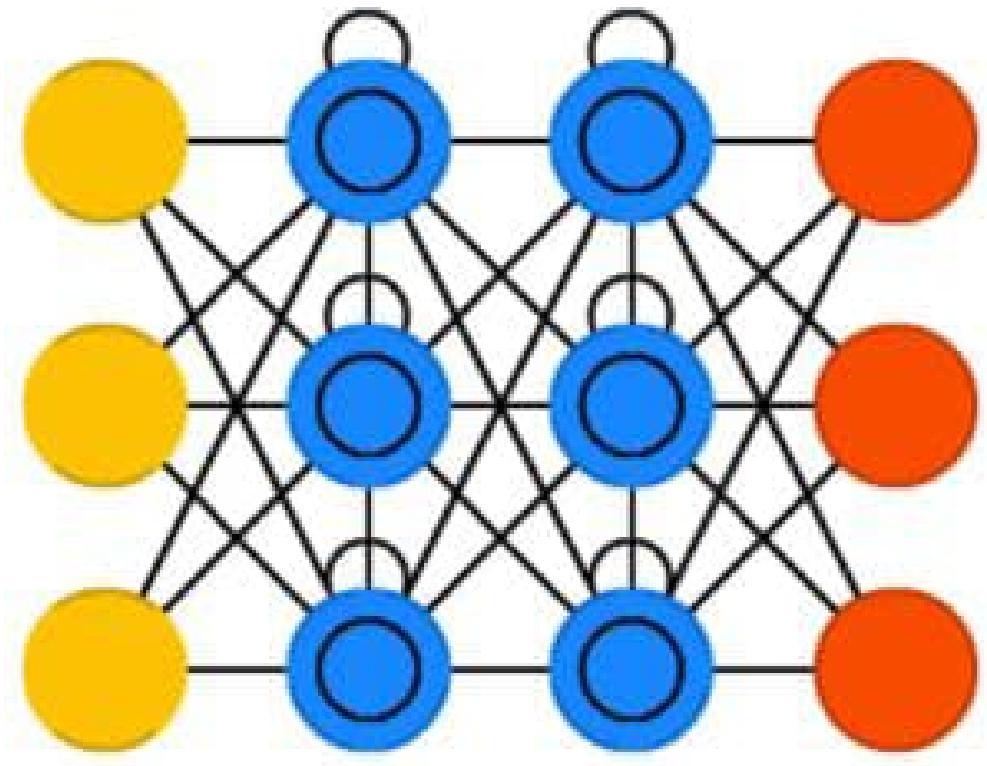


Рис. 2.6 – Схема строения рекуррентной нейронной сети с долгой краткосрочной памятью

Память в LSTM называется ячейками (синие нейроны на рисунке 2.6 имеют в себе черный круг, символизирующий внутреннюю ячейку памяти). Помимо стандартной для рекуррентных нейронов передачи данных - принимают в качестве входных данных текущий входной параметр и предыдущее состояние - они имеют специальное внутреннее строение, что отличает их от обычной рекуррентной сети. Внутри у каждой клетки памяти есть три фильтра: входной, выходной и забывающий, которые решают, какую память сохранить и какую стереть.

На рисунке 2.7 изображена ячейка LSTM в развертке. Обозначим  $\mathbf{x}$  - входное значение (голубой кружок),  $\mathbf{h}_t$  - выходной вектор (желтый кружок). Ячейка состоит из нескольких компонентов, разделенных на три группы по цвету. Входной фильтр (компоненты зеленого цвета) определяет, сколько информации из предыдущего слоя будет храниться в клетке. Выходной фильтр (компоненты розового цвета) определяет, сколько информации получат следующие слои. Ну и забывающий фильтр (компоненты фиолетового цвета) – какую часть информации можно отсечь. Целью этих фильтров является защита информации внутри. Затем они объединяют предыдущее состояние, текущую память и входной параметр.

Существуют различные модификации этой «классической» схемы ячейки. Сети LSTM способны научиться создавать более сложные структуры, чем классические рекуррентные сети.

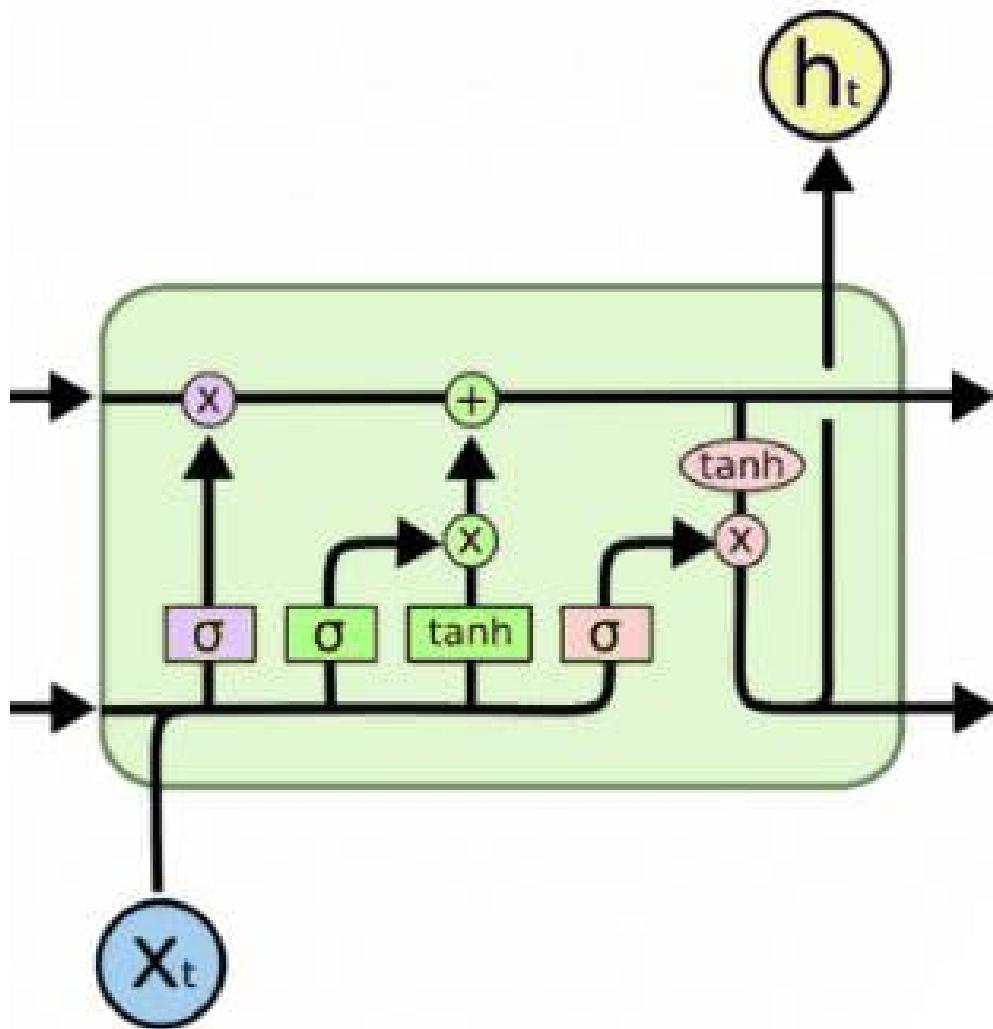


Рис. 2.7 – Ячейка LSTM в развертке

Решение задачи распознавания речи с использованием этой архитектуры было применено в исследовании [18].

Двунаправленные RNN (Bidirectional Recurrent Neural Networks, BRNN) основаны на той идее, что выход в момент времени  $t$  может зависеть не только от предыдущих элементов 15 в последовательности, но и от будущих. Другими словами, сеть будет рассматривать данные как две последовательности – в прямом и в обратном направлении. Двунаправленные рекуррентные нейронные сети (BRNN) соединяют два скрытых слоя, работающих в противоположных направлениях, с одним выходом, позволяя им получать информацию как из прошлых, так и из будущих состояний. BRNN разбивает нейроны классической рекуррентной сети на два направления, одно для прямых состояний (положительное направление времени), а другое для обратных состояний (отрицательное направление времени). Выходы из прямых состояний не связаны со входами обратных состояний, и наоборот. Это приводит к общей структуре, которую можно увидеть на рисунке ниже (сеть развернута на 4 временных шага).

Например, если нужно предсказать недостающее слово в середине последовательности (предложения), то нужно учитывать и левый, и правый контекст. Двунаправленная рекуррентная нейронная сеть представляет собой две рекуррентные сети, уложенные друг на друга. Без обратных состояний эта структура упрощается до обычной односторонней прямой RNN. Если прямые состояния исключены, получается классическая рекуррентная сеть с перевернутой временной осью.

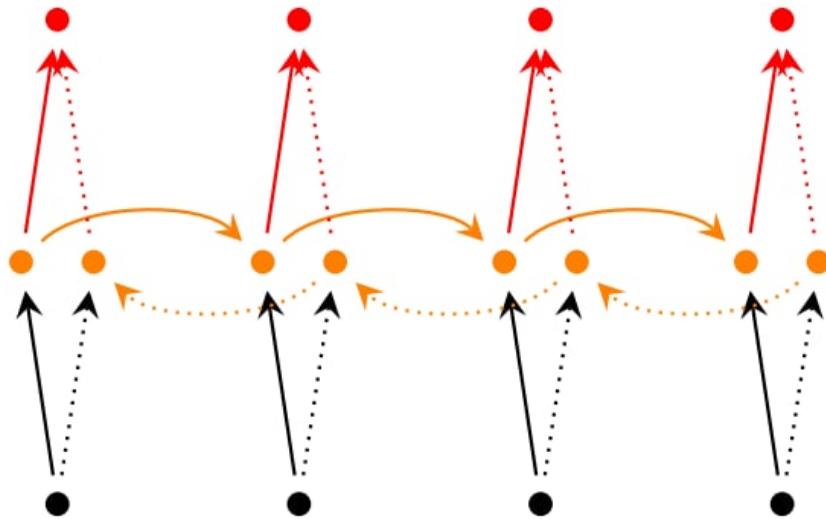


Рис. 2.8 – Общая структура двунаправленной рекуррентной нейронной сети.

На рисунке 2.8 изображена общая структура двунаправленной рекуррентной нейронной сети (BRNN). В процессе её обучения прямые и обратные состояния обрабатываются сначала в прямом проходе (на рисунке сплошной линией изображено положительное/прямое направление), выходные нейроны вычисляются последними (на рисунке пунктирной линией изображено отрицательное/обратное направление). При обратном проходе происходит обратное: сначала обрабатываются выходные нейроны, затем передаются состояния вперед и назад. Выход вычисляется на основе скрытого состояния обоих сетей RNN - веса обновляются только после завершения прямого и обратного проходов.

Двунаправленная рекуррентная нейронная сеть находит свое применение в задачи распознавания речи во многих успешных исследованиях [19].

В последнее время для решения задачи распознавания речи обретает популярность [20][21][22] метод, использующий свёрточные нейронные сети. Архитектура сети изображена на рисунке 2.9.

Для описания модели введем обозначения:

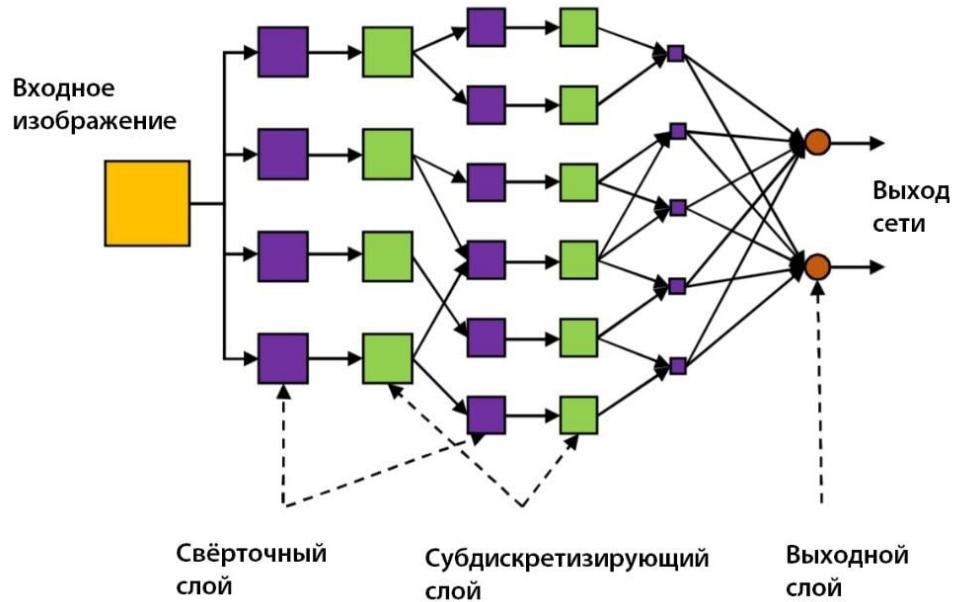


Рис. 2.9 – Архитектура свёрточной нейронной сети

- $l \in [1; L]$  - слой нейронной сети, где  $L = 2n + 2$  – количество слоев в сети
- $a \in Z^+$  - количество скрытых слоёв в сети
- $N^l$  - количество карт признаков (фильтров) на слое  $l$
- $y_n^l$  -  $n$ -ая карта признаков на слое  $l$

Карта признаков является тензором и представляет собой выход после каждого слоя сети. К примеру, при применении  $N$  ядер (свёрток) к одному входному изображению получим  $N$  карт признаков. Обычно происходит чередование свёрточных и субдискретизирующих (пуллинга) слоёв. Тогда свёрточные слои стоят на нечётных позициях  $l = 1, 3, \dots, 2n + 1$ , слои субдискретизации на четных позициях  $l = 2, 4, \dots, 2n$ .

Опишем устройство свёрточного слоя. Будем рассматривать слой  $l$ , где  $l$  принимается нечётным числом  $l = 1, 3, \dots, 2n + 1$ . Тогда для карты признаков  $n$  введем следующие обозначения:

- $w_{m,l}^l(i, j)$  – свёртка (ядро, фильтр), применяемая к карте признаков  $\otimes$  слоя  $(l - 1)$ , на слое  $l$  с картой признаков  $n$ ;
- $b_n^l$  – пороговые значения, которые присоединяются к карте признаков  $n$  на слое  $l$ ;
- $V_n^l$  – список всех карт признаков слоя  $(l - 1)$ , которые соединяются с картой признаков  $n$  слоя  $l$ .

Таким образом, карта признаков  $n$  свёрточного слоя  $l$  будет вычисляться следующим образом:

$$y_n^l = f_l(\sum_{m \in V_n^l} y_m^{l-1} \otimes W_{m,n}^l + b_n^l) \quad (2.9)$$

Где оператором  $\otimes$  обозначена математическая операция двумерной свёртки. Пример вычисления изображен на 2.10.

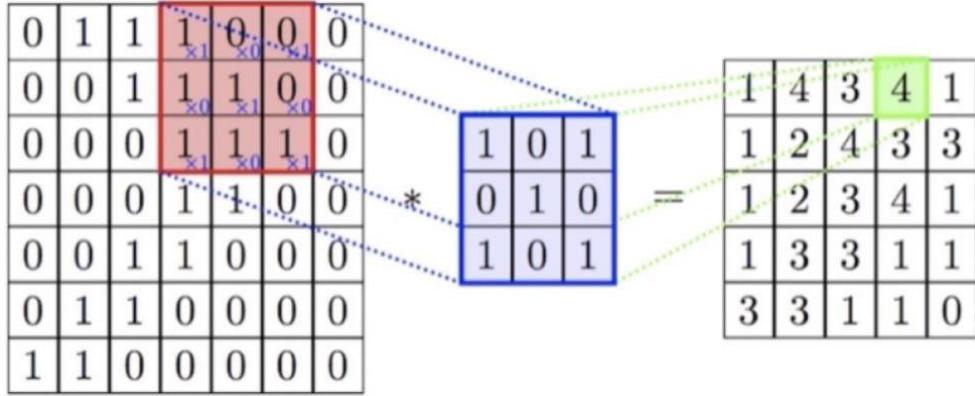


Рис. 2.10 – Двумерная свёртка карты признаков (слева) и ядра (голубая матрица). [23]

Предположим, что размер входных карт признаков  $y_m^{l-1}$  равен  $H^{l-1} \times W^{l-1}$ , а размер применяемой к ним свёртки  $w_{m,n}^l$  равен  $r^l \times c^l$ . Тогда размер выходной карты признаков  $\otimes\otimes\otimes$  вычисляется как:

$$(H^{l-1} - r^l + 1) \times (W^{l-1} - c^l + 1) \quad (2.10)$$

Подробная схема устройства слоя изображена на рисунке 2.11. Входная карта признаков представлена зеленым квадратом слева, свертка бежевым кругом, результат свертки – желтым квадратом. Затем, в случае многомерного фильтра (например, в случае работы с многоканальным изображением, когда каждый канал обрабатывается отдельно), результаты суммируются по формуле выше и применяется функция активации. Иначе – желтый квадрат уже является ответом.

Введём в рассмотрение субдискретизирующий слой (пуллинговый). Его основная цель – уменьшение размерности входной карты признаков. В свёрточной нейронной сети номер  $l$  субдискретизирующего слоя принято принимать чётный числом, то есть  $l = 2, 4, \dots, 2a$ .

Разделим карту признаков  $n(l-1)$ -ого слоя на непересекающиеся блоки размером 2x2 пикселя (Для простоты описания используется размер 2x2, однако на практике может использоваться и отличающийся размер). Затем просуммируем значения четырёх пикселей в каждом блоке и в результате получим матрицу  $z_n^{l-1} = z_n^{l-1} = \{z_n^{l-1}(i, j)\}$ . Её элементами будут являться соответствующие значения сумм. Таким образом, формула для вычисления имеет следующий вид:

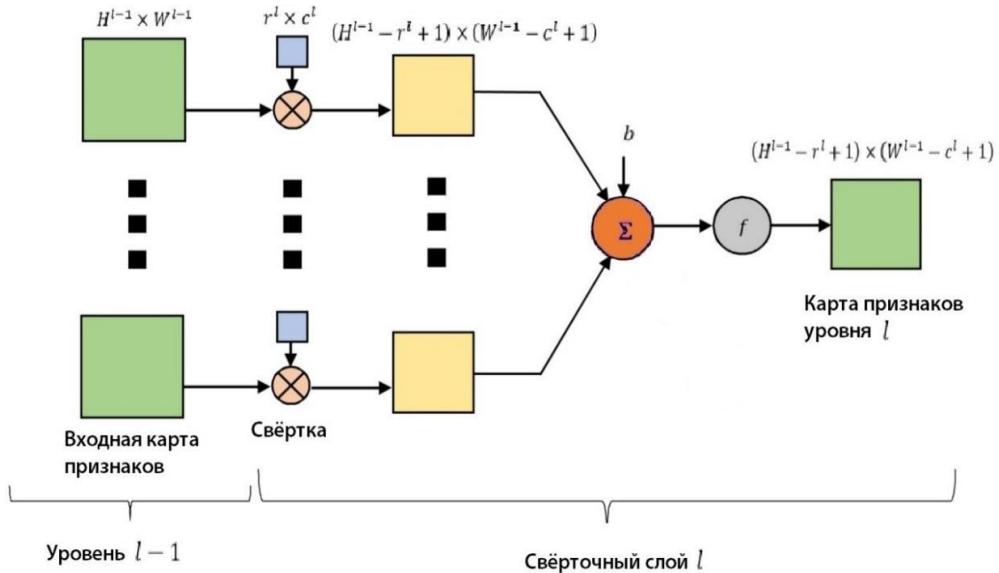


Рис. 2.11 – Схема свёрточного слоя  $l$

$$z_n^{l-1} = y_n^{l-1}(2i-1, 2j-1) + y_n^{l-1}(2i-1, 2j) + y_n^{l-1}(2i, 2j-1) + y_n^{l-1}(2i, 2j) \quad (2.11)$$

Картой признаков  $n$  субдискретизирующего слоя  $l$  будет являться полученная матрица  $z_n^{l-1}$ .

При этом вместо суммирования может использоваться любая другая функция (взятие максимума, взятие среднего, суммирование, разность и т.д.). Пример вычисления субдискретизирующего слоя с функцией взятия максимума и разбиением на блоки представлен на рисунке 2.12.



Рис. 2.12 – Вычисление пуллингового слоя с функцией максимума и блоками 2x2. [24]

Таким образом, размер  $H^l \times W^l$  карты признаков  $y_n^l$  субдискретизирующего слоя  $l$  при блоках 2x2 будет равен:

$$\begin{aligned} H^l &= \frac{H^{l-1}}{2} \\ W^l &= \frac{W^{l-1}}{2} \end{aligned} \tag{2.12}$$

Существуют модификации слоя субдискретизации. Например, при разбиении карты признаков на блоки, последние могут перекрываться. Таким образом, можно задавать шаг смещения следующего блока относительно предыдущего. Другая модификация заключается в добавлении дополнительной «пустой рамки» к карте признаков. Так, при выполнении операции над блоком, оригинальные (не пустые) ячейки карты признаков могут вносить больший вклад в результат. Также, из-за добавления дополнительных ячеек выходная карта признаков будет иметь тот же размер, что и входная.

Выходной слой имеет номер  $L = 2a + 2$ , и состоит из  $NL$  нейронов. Он представляет собой слой обычного многослойного персептрона. Формула для расчета значения выходного нейрона  $n$ :

$$y_n^L = f_l\left(\sum_{m=1}^{N^{L-1}} y_m^{L-1} \otimes W_{m,n}^L + b_n^L\right) \tag{2.13}$$

Где:

- $w_{m,n}^L$  – фильтр, применяемый к карте признаков  $\otimes$  последнего свёрточного слоя для получения перехода к нейрону  $n$  выходного слоя (матрица весовых коэффициентов);
- $b_n^L$  – пороговое значение, добавляемое к нейрону  $n$  (коэффициент сдвига).

Таким образом, выходом свёрточной нейронной сети является вектор следующего вида:

$$y = [y_1^L, y_2^L, \dots, y_{NL}^L] \tag{2.14}$$

Для применения свёрточных сетей в задаче распознавания речи первоначально используется спектральное представление звукового потока.

На основе полученного спектрального изображения (на рисунке 2.13 изображена звуковая дорожка и соответствующее ей спектральное представление) стандартная свёрточная сеть для обработки спектрального представления сигнала выдает готовый распознанный результат.

## 2.4 Семантический анализ текста и его виды

Помимо анализа текста посредством определения тональности текста, одним из типов анализа текста – является семантический анализ текстов.

Семантический анализ текста – одна из основных проблем как теории создания систем искусственного интеллекта, относящаяся к обработке естественного языка (Natural Language

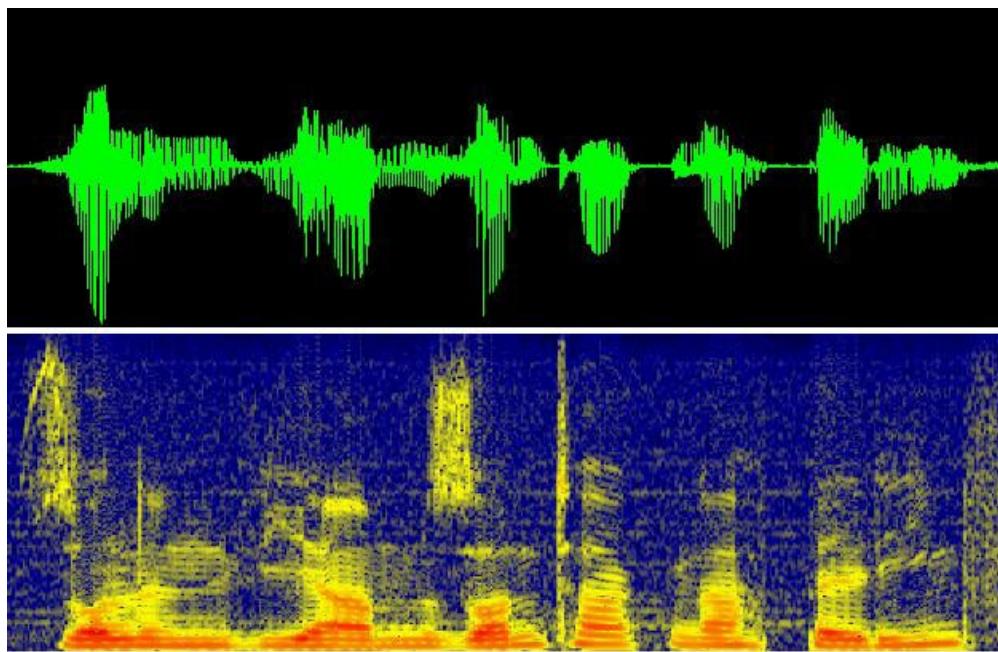


Рис. 2.13 – Изображение звуковой дорожки и соответствующего ей спектрального представления. Источник [25]

Processing, NLP), так и компьютерной лингвистики. В наш век компьютерных технологий чрезвычайно важно получать информацию на естественном языке и сделать так, чтобы эта информация могла перерабатываться с помощью компьютера [4: 208].

На текущий момент выделяют следующие модели семантического анализа текста:

1. Семантическая сеть
2. Фреймовые модели
3. Онтологическая модель
4. Тезаурус

Семантическая сеть – модель предметной области, имеющая вид ориентированного графа, вершины которого соответствуют объектам предметной области, а дуги (ребра) задают отношения между ними. Семантическая сеть появилась на основе математических формул. Семантическая сеть представляет собой схему, где есть концепты предметов, событий, состояний, с помощью линий показано отношение между концептами.

Фреймовые модели – это структура, которая нужна для того, чтобы описать понятия или ситуации, состоящая из характеристик этой ситуации и их значений. Фрейм можно считать элементом семантической сети.

Онтологическая модель – это детальное описание какой-то предметной или проблемной области, которое можно использовать для формулирования утверждений общего характера. Эта модель помогает создавать понятия, которые впоследствии становятся пригодны для ма-

шинной обработки.

Тезаурус – лексический словарь, в котором показаны семантические отношения между лексическими единицами. Благодаря этому словарю можно понять смысл, не только с помощью определения, но и соотнеся слова с другими понятиями и их группами, благодаря чему можно использовать искусственный интеллект для того, чтобы наполнить базы знаний. Обычно в тезаурусах используют такие семантические отношения как, синонимы, антонимы, гипонимы, гиперонимы, меронимы, холонимы и паронимы. WordNet можно привести как пример тезауруса. Синсет (синонимический ряд) является основной словарной единицей WordNet, он объединяет слова с похожими значениями. Синсеты это слова одной и той же части речи, что и слово, которое вы вводите. У каждого такого слова есть своя дефиниция, которая объясняет значение этого слова. Например, к слову ручка (pen) в таком словаре есть разные значения: crayon, pencil, marker.

Из вышеперечисленного можно сделать вывод, что в век компьютеризации, вопрос о семантическом анализе текста становится все более популярным. Из-за этого, область автоматической обработки текстов сфокусировалась на прикладном аспекте. Семантический анализ представляет собой одну из сложных математических задач, несмотря на востребованность практически во всех областях жизни современного человека. Главной задачей является сделать так, чтобы компьютер смог корректно трактовать образы, которые авторы текстов хотят передать читателям или слушателям.

## 2.5 Wav2vec

Текущие современные модели распознавания речи требуют больших объемов расшифрованного звука. Данные для достижения хороших результатов. В последнее время предобучение нейронных сетей стал эффективным методом для условий, где размеченных данных недостаточно. Ключевая идея состоит в том, чтобы изучить общие представления в установке, где имеется значительное количество размеченных или неразмеченных данных. Доступны и использовать изученные представления для повышения производительности в последующей задаче. для которых количество данных ограничено. Это особенно интересно для задач, где существенные требуются усилия для получения помеченных данных, таких как распознавание речи.

В данный момент принято использовать предварительное обучение для улучшения распознавания речи с учителем. Это позволяет использовать немаркованные аудиоданные, которые гораздо проще собрать, чем помеченные данные. Под предварительным обучением подразумевается факт обучения нейронной сети для задачи, в которой используется большой

объем данных. Это широко применялось в "computer vision", обработке естественного языка и, в последнее время, для определенных речевых задач.

Предварительное обучение бывает:

- контролируемым образом
- в неконтролируемом режиме

Предварительное обучение похожа на трансферное обучение, когда вы предварительно обучаете модель, зная что есть  $X$  и  $y$ . Но для неконтролируемого предварительного обучения вы изучаете представление речи. wav2vec — это сверточная нейронная сеть (CNN), которая принимает необработанный звук в качестве входных данных и вычисляет общее представление, которое может быть введено в систему распознавания речи. Целью является контрастная потеря, которая требует отличить истинный будущий звуковой образец от негативов.

Учитывая контекст входного сигнала цель состоит в том, чтобы предсказать следующую обсервацию из этого образца речи. Есть сопутствующая проблема, которая заключается в том, что следует иметь четкое представление как именно моделировать распределение  $p(x)$  для речевых примеров. Решение это проблемы заключается в том что следует понизить размерность образцов речи при помощи "сети кодировщика", а уже затем использовать контекстную сеть для прогнозирования следующих значений. wav2vec изучает представления аудиоданных, решая задачу прогнозирования контекста.

Конкретно  $x_i \in X$

- обучение первой сети кодировщика на основе CNN, которая мэпит  $X$  к  $Z$

$$f : X \rightarrow Z \tag{2.15}$$

- обучение второй контекстной сети, также основанной на CNN, которая отображает  $Z$  к одному контекстуальному тензору

$$g : Z \rightarrow C \tag{2.16}$$

Представление этих двух сетей содержится на рисунке 2.14:

Вот подробности имплементации нейросетей:

- кодировщик представляет собой 5-слойную CNN с размерами ядра  $(10, 8, 4, 4, 4)$  и шагами  $(5, 4, 2, 2, 2)$  и охватывает 30 мс звука. Слои имеют 512 каналов, слой групповой нормализации и нелинейность ReLU.

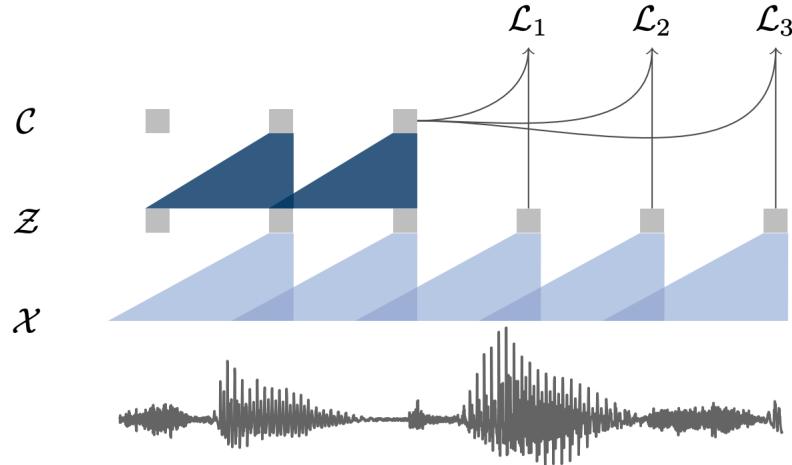


Рис. 2.14 – Представление двух нейросетей

- контекстная сеть имеет девять слоев с размером ядра три и шагом один, а общее рецептивное поле контекстной сети составляет около 210 мс. Слои имеют 512 каналов, слой групповой нормализации и нелинейный ReLU.

Модель учат распознавать истинный образец  $z_{i,k}$ ,  $k$  шагов в будущем, от распределения предложения  $p_n$ , называемой констрастной потерей определяемой:

$$L_k = - \sum_{i=1}^{T-k} (\log_\sigma(Z_{i+k}^t h_k(c_i)) + E_{z \sim p_n} [\log_\sigma(-Z^{\sim T} h_k(c_i))]) \quad (2.17)$$

Затем мы оптимизируем потери за несколько временных шагов:

$$L_k = \sum_{k=1}^K L_k \quad (2.18)$$

С целью получения ожидаемого распределения, отделяется условно десеть негативных примеров, выбирая одинаковые "триггер" факторы из этих негативных звуковых последовательностей. Другими словами, мы усредняемся десять равномерно выбранных значений из разных аудиосэмплов.  $\lambda$  - количество негативных примеров.

Предсказывая следующие шаги, мы выполняется задача, называемая самостоятельным обучением речи. Это также широко применяется в NLP и CV.

## 2.6 kNN

Метод  $k$ -ближайших соседей (англ. k-nearest neighbors algorithm, k-NN) – метрический алгоритм для автоматической классификации объектов или регрессии. Это один из самых простых алгоритмов классификации. Благодаря своей простоте, он является хорошим примером, с которого можно начать знакомство с областью Machine Learning. В данной статье рассмотрен

пример написания кода такого классификатора на python, а также визуализация полученных результатов.

Для задач классификации метка класса присваивается на основе большинства голосов, т.е. используется метка, которая чаще всего представлена вокруг данной точки данных. Хотя технически это считается «многочисленным голосованием», термин «мажоритарное голосование» чаще используется в литературе. Различие между этими терминологиями заключается в том, что для «голосования большинством» технически требуется большинство, превышающее 50%, что в первую очередь работает, когда есть только две категории. Когда у вас есть несколько классов, например, четыре категории, вам не обязательно нужно 50% голосов, чтобы сделать вывод о классе; вы можете присвоить метку класса при голосовании более 25%. Схема работы описана на рисунке 2.15.

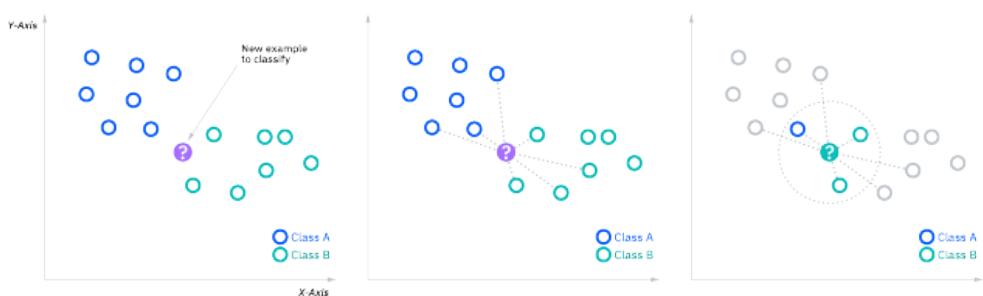


Рис. 2.15 – Диаграмма kNN

В задачах регрессии используется та же концепция, что и в задачах классификации, но в этом случае для предсказания классификации берется среднее значение к ближайших соседей. Основное отличие здесь в том, что классификация используется для дискретных значений, а регрессия – для непрерывных. Однако, прежде чем можно будет провести классификацию, необходимо определить расстояние. Чаще всего используется евклидово расстояние, о котором мы поговорим ниже.

Напомним, что целью алгоритма k-ближайших соседей является определение ближайших соседей данной точки запроса, чтобы мы могли присвоить этой точке метку класса. Для этого у KNN есть несколько требований:

Определите свои показатели расстояния

Чтобы определить, какие точки данных находятся ближе всего к заданной точке запроса, необходимо рассчитать расстояние между точкой запроса и другими точками данных. Эти метрики расстояния помогают формировать границы решений, которые разбивают точки запроса на разные регионы. Обычно вы видите границы решений, визуализированные с помощью диаграмм Вороного.

Хотя есть несколько мер расстояния, которые вы можете выбрать, в этой статье будут рассмотрены только следующие:

Евклидово расстояние ( $p = 2$ ): это наиболее часто используемая мера расстояния, и она ограничена векторами с действительными значениями. Используя приведенную ниже формулу, он измеряет прямую линию между точкой запроса и другой измеряемой точкой.

$$d(x, y) = \sqrt{\sum_{i=1}^n (y_i - x_i)^2} \quad (2.19)$$

Манхэттенское расстояние ( $p = 1$ ): это еще одна популярная метрика расстояния, которая измеряет абсолютное значение между двумя точками. Его также называют расстоянием такси или расстоянием до городских кварталов, поскольку оно обычно визуализируется с помощью сетки, иллюстрирующей, как можно перемещаться от одного адреса к другому по улицам города.

$$\text{M Distance} = d(x, y) = \left( \sum_{i=1}^m |x_i - y_i| \right) \quad (2.20)$$

Как и у любого алгоритма машинного обучения, у k-NN есть свои сильные и слабые стороны. В зависимости от проекта и приложения это может быть или не быть правильным выбором.

Плюсы данного подхода заключаются в следующем:

- Простота реализации: учитывая простоту и точность алгоритма, это один из первых классификаторов, который выучит новый специалист по данным.
- Легко адаптируется: по мере добавления новых обучающих выборок алгоритм подстраивается под любые новые данные, поскольку все обучающие данные сохраняются в памяти.
- Несколько гиперпараметров: для KNN требуется только значение  $k$  и метрика расстояния, что является низким по сравнению с другими алгоритмами машинного обучения.

Минусы же выражены в следующем:

- Плохо масштабируется: поскольку KNN является ленивым алгоритмом, он занимает больше памяти и места для хранения данных по сравнению с другими классификаторами. Это

может быть затратно как с точки зрения времени, так и денег. Больше памяти и хранилища повысит бизнес-расходы, а для обработки большего объема данных может потребоваться больше времени. Хотя для устранения неэффективности вычислений были созданы различные структуры данных, такие как Ball-Tree, в зависимости от бизнес-задачи может подойти другой классификатор.

- Алгоритм KNN имеет тенденцию становиться жертвой проблем размерности, что означает, что могут возникать проблемы с входными данными.

Однако минусы данного подхода могут быть нивелированы благодаря метода двойного расстояния, благодаря которому повышается точность распознавания. Тем самым kNN можно охарактеризовать как хороший метрический алгоритм.

## 2.7 SVM

В этой главе показана стратегия группировки звукоого знака с использованием новой методологии WOA-SVM и MapReduce. Система MapReduce выполняет анализ огромного объема информации (в данном случае аудиодокумента) с использованием преобразователя и редьюсера, где редуктор выполняет группировку с использованием ожидаемого классификатора WOA-SVM. Фундаментальные задачи по предложенному плану классификации аудио заключаются в следующем:

- Улучшение модели извлечения бликов с восемью новыми бликами внесло свой вклад, как и области повторения, за исключением двух бликов пространства коэффициентов, которые созданы для эффективной классификации при анализе звука.
- Знакомство нового WOA-SVM с групповым звуковым флагом путем продвижения параметров SVM с помощью процедуры WOA.
- Исполнения рекомендуемого алгоритма классификации WOA-SVM на этапе MapReduce, поскольку аудиозаписи имеют гигантские размеры и лимит.

SVM [26] уникален для наиболее часто используемых классификаторов. Основная идея SVM состоит в том, чтобы изолировать различные классы, используя гиперплоскости. SVM достиг высоких показателей точности, когда информация прямо размечена. Как бы то ни было, представление SVM не может отделить непрямо размеченную информацию. Эту проблему можно решить, используя функции ядра, которые используются для изменения информации, одержимой другим многомерным пространством; отныне информация может быть выделена напрямую. Выбор разумной функции ядра и изменение их ограничений – две основные трудности классификатора SVM. В этом сегменте будет представлено краткое описание идеи SVM в контексте группировки. Общая процедура работы алгоритма SVM пояснена на 2.16.

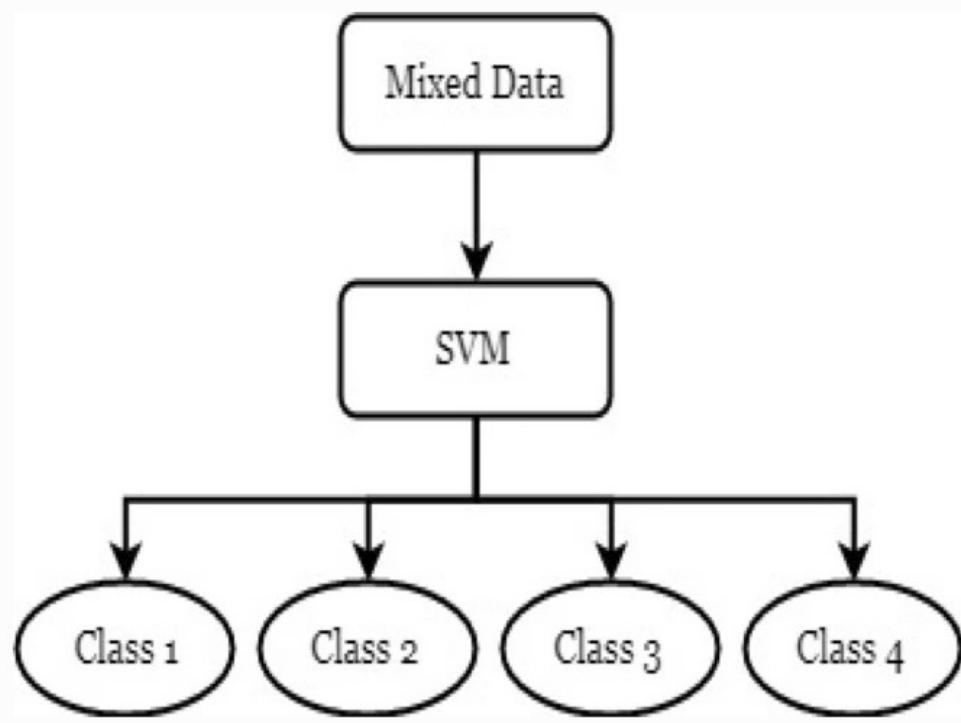


Рис. 2.16 – Базовы операции работы SVM

Даны  $N$  линейно разделимых обучающих выборок,  $X = \{x_1, x_2, \dots, x_N\}$  где  $x_i$  это "итый" тренировочный сэмпл и каждая модель имеет  $a$  характеристик в бинаризованных классах  $y \in \{\pm 1\}$ .

Линия  $w^T x + b = 0$  обозначает границу произношения между двумя модулями,  $w$  обозначает вектор весов,  $b$  представляет собой смещение, а  $b$  представляет собой предварительную модель.

Гиперплоскость делит Вселенную на два пространства. Цель состоит в том, чтобы найти оценку  $w$  и  $b$ , чтобы расположить гиперплоскость так, чтобы она находилась за пределами того, что многие считают возможным из ближайших тестов, например опорных векторов (SV), и построить две плоскости, Р1 и Р2.

## 2.8 MLP

Многослойный персептрон (MLP) представляет собой полно связанный класс искусственных нейронных сетей с прямой связью (ANN). Термин MLP используется неоднозначно, иногда в широком смысле для обозначения любой ИНС с прямой связью, иногда строго для обозначения сетей, состоящих из нескольких уровней персепtronов [27].

Обучение происходит в персептроне путем изменения веса соединения после обработки каждого фрагмента данных в зависимости от количества ошибок в выходных данных по сравнению с ожидаемым результатом.

Мы можем представить степень ошибки в выходном узле  $e_j(n) = d_j(n) - y_j(n)$ , где  $d$  целевое значение, а  $y$  - значение, создаваемое персепtronом.

$$\varepsilon(n) = \frac{1}{2} \sum_j e_j^2 \quad (2.21)$$

Используя градиентный спуск, изменение каждого веса равно:

$$\Delta w_{ji}(n) = -\eta \frac{\partial \varepsilon_n}{\partial v_i(n)} y_i(n) \quad (2.22)$$

Где  $y_i$  это выход предыдущего нейрона, а  $\eta$  – скорость обучения, которая выбирается таким образом, чтобы веса быстро сходились к ответу без колебаний.

Вычисляемая производная зависит от индуцированного локального поля  $v_j$ , которое само изменяется. Легко доказать, что для выходного узла эту производную можно упростить до

$$-\frac{\partial \varepsilon(n)}{\partial v_j(n)} = e_j(n) \phi'(v_j(n)) \quad (2.23)$$

где  $\phi'$  – производная описанной выше функции активации, которая сама по себе не меняется. Анализ более сложен для изменения весов в скрытом узле, но можно показать, что соответствующая производная равна

$$-\frac{\partial \varepsilon(n)}{\partial v_j(n)} = \phi'(v_j(n)) \sum_k -\frac{\partial \varepsilon(n)}{\partial v_k(n)} \omega_{kj}(n) \quad (2.24)$$

Это зависит от изменения веса  $k$ -х узлов, которые представляют выходной слой. Таким образом, чтобы изменить веса скрытого слоя, веса выходного слоя изменяются в соответствии с производной функции активации, и поэтому этот алгоритм представляет собой обратное распространение функции активации.

## 2.9 Выводы

Было изучено описание работы модели виртуального актора. Изучены подходы и методы применительно к задаче распознавания речи. Были рассмотрены ключевые подходы машинного обучения, которые будут использоваться в работе, а именно рекуррентные нейронные сети

### **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 Актера. В начале будет описан алгоритм по выбору ответного действия Виртуального Актера на действие человека.

На вход поступает действие человека, целью которого является влияние на Виртуального Актера. В зависимости от характеристики действия пересчитываются оценки Appraisals человека и Виртуального Актера 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 Актером. 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 Инструменты извлечения речевых признаков из речи

В связи с задачей эмоционального взаимодействия пользователя с виртуальным актором ставится задача распознавания 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 дата-сет для обучения данной нейросети. В большинстве работ по классификации эмоций присутствующих в речи человека используются открытые данные RAVDESS. Этот дата-сет включает в себя 7356 аудио записей с эмоциональным наполнением.

Область распознавания речи и ее эмоциональной 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):

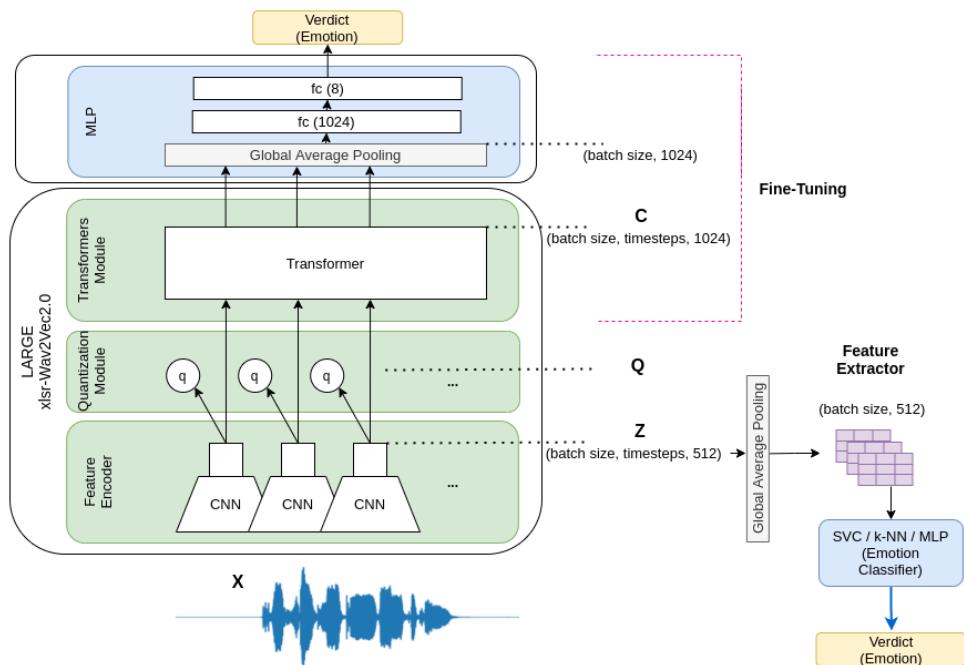


Рис. 3.1 – Предлагаемые конвейеры для распознавания речевых эмоций.

SVM с ядром «RBF» - является одной из наиболее популярных методологий обучения по precedентам, 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 используют нелинейную функцию активации.

### **3.3 Модель определения эмотивной составляющей в тексте**

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.

В современных системах автоматического определения эмоциональной оценки текста чаще всего используется одномерное эмотивное пространство: позитив-негатив, то есть хорошо-плохо.

Поэтому для упрощения будем классифицировать тексты по шкале:

1. негативная оценка
2. позитивная оценка

Автоматическое определение тональности текста подразумевает выделение тех фрагментов текста, которые выражают позитивную или негативную эмоциональность по отношению к объекту эмоциональной оценки (объекту тональности). Объект эмоциональной оценки может быть задан как один в целом для текста (с учетом его синонимических и анафорических употреблений), так и определяться в предложениях как любое имя собственное или даже нарицательное.

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 диалогов. Поэтому будет взята модель анализа тональности текста с использованием правил объединения слов в цепочки и определения тональности у объекта на основе предикационных отношений в пропозиции и дообучена на диалоговых данных, которые будут собраны посредством фильтрации диалогов из публично доступных данных с сохранением разметки.

### **3.4 Инструменты для семантического анализа текста**

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

Возможность идентификации семантической близости между словами сделала модель word2vec широко используемой в NLP-задачах, которые подробно описываются в [28]. 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 [29].

Таким образом, извлечение семантических отношений (отношение синонимии, родственные 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 [30].

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  $\text{vec('Madrid')} - \text{vec('Spain')} + \text{vec('France')}$  будет ближе к  $\text{vec('Paris')}$ , чем к другим векторам из распределения. Таким образом, разница векторов может быть использована для поиска семантических отношений между словами [31].

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. Бессспорно, 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 [32]. Данный пакет позволяет скачивать данные с веб ресурсов посредством GET запроса.

Для того чтобы модель word2vec, описанная в [33] 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. 3.2):

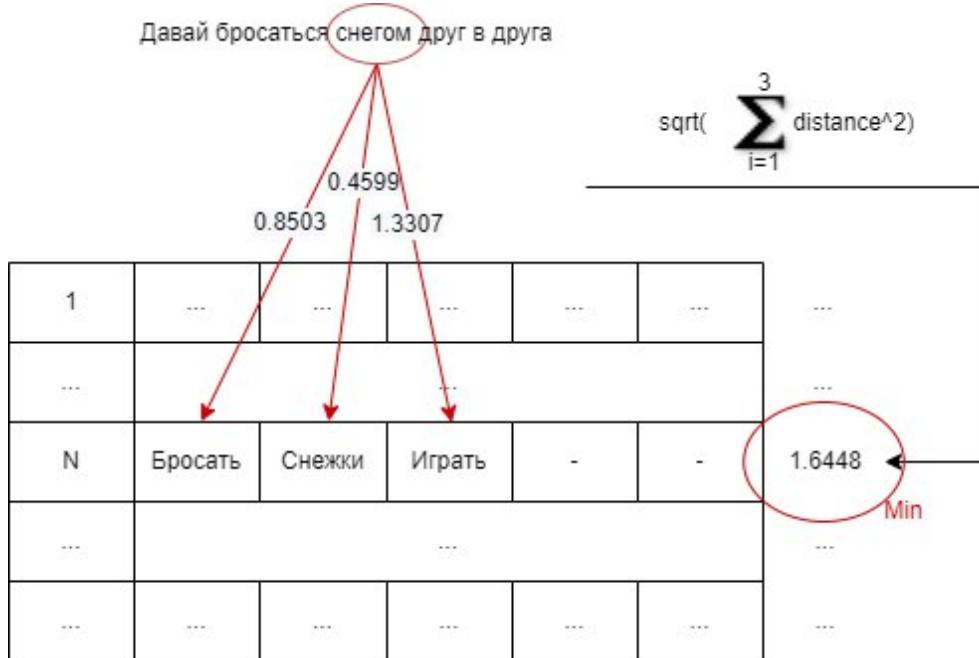


Рис. 3.2 – принцип работы word2vec

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 разобъем 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 [34].

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 итера-

цию 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 [35]). 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.

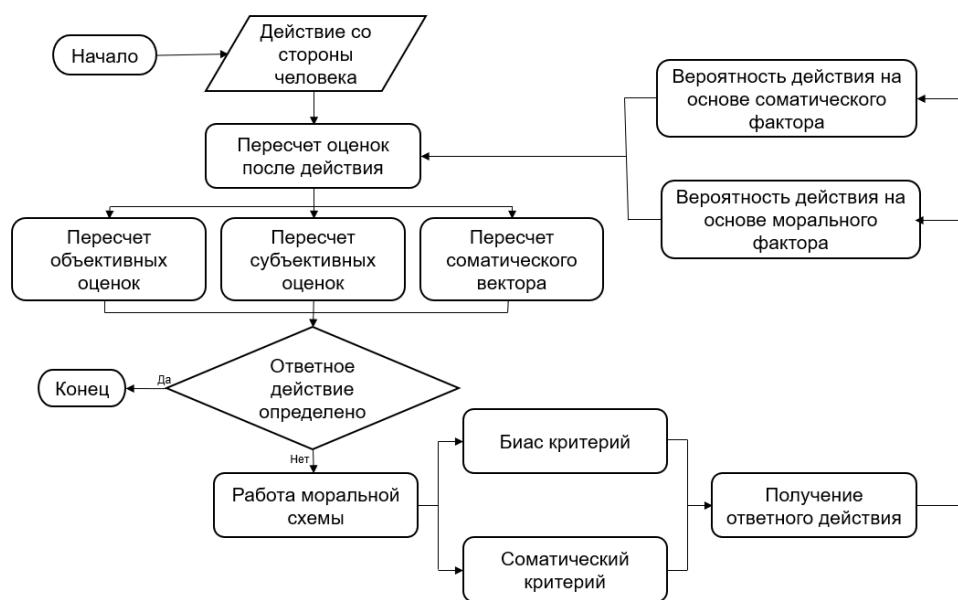


Рис. 3.3 – Блок-схема общей работы алгоритма

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 соответственно. Далее на основе данных вероятностей определяется итоговое действие, которое будет выполнено пингвином. Данное действие определяется наибольшей вероятностью.

рассмотрим 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.

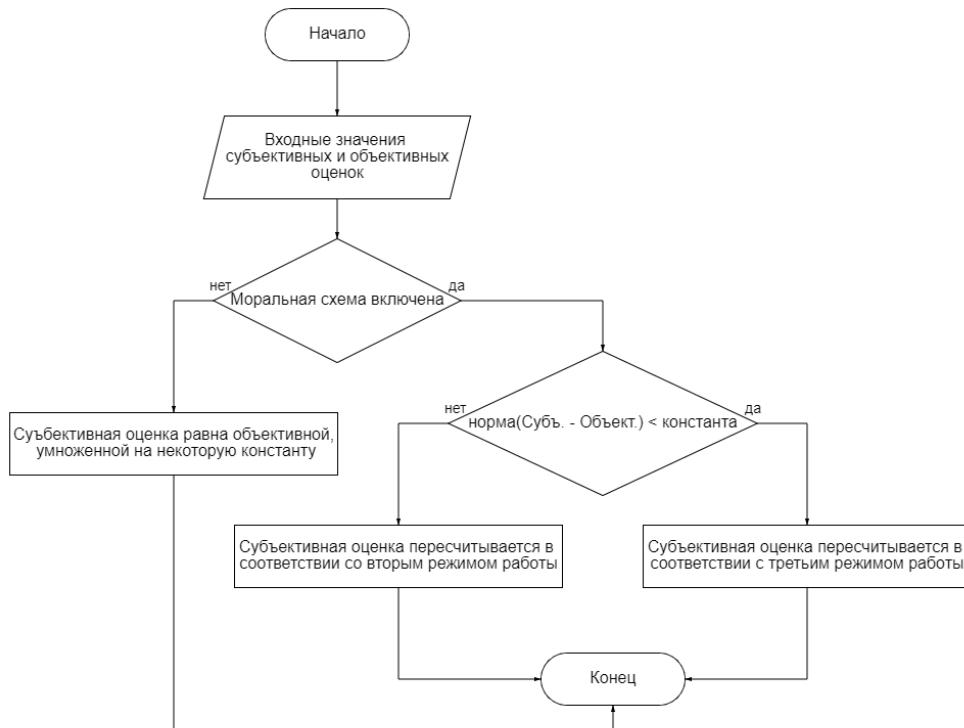


Рис. 3.4 – Блок-схема алгоритма пересчета значений Feelings акторов

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 пингвиным. 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 севе-

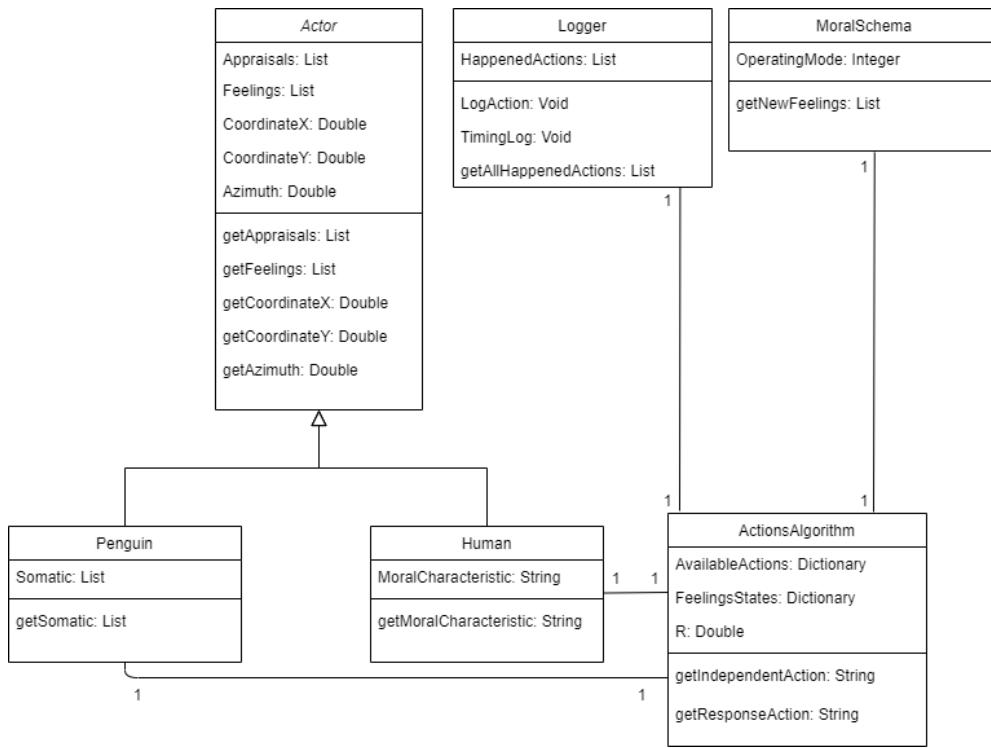


Рис. 3.5 – Диаграмма классов, описывающая взаимодействие Виртуального агента и человека

ра (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, представленного 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 фильтрацию 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 со-

ставляющую, но уже из текста. CommandFromText итерируется по полученному тексту и сопоставляет ключевые слова с действиями пингвина по алгоритму из рисунка 3.2 . Полученные команды передаются в ActionAlgorithm для того, чтобы увеличить вероятность выполнения распознанных действий. Все эмоциональные оценки передаются в Penguin и меняют его состояние соответствующе.

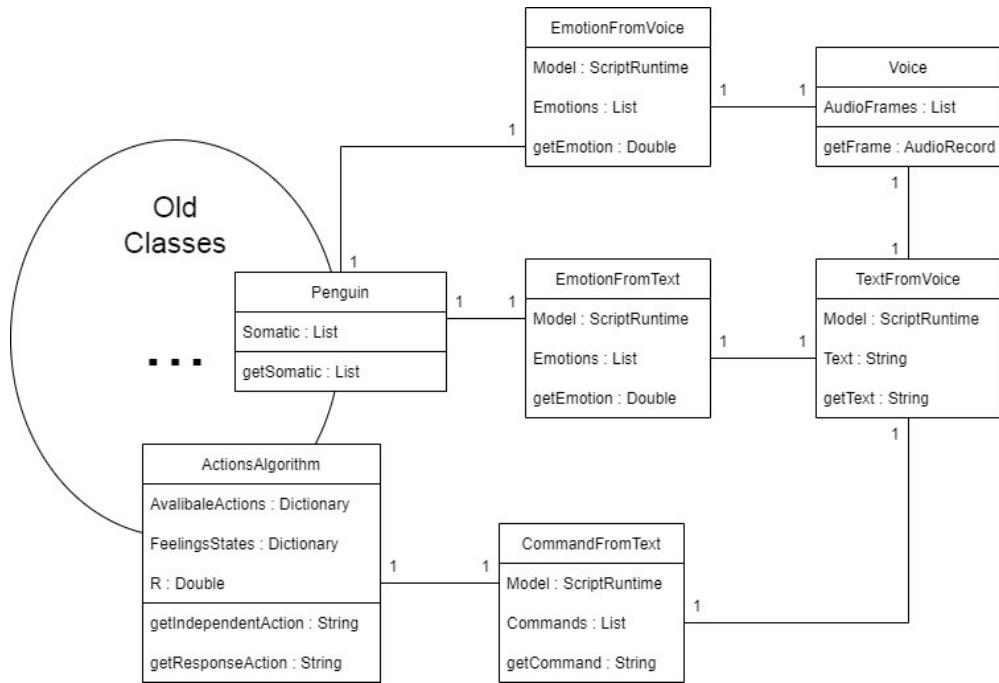


Рис. 3.6 – Расширенная диаграмма классов с участием голосового распознавания

### 3.6 Выводы

Производилось проектирование модифицированного прототипа виртуального актора с учетом всех ключевых особенностей парадигмы объектно ориентированного программирования. Были разработаны модели семантического, эмоционального анализа речи.

## 4. Реализация программного продукта

В данном разделе приводятся измененный алгоритм поведения Виртуального Актора.

### 4.1 Текущая версия реализованной модели

Для решения задачи разработки экспериментальной платформы на базе виртуального окружения для изучения социально – эмоционального поведения акторов целесообразно использовать специальные программные окружения для разработки виртуальных окружений и сопутствующих компонент – «игровые движки». По результатам анализа стало понятно, что такая среда должна прежде всего обладать:

1. Большим сообществом разработчиков
2. Подробной документацией
3. Относительно низкой сложностью
4. Простотой установки
5. Модульной системой

Всеми данными свойствами обладает лишь Unity3D, как самый распространённый, на данный момент инструмент построения виртуальных окружений. На базе Unity3D сделано больше игр, чем на любой другой технологии, поэтому он и обладает наиболее развитым сообществом разработчиков на данный момент. В сети имеется большое множество документации и курсов по данной технологии. Благодаря модульной системе, можно найти специальные программные модули, которые легко встраиваются в разработанный продукт, расширяя возможности всей системы. Преимуществом данной среды также следует считать относительную простоту переноса разработанного виртуального окружения на другие платформы (например, смартфоны, планшеты или же любую из существующих операционных систем). Также стоит отметить, что программная среда имеет бесплатную лицензионную версию для небольших команд разработчиков.

Unity предлагает разработчику возможность использовать три различных сценарных языка: C#, JavaScript (его модификация) и Boo (собственный диалект Python). Для разработки данной экспериментальной платформы был выбран C#, как де facto, стандарт при разработке на Unity3D.

В качестве среды разработки используется Visual Studio 2019. Для контроля версий использовать облачное хранилище Dropbox и GitLab. В качестве средства проектирования использовать

вался стандартный модуль Visual Studio для построения диаграмм классов.



Рис. 4.1 – Скриншот взаимодействия человека с Виртуальным Актором

В работе было реализовано программное приложение с помощью движка юнити результаты работы которого вы можете видеть на (рис. 4.1)

## 4.2 Интеграция моделей из python в C#

В расширенной диаграмме классов, было выделено какие стадии обработки проходит аудиозапись (Рис. 3.6).

Получение Audio из UNITY3D осуществлялось при помощи класса Microphone и далее помещалось в контейнер для аудио данных AudioClip, который хранит аудиофайл либо сжатым в ogg vorbis, либо без сжатия.

Далее, полученная аудио проходит обработку при помощи моделей реализованных на языке программирования python, при непосредственном использовании IronPython, который представляет из себя реализацию языка программирования Python с открытым исходным кодом, которая тесно интегрирована с .NET Framework. IronPython может использовать библиотеки .NET Framework и Python, а другие языки .NET могут также легко использовать код Python».

Существует несколько методов для работы со скриптами в Ironpython (Рис. 4.2):

Мы использовали метод Executefile(), так как в нашем случае он самый подходящий.

В указанном выше методе происходит следующее:

- В коде C# в метод Executefile(@"/home/...") помещен путь к файлу .py
- Функция из Python определяется в C#,
- Возвращается результат по завершению исполнения кода .py,



Рис. 4.2 – Методы для работы со скриптами в Ironpython

Таким образом проходит интеграция реализации динамического языка программирования IronPython в проект.

### 4.3 Реализация и дообучение модели

Как говорилось ранее для создания возможности эмоционального взаимодействия пользователя с виртуальным актором - берется предобученная модель "jonatasgrosman/wav2vec2-large-xlsr-53-russian". Данная модель работает в составе библиотеки huggingsound. На выходе модели для отдельного аудиофайла получаются 512-мерные вектора. Далее высчитывается средней вектор, который передается в модели для распознавания эмоций. Используется python3.8 и библиотека машинного обучения pytorch и sklearn.

Для SVC с ядром RBF менялся параметр регуляризации, а именно он принимал значения 1, 10 и 100.

Для k-NN менялось количество соседей от 10 до 40 с шагом в 10.

Для MLP выходной слой не менялся и состоял из 8 нейронов, тогда как количество внутренних слоев менялось от 1 до 3 так, что количество нейронов в них оставалось неизменным и равным 1024. В качестве функции активации была выбрана функция гиперболического тангенса.

В результате было получено (рис. ??):

При обучении датасет разбился на парти в 100 образцов в каждой так, чтобы была возможность обучать модели на GPU. Обучение проводилось до 10 эпох так, что результирующей мо-

деюсь становилась та, что показывала максимальный результат в какой-либо эпохе. Так как ставится задача классификации, то в качестве функции потерь используется функция потерь перекрестной энтропии. В качестве оптимизатора использовался оптимизатор Адам.

#### 4.4 Выводы

Было реализовано приложение позволяющее взаимодействовать с виртуальным окружением, в частности с виртуальным актором. Были дообучены модели машинного обучения с целью построения эмоционального взаимодействия. Для чего в приложение были интегрированы различные технические средства.

## **Заключение**

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 реализована система 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, основывающиеся на семантической 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.

1. Были проанализированы различные когнитивные архитектуры.
2. Проанализированы методы распознавания речи.
3. Доработан алгоритм поведения Виртуального Актора.
4. Унифицирован алгоритм поведения Виртуального Актора.
5. Был разработан, алгоритм выявления эмоций из человеческой речи.
6. Добавлено возможность эмоционального взаимодействия с виртуальным актором.
7. 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. V. S. A. Goal reasoning as a general form of metacognition in BICA //Biologically Inspired Cognitive Architectures. – 2014.
12. MyStem. Я. технология. Распознавание речи. – 2013. – URL: <https://habr.com/ru/company/yandex/bl>
13. В.И. А. Ж. В. Р. Алгоритм и методы распознавания речи. – 2016. – URL: <http://ainsnt.ru/file/out/8412>
14. Черняк Е. Глубинное обучение в обработке и анализе текстов. – 2018. – URL: <https://postnauka.ru/l>
15. С.В. Г. Аналитический обзор методов распознавания речи в системах голосового управления. – 2009. – URL: [http://vestnik.ispu.ru/sites/vestnik.ispu.ru/files/publications/83-85\\_0.pdf](http://vestnik.ispu.ru/sites/vestnik.ispu.ru/files/publications/83-85_0.pdf).

16. *recурс] W.* [ N-грамма. —. — URL: <https://ru.wikipedia.org/wiki/N-грамма> (дата обращения 20.10.2020).
17. *Донгес Н.* Периодические нейронные сети и LSTM. —. — TowardsDataScience.com, URL: [https://towardsdatascience.com/perIODIC-neural-networks-and-lSTM-4b601dd822a5](https://towardsdatascience.com/periodic-neural-networks-and-lstm-4b601dd822a5) (дата обращения 30.09.2020).
18. *Sovse A.* Rus speech recognition. —. — URL: <https://github.com/sovse/Rus-SpeechRecognition-LSTM-CTC-VoxForge>.
19. *Mike Schuster K. K. P.* Bidirectional recurrent neural networks. — 1997. — URL: [https://www.researchgate.net/publication/2204577/Bidirectional\\_recurrent\\_neural\\_networks](https://www.researchgate.net/publication/2204577/Bidirectional_recurrent_neural_networks)
20. *I. O.* How Neural Networks Recognize Speech-to-Text. — 2019. — URL: <https://dzone.com/articles/how-to-train-a-neural-network-to-recognize-speech>.
21. *Xuejiao Li Z. Z.* Speech Command Recognition with Convolutional Neural Network. —. — URL: <http://cs229.stanford.edu/proj2017/final-reports/5244201.pdf>.
22. *Dimitri Palaz Mathew Magimai-Doss R. C.* Convolutional neural networks-based continuous speech recognition using raw speech signal. —. — URL: [https://ronan.collobert.com/pub/matos/2015\\_rawspeech.pdf](https://ronan.collobert.com/pub/matos/2015_rawspeech.pdf)
23. Глубокое обучение для новичков: распознаем изображения с помощью сверточных сетей. — 2016. — URL: <https://www.pvsm.ru/algoritmy/210060>.
24. *Stanford.* Convolutional Neural Networks for Visual Recognition. — 2021. — URL: <https://cs231n.github.io/convolutional-neural-networks/#pool>.
25. *Christensen H. I.* Pattern Recognition – Introduction. —. — URL: [http://www.hichristensen.net/CSE291/pattern\\_recognition/introduction.html](http://www.hichristensen.net/CSE291/pattern_recognition/introduction.html)
26. *Tharwat A Gabel T H. A.* Parameter optimization of support vector machine using dragon fly algorithm. —. — International conference on advanced intelligent systems and informatics.
27. Multilayer perceptron. —. — URL: [https://en.wikipedia.org/wiki/Multilayer\\_perceptron](https://en.wikipedia.org/wiki/Multilayer_perceptron).
28. *Y. K.* Convolutional neural networks for sentence classification // Proc. of the 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP 2014). — 2014.
29. Селезнев К., Владимиров А. Лингвистика и обработка текстов // Открытые системы. — М., 2013.
30. *Mirkin B.* Core Concepts in Data Analysis: Summarization, Correlation and Visualisation, DOI. — 2011.
31. *Manning C., Schuetze H.* Foundations of Statistical Natural Processing. MIT. — М., 1999.
32. *М. Л. Ю.* Люди и знаки. — 2010.
33. *MyStem. Я. технология.* MyStem. —. — URL: <https://tech.yandex.ru/mystem>.

34. *Xin. R.* Word2vec parameter learning explained. 2014. arXiv preprint arXiv: 1411.2738. — 2010.
35. *Weisfeld. M.* The Object-Oriented Thought Process. — Fourth Edition. — Addison-Wesley Professional. — 2013.