# Использование агентного подхода с моделями глубокого обучения для обработки текстовых и табличных данных при диагностике заболеваний щитовидной железы

Е.В. Дюльдин, А.Ж. Маканов, Е.В. Боброва, К.С. Зайцев, А.А. Гармаш,
Д.Д. Шарипов, И.А. Кузнецов, С.С. Основин

***Аннотация.*** **Целью настоящей работы является исследование систем преобразования табличных данных и алгоритмов генерации меток заключений врача на основе вложенных форматов данных. В результате исследования данных биомедицинского домена получена система-конвейер для поэтапной конвертации табличных данных в скрытые вложения и генерации классификационных меток по системе Bethesda. Для модельного проектирования использован агентный подход и трансформенные методы на основе бустинга выходных ответов решателей для формирования результирующего ансамбля моделей машинного обучения. В работе предложены методы для формирования и сэмплирования итоговых наборов данных на основе алгоритмов генерации врачебных данных и заключений по классификации Bethesda Thyroid. Основным результатом является конвейер для генерации классов меток по системе Bethesda. При решении задач были выбраны подходы на основе идей автоэнкодеров и их модификаций для дистилляции знаний на основе подхода учитель-ученик, вспомогательная архитектура основана на бустинге и выделении наиболее важных признаков для построения решающих деревьев. Предложенное решение используется в рамках системы умного помощника врача для минимизации времени принятия решений для высокоуровневых специалистов и являться помощником для начинающих карьеру врачей. Система автоматизирует рутинные задачи и улучшает качество диагностики в цитологическом домене.**

***Ключевые слова*** **– агентная модель, глубокое обучение, языковая модель, трансформер, Bethesda, генерация, классификация.**

## I. ВВЕДЕНИЕ

Для назначения эффективного оперативного лечения медицинский цифровой домен в любой нозологии включает множественные исследования, необходимые для точного определения первичных проблем пациента, дальнейшего диагностирования и структурирования информации о заболевании.

В итоге лечащие врачи получают разные исследовательские (тестовые) данные и, используя свой экспертный опыт, приводят эти данные к единому согласованному формату, который может применяться в дальнейшем специалистами других областей.

Одной из проблем предобработки данных является избыточность текстовых и численных показаний, т.е. одно итоговое врачебное заключение может включать описания нескольких текстовых заключений заболевания, сделанных отдельными специалистами. Часто, чтобы их согласовать, приходится назначать пациенту дополнительные проверки (тесты) для поиска решения конкретной медицинской проблемы.

Поэтому возникает задача упрощения сложных наборов текстовых данных и приведения их к векторному виду или виду скрытых вложений, что является простым формированием слоя эмбеддингов, как проекции вектора одной размерности в другую при условии сохранения логической связи внутренних состояний. Идея такого формирования скрытых вложений описана в статье десятилетней давности [1].

В настоящей работе исследуются не только текстовые вложения, но и числовые табличные форматы данных, что накладывает ряд ограничений на используемые методы создания эмбеддингов. Центральной идеей при формировании результирующего скрытого вложения в этом случае становится использование конкатенации выходного слоя эмбеддингов и дополнительного вектора, получаемого в результате группировки и нормализации числовых данных. Итоговый размер результирующего вектора подбирается на валидационной выборке.

После получения выходного результирующего вектора, продолжением этой работы является построение классификатора для системы оценивания. Для проверки предлагаемого подхода используются анонимированные данные цитологических исследований ЭНЦ Минздрава с классификатором (категоризатором) Bethesda (The Bethesda System for Reporting Thyroid Cytopathology, сокр. TBSRTC) [2], которая помогает стандартизировать интерпретацию данных в разных странах и определить дальнейшую траекторию ведения пациента.

Для классификации полученных ранее наборов данных будут применяться такие методы классического машинного обучения, как бустинг -

для формирования решающих пней небольшой размерности и минимизация ошибки, получаемой после каждого шага бустинга. Итоговой моделью будет последовательная композиция большого набора решающих деревьев [3] и стекинг, что по сути является ансамблем решателей любого вида, В настоящей работе будет применен ансамбль из нескольких моделей бустинга.

Для улучшения результирующей целевой метрики на каждом шаге генерации применяются нейросетевые подходы на основе автоэнкодеров и слоёв внимания [4].

В ходе исследования рассмотрены различные решения, базирующиеся на современных архитектурах нейронных сетей и на классических методах машинного обучения [5].

## II. ИСХОДНЫЕ ДАННЫЕ

Исходные данные представлены в виде первичного набора простых json файлов с разным уровнем вложенности.

Табличные данные частично сгенерированы из эмпирических распределений и экспертного клинического опыта, а также аргументированы, опираясь на систему Bethesda. Начальный объем данных соответствовал 4000 примеров из всех групп системы Bethesda, которая накладывает ряд ограничений на возможные выходные данные для каждого случая заболевания. Это позволяет корректно обрабатывать граничные случаи при обогащении данных генеративными вложениями.

Структура json файла представлена в виде набора необходимых колонок, которые соответствуют следующим полям:

wsi_class - класс Bethesda (выделяются шесть основных классов по степени серьёзности заболевания от меньшего к большему);

probs - вероятность принадлежности меток каждого класса к определённой группе, где наибольшее значение принимает целевая метка, остальные значения подчиняются нормальному распределению;

cluster_characteristics - набор вложенных значений, описывающий общие характеристики наблюдаемого кластера клеток данных;

cell_characteristics - набор значений, описывающий общие характеристики для всего набора клеток;

cell_clusters - набор большой вложенности, описывающий каждый кластер отдельно с частотой появления клеток кластера и общего врачебного заключения для каждой позиции в рассматриваемом наборе значений;

cells - набор описаний каждой клетки, включающей детальное описание каждого ядра и вложений рассматриваемой клетки.

Для решения задачи можно использовать рассмотренные выше поля, используя подходы по агрегации значений и выделения ключевых позиций, можно получить вектор чисел, характеризующий каждый кластер и наборы клеток внутри него.

Числовые данные являются частью информации, которую мы можем получить, преобразуя json к формату векторов. Цикл преобразования данных в реальном мире можно описать, как конвейер, где шаг за шагом мы получаем всё более общее представление, что логично ведь врач опирается на базовые тесты и свой опыт формирует финальное заключение с меткой Bethesda заболевания пациента.

Формализуя цикл преобразования данных, выделим следующие этапы.

1. Описание проблемы пациента - первичный анализ.
2. Описание тестов, пройденных пациентом - визуальная составляющая цитологических снимков.
3. Описание видимого врачом на снимке после прохождения пациентом всех необходимых процедур.
4. Заключение врача - результат анализа данных на основе врачебного опыта и критериев Bethesda.
5. Генеративная метка Bethesda - выделенная метка заболевания.

В этой работе мы предсказываем метку Bethesda и выделяем те наборы данных для первой итерации, которые помогут нам увеличить итоговую вероятность получения целевой метки.

Итоговое распределение по каждому классу принимает вид (таблица 1).

Таблица 1. Распределение меток в наборе данных.

| Класс | Число примеров | Процент вхождения |
|---|---|---|
| 1 | 15260 | 0.15260 |
| 2 | 13098 | 0.13098 |
| 3 | 17921 | 0.17921 |
| 4 | 11008 | 0.11008 |
| 5 | 21000 | 0.21000 |
| 6 | 21713 | 0.21713 |

Число примеров будет оставаться неизменным, но будут использоваться различные подходы по агрегации и нормировке данных для получения более качественных представлений результирующих векторов.

## III. ПРЕДОБРАБОТКА НАБОРА ДАННЫХ

Чтобы задачи генерации заключений и классификации заболевания были успешными, необходимо провести предварительную обработку данных. Она включает в себя 3 этапа.

*Очистка данных*. Удаляются записи без меток и неинформативные заключения, что обеспечивает единообразие и полноту данных. Также здесь исправляются или удаляются записи с отсутствующими или противоречивыми данными, что поможет избежать ошибок и искажений при дальнейшем анализе.

*Нормализация данных*. Все метки приводятся к единому виду. Это предотвращает ошибки при интерпретации и анализе данных. Например, если

метки Bethesda представлены в разных форматах, их нормализация поможет унифицировать данные.

*Разделение данных*. Данные делятся на обучающую, валидационную и тестовую выборки. Это позволяет эффективно обучить модель и проверить её на независимых данных. Обычно используется соотношение 70% для обучения, 15% для валидации и 15% для тестирования, что обеспечивает сбалансированное распределение данных для каждой из задач.

Для того чтобы алгоритмы машинного обучения могли работать с категориальными данными, необходимо преобразовать их в числовые значения. Это называется кодированием категориальных переменных.

Также важно обработать пропущенные значения: заменить их или удалить. Замена пропущенных значений может осуществляться с помощью средних, медианных или предсказанных значений на основе других данных.

Удаление записей с пропущенными значениями возможно, если таких записей немного.

Медицинские тексты обычно слабо структурированы, что усложняет задачу выделения ключевой информации и удаления шума для дальнейшего анализа и создания признаков.

Также структура медицинских текстов плохо выражена, из-за чего сложно убрать лишние данные и выделить важную информацию для анализа и создания признаков.

Чтобы выделить целевые метки и ключевые слова в неразмеченных данных, лучше использовать регулярные выражения и вероятностный поиск наиболее часто встречающихся слов в предложениях. Регулярные выражения помогают найти разные комбинации меток Bethesda в текстах.

После очистки данных и создания меток предложения можно разделить на группы по классам. Также этот процесс включает определение количества предложений в тексте и вероятности появления слов в каждом предложении. Это важный параметр для методов увеличения объёма данных и проверки результатов. Разделение длинных заключений на отдельные предложения с метками Bethesda позволяет расширить набор признаков на 8,4%.

Таким образом, тщательная предобработка данных является фундаментом для успешного применения методов глубокого обучения в задачах многоклассовой классификации и генерации медицинских данных по системе Bethesda.

Правильная очистка, нормализация и кодирование данных, а также обработка пропущенных значений и структурирование табличных данных обеспечивают основу для построения эффективной модели, способной решать задачи медицинской диагностики.

## IV. ПРИМЕНЕНИЕ МЕТОДОВ ОПТИМИЗАЦИИ ВТОРОГО ПОРЯДКА

Первый этап моделирования меток класса по методологии Bethesda включает получение внутренних вложений векторов для дальнейшего обучения. На этапе формирования примеров и очистки данных было предложено два основных способа по генерации json формата файлов и дальнейшего снижения размерности для векторных вложений.

В этом разделе рассмотрим методы, использующие классические подходы машинного обучения, главным из которых будет семейство бустингов, опирающиеся на минимизацию ошибки предсказания на каждом шаге, а не функции потерь, как принято в подходах классической реализации градиентного бустинга [6].

Предлагаемые подходы основаны на трёх основных алгоритмах автоматического построения решающих деревьев малой размерности [7], что позволит на этапе валидации выбрать не только лучшую модель, но и принять решение о стекинге нескольких подходов, что предполагает на выходе агентную систему с голосованием каждого кандидата по принадлежности результирующей метки верному классу Bethesda.

Построим решение на основе одного из классических подходов [7]. Набор множества решающих деревьев на основе эмпирического опыта решения классификационных задач будет являться отправной точкой для дальнейшего исследования, так как изначально использование других техник машинного обучения будет накладывать значительные ограничения и не позволит провести эксперименты без значительных временных сложностей, что противоречит основной идее выбранного подхода построения базового решения решаемой задачи.

Первый рассматриваемый алгоритм относится к решающим деревьям. Для обучения дерева была выбрана механика поиска по сетке, где настраиваемые оптимизационные параметры представлены в таблице 2.

Таблица 2. Поиск параметров по сетке.

| Тип параметра | Численное значение |
|---|---|
| n_estimators | 100, 200, 300, 400, 500 |
| criterion | gini |
| max_depth | 6, 8, 10, 12, 14, 16, 18, 22, 28, 30, 32 |
| min_samples_split | 1, 2, 3, 4, 5 |
| max_features | sqrt, log2 |

Наиболее важными параметрами является число деревьев, которое показывают общее количество обучаемых отдельных деревьев в ансамбле. В настоящей работе предполагается наибольшее число решающих деревьев до достижения порога переобучения модели - процесса, когда модель умеет приближать тренировочные данные, но не

понимает закономерностей на валидационной выборке.

Остальные критерии подбираются по сетке, где используемый алгоритм перебора значений является случайным сэмплированием, что позволяет выбирать не только граничные точки, но и их интервальные представления. Результаты обучения представлены в таблице 3.

В приведённой таблице показаны три лучших результата поиска по сетке. Наблюдаемые параметры являются выходами лучших моделей, а результирующая метрика это f1-score (формула 1).

Таблица 3. Обучение Random Forest.

| | n_estimators | criterion | max_depth | min_samples_split | max_features | f1-score |
|---|---|---|---|---|---|---|
| 1 | 270 | gini | 8 | 2 | sqrt | 0.41 |
| 2 | 350 | gini | 10 | 1 | log2 | 0.57 |
| 3 | 400 | gini | 6 | 1 | log2 | 0.39 |

$$\frac{(1 + B^2) * TP}{(1 + B^2) * TP + FP + FN} \quad (1)$$

В этой формуле $B$ - является настраиваемым параметром, показывающим смещённость выходных предсказаний в сторону полноты или точности модели. $TP$ - правильно предсказанные метки классов, $FP$ - ложноположительные предсказания, $FN$ - ложноотрицательные предсказанные метки классов.

При использовании случайного леса большой глубины происходит деградация целевого качества на валидационной выборке. В этой задаче ограничено число признаков в каждом примере, что и приводит к переобучению глубоких деревьев. Изменение целевой метрики при константных параметрах, но различной глубине случайного леса, представлена в таблице 4.

Таблица 4. Зависимость метрики от глубины леса.

| max_depth | f1-score |
|---|---|
| 12 | 0.375 |
| 14 | 0.372 |
| 16 | 0.358 |
| 18 | 0.12 |
| 20 | 0.17 |

В результате при увеличении общего числа деревьев и глубины каждого из них на валидационной выборке получаем функцию монотонно убывающей величины f1-score.

Следующим алгоритмом является набор градиентных бустингов над решающими пнями. Настраиваемые параметры обучения представлены в таблице 5.

Таблица 5 - параметры обучения моделей бустинга

| Тип параметра | Численное значение |
|---|---|
| iterations | 10, 20, 30, 40, 45, 55, 60 |
| learning rate | 1e-1, 1e-2, 1e-3, 1e-4, 1e-5 |
| depth | 1, 2, 3, 4, 5, 6, 7, 8 |
| random_seed | 42 |

Обучим модели на основе трёх видов бустинга со сбалансированными деревьями CatBoost [3], с несбалансированными деревьями LGB [8] и с использованием масштабирования Xgboost [9]. Для каждого из выбранных алгоритмов проведём партии независимых экспериментов и получим лучшую целевую метрику f1-score на валидационной выборке.

Выходом обучения по случайной сетке на заданных параметрах при выбранных моделях будут результаты таблицы 6.

Таблица 6 - оценки моделей бустинга.

| Тип бустинга | F1-score |
|---|---|
| LGB | 0.592 |
| Xgboost | 0.631 |
| CatBoost | 0.679 |

Как видно из таблицы, наилучшим образом показывает себя CatBoost, который на валидационной выборке выдаёт наилучшую целевую метрику.

В процессе обучения были применены два типа отбора признаков. Первый - основан на выходной важности каждого веса на каждом слое, для бустинга в наборе решающих деревьев. Второй - подразумевает методологию SHAPE [10], которая основана на математическом ожидании прогнозов при выборе каждого отдельного признака во входной последовательности. Визуальное отображение этого метода на множество данных представлено на рисунке 1.

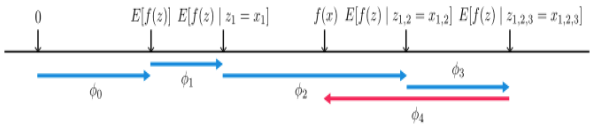


Рисунок 1 - математическое ожидание SHAPE

В моделях использовались, только те признаки, для которых значение SHAPE было не нулевым, что означало их результирующий вклад в решение задачи.

Последним этапом создания конвейера классического машинного обучения является калибровка моделей [11], что позволяет преобразовать вероятностное пространство методами замены функций в область, где каждому элементу случайной величины будет соответствовать значения наибольшей вероятности по выборке из генеральной совокупности [12].

В этой работе не рассматриваются сильные и слабые подходы для калибровки моделей, но для получения более точных значимых результатов используется подход на основе оценки выходных весов функций из решающих деревьев [13]. Этот метод выравнивает результирующую функцию

вероятностей полученной случайной величины выходов градиентного бустинга.

## V. ВЕРОЯТНОСТНАЯ ИНТЕРПРЕТАЦИЯ ДАННЫХ

Интересен анализ модели с вероятностной точки зрения, поскольку наши данные были сгенерированы, т.е. создавались из заранее заданных эмпирических распределений. Наилучшая интерпретация работы модели может быть достигнута путем предположения того, что целевая переменная $y$ (класс Bethesda) зависит от исходных числовых признаков $x$ линейно (за исключением softmax в самом конце)

$$y = \theta^T x;\ x, \theta \in R^d$$
$$E[y] = \theta^T E[x],$$

где $\theta$ – $d$-мерный вектор в вещественном пространстве.

Если удастся найти математическое ожидание [14] каждого из признаков, то из этого можно будет вывести математическое ожидание целевой переменной для различных меток Bethesda. Если $E[y]$ будет различаться, то можно будет сделать вывод, модель обладает дискриминативной силой, и процесс генерации данных из эмпирических распределений не содержит ошибок.

Самый первый признак – $x_1$ распределен равномерно:

$$x_1 \sim Uniform[a, b], x \in R$$
$$E[x_1] = \frac{a+b}{2}, \quad (2)$$

где $a$ и $b$ – параметры равномерного распределения, нижняя и верхняя границы отрезка.

Второй признак $x_2$ выражается следующим образом:

$$x_2 = C_1 x_1,$$

Где $C_1$ – линейный коэффициент, множитель, являющийся отношением второго признака к первому.

Таким образом, математическое ожидание второго признака выражается как:

$$E[x_2] = C_1\, E[x_1] = C_1 \frac{a+b}{2} \quad (3)$$

Третий признак $x_3$ равняется:

$$x_3 = \sqrt{x_2} + x';\ x' \sim Uniform[c, d] \quad (4)$$

Где $x'$ - некое случайное равномерное смещение относительно $\sqrt{x_2}$. Найдем математическое ожидание корня случайной переменной, распределенной изначально равномерно:

$$\int_a^b \frac{1}{b-a}\sqrt{C_1 x}dx = \frac{2}{3}\sqrt{C_1(b-a)} \quad (5))$$

Имея это, можем вычислить математическое ожидание третьего признака:

$$E[x_3] = \frac{2}{3}\sqrt{C_1(b-a)} + \frac{c+d}{2} \quad (6)$$

Таблица 7. Упомянутые ранее константы в распределениях.

| a | b | $C_1$ | c | d |
|---|---|---|---|---|
| 10 | 50 | 500 | 0 | 25000 |

Некоторые из признаков генерируются из одних и тех же распределений (равномерных), однако эти параметризуются в зависимости от других случайных переменных. Во всех случаях эти переменные исходят из категориальных распределений, или из распределений Бернулли.

В этом случае математическое ожидание конечного признака можно принять как:

$$\mathrm{E}[x] = \sum_n p_n * E_{z \sim Uniform(n)}[z]), \quad \text{где} \quad (7)$$

$n$ – один конкретный исход из категориального распределения/распределения Бернулли,
z – значение признака,
*Uniform(n)* – соответствующее этому исходу равномерное распределение.

## VI. АНАЛИЗ ЗАДАЧИ КЛАССИФИКАЦИИ

Классификация данных сводится к выбору класса с наибольшей апостериорной вероятностью. Рассмотрим простейший пример, когда логиты $c$, подаваемые на softmax (функция преобразования в вероятности), представляют собой функцию линейную относительно вектора входных признаков $x$.

$$c = \Theta x; \quad \Theta \in R^{C \times d},$$

где $\Theta$ – матрица линейных коэффициентов модели, $C$ – количество возможных классов. Также, следует помнить, что мы заранее знаем распределения признаков относительно целевого класса, т.е. можно рассчитывать их среднее.

$$E_{x \sim p_c(x)}[x]$$

Где $p_c(x)$ – априорное распределение признака в классе $c$.

Таким образом, можно задать Loss-функцию (8):

$$L(\Theta) = -\sum_x \sum_c p_c(x) \log\big(q(x|\Theta)\big)$$
$$= -\sum_c \sum_x p_c(x) \log\big(q(x|\Theta)\big) =$$
$$= -\sum_c \sum_x p_c(x) \log(softmax(\Theta x)_c) =$$
$$= -\sum_c \sum_x p_c(x) \left(\theta_c^T \mathrm{x} - \log\left(\sum_{c'} \exp\big(\theta_{c'}^T \mathrm{x}\big)\right)\right) \quad (8)$$

Где $q(x|\Theta)$ – предсказанная вероятность класса c, обусловленная параметрами модели Θ, $\theta_c$- строчка матрицы параметров модели Θ, соответствующая конкретному классу $c, c'$ - метка класса.

Рассчитаем градиент, необходимый для обучения с использованием градиентного спуска. Для удобства можно сначала рассчитать градиент относительно $\theta_c$ (9):

$$\nabla_{\theta_c} L(\Theta) = -\nabla_{\theta_c} \sum_x p_c(x)\theta_c^T \mathrm{x} + \nabla_{\theta_c} \sum_x p_c(x) \log\left(\sum_{c'} \exp\left(\theta_{c'}^T \mathrm{x}\right)\right) = -\sum_x p_c(x)\mathrm{x} + \sum_x p_c(x) \frac{\nabla_{\theta_c}\left(\sum_{c' \setminus c} \exp\left(\theta_{c'}^T \mathrm{x}\right) + \exp\left(\theta_c{}^T x\right)\right)}{\sum_{c'} \exp\left(\theta_{c'}^T \mathrm{x}\right)} = -\sum_x p_c(x)\mathrm{x} + \sum_x p_c(x) \frac{\exp\left(\theta_c{}^T x\right) x}{\sum_{c'} \exp\left(\theta_{c'}^T \mathrm{x}\right)} = -\sum_x p_c(x)\mathrm{x} + \sum_x p_c(x)\, q(x|\Theta) x = \mathbb{E}_{x\sim p_c(x)}[x * q(x|\Theta)] - \mathbb{E}_{x\sim p_c(x)}[x] \quad (9)$$

Лосс функция $L(\Theta)$ представляет собой 2 суммы, взвешенные на априорные вероятности x для класса $c$: сумма значения логитов $\theta_c^T \mathrm{x}$, соответствующих классу $c$ и логарифма суммы экспонент всех логитов, относящихся ко всем остальным классам $c$.

Видно, что в градиенте фигурирует константный член – математическое ожидание вектора признаков $x$, взятого из эмпирического распределения класса $c$.

Обучающее правило градиентного спуска будет выглядеть следующим образом (удобно его расписывать относительно каждого $\theta_c$, а не целой $\Theta$):

$$\theta_c \coloneqq \theta_c - \eta\left(E_{x\sim p_c(x)}[x * q(\Theta)] - E_{x\sim p_c(x)}[x]\right) = -\eta_1 E_{x\sim p_c(x)}[x * q(\Theta)] + \eta_2 E_{x\sim p_c(x)}[x] \ , \quad (10)$$

где $\eta$ – гиперпараметр, отвечающий за скорость обучения. Представляется интересным изменять параметры $\eta_1$ и $\eta_2$, и определить каким образом они влияют на процесс обучения простой линейной классификационной модели.

Применим оптимизированный градиентный спуск [17]:

$$L(\Theta) = -\sum_c p_c(x)\left(\theta_c^T \mathrm{x} - \log\left(\sum_{c'} \exp\left(\theta_{c'}^T \mathrm{x}\right)\right)\right)$$

$$\nabla_{\theta_c} L(\Theta) = -p_c(x)\mathrm{x} + p_c(x) q(x|\Theta) x \quad (11)$$

Таким образом, в случае оптимизированного градиентного спуска из формулы пропадает математическое ожидание, т.к. оценка точечная.

Рассмотрим правило обновления весов оптимизированного градиентного спуска:

$$\theta_c \coloneqq \theta_c - \eta(-p_c(x)x + p_c(x)q(\Theta)x) = \eta p_c(x)x - \eta p_c(x)q(\Theta)x \quad (12)$$

Видно, что первый член никак не зависит от $\Theta$, поэтому на каждой итерации он постоянно прибавляется к весам, в то время как второй член непостоянен в зависимости от итерации, и его кумулятивное влияние на итоговые веса можно вынести в отдельный, динамический член $A$, аналитический расчет которого в общем случае представляется невозможным. Рассчитаем формулу итоговых весов $\theta_c$ после прохождения обучения на всем датасете:

$$\theta_c^N = \theta_c^0 + \eta \sum_x p_c(x)\mathrm{x} + \eta A = \theta_c^0 + \eta \mathbb{E}_{x\sim p_c(x)}[x] + \eta A \quad (13)$$

Также интересно оценить долю влияния члена $\eta E_{x\sim p_c(x)}[x]$ на итоговые параметры модели.

Для этого можно рассчитать:

$$\alpha = \frac{\left|E_{x\sim p_c(x)}[x]\right|_2^2}{\left|A + E_{x\sim p_c(x)}[x]\right|_2^2} \ , \quad (14)$$

где $A$ – динамический член. Исследуем зависимость отношения вклада члена математического ожидания и вклада динамического члена от скорости обучения $(\eta)$. График на рис.2 демонстрирует эту зависимость.

Видно, константный член постоянно дает больший вклад, при этом, чем более агрессивно мы обновляем веса $\theta_c$, тем более влияние динамического члена приравнивается к константному.

График на рис.3 показывает различные сценарии изменения доли динамического члена в процесса обучения при различных $\eta$.

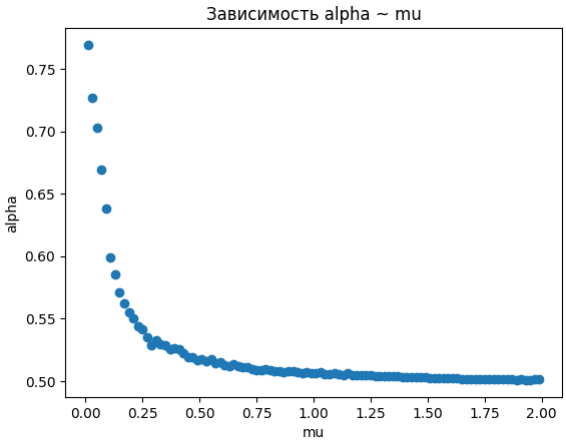


Рисунок 2 – Зависимость $\alpha$ от $\eta$

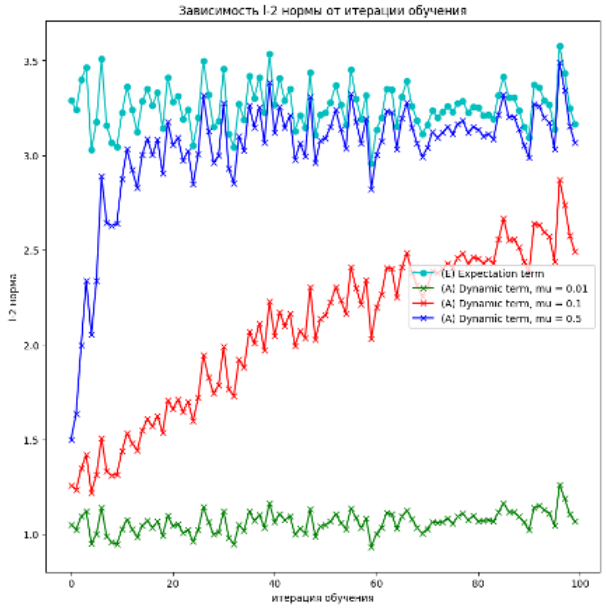


Рисунок 3 – Графики зависимости $l2$-нормы динамического члена от итерации обучения.

## VII. ЗАКЛЮЧЕНИЕ

В результате работы был построен конвейер по обработке табличных данных в формате для решения классификационной задачи по методологии Bethesda.

Предложен подход обработки данных, включающий в себя шаги по очистке, нормализации и аугментации новых примеров для обучения моделей.

В цикле последовательного обучения и подбора гиперпараметров получены необходимые веса и решающие условия для дальнейшего использования конвейера машинного обучения, как средства для решения классификационной задачи по методологии Bethesda.

Полученная модель опирается на стохастические подходы и основана на аппарате оптимизированного стохастического градиентного спуска для моделей классификации табличных данных.

Рассмотрена линейная модель, без функций активаций, но с softmax для преобразования логитов в вероятности.

В результате анализа табличных данных было найдено, что в случае, когда заранее известны априорные распределения, порождающие данные, правило градиентного спуска можно декомпозировать на константный член – легко аналитически выводимый и динамический член – значение которого меняется от итерации к итерации.

Также было показано, что с уменьшением скорости обучения модели, роль предрасчитанного математического ожидания становится заметно больше динамического члена $A$ в правиле обновления весов. В то время как при увеличении скорости можно видеть процесс обучения обуславливается все больше $A$, причем эта доля с ростом $\eta$ выходит на плато.

## БЛАГОДАРНОСТИ

Авторы выражают благодарность Высшей инжиниринговой школе НИЯУ МИФИ за помощь в возможности опубликовать результаты выполненной работы и руководству ФГБУ «НМИЦ эндокринологии» Минздрава России за предоставленные табличные и текстовые данные.

## БИБЛИОГРАФИЯ

[1] Mikolov T. Efficient estimation of word representations in vector space //arXiv preprint arXiv:1301.3781. – 2013. – Т. 3781.

[2] Juhlin C. C., Baloch Z. W. The 3rd edition of Bethesda system for reporting thyroid cytopathology: Highlights and comments // Endocrine Pathology. – 2024. – Т. 35. – №. 1. – С. 77-79.

[3] Prokhorenkova L. et al. CatBoost: unbiased boosting with categorical features //Advances in neural information processing systems. – 2018. – Т. 31.

[4] Kingma D. P. Auto-encoding variational bayes //arXiv preprint arXiv:1312.6114. – 2013.

[5] Генерация врачебных заключений и классификация по Bethesda с использованием глубокого обучения / Е. В. Боброва, А. Ж. Маканов, С. С. Основин [и др.] // International Journal of Open Information Technologies. – 2023. – Т. 11, № 10. – С. 119-129. – EDN WAVOVQ.

[6] Fuhrer B., Tessler C., Dalal G. Gradient Boosting Reinforcement Learning //arXiv preprint arXiv:2407.08250. – 2024.

[7] Louppe G. Understanding random forests: From theory to practice //arXiv preprint arXiv:1407.7502. – 2014.

[8] Sheridan R. P., Liaw A., Tudor M. Light gradient boosting machine as a regression method for quantitative structure-activity relationships //arXiv preprint arXiv:2105.08626. – 2021.

[9] Chen T., Guestrin C. Xgboost: A scalable tree boosting system //Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. – 2016. – С. 785-794.

[10] Lundberg S. A unified approach to interpreting model predictions //arXiv preprint arXiv:1705.07874. – 2017.

[11] Wang C. Calibration in deep learning: A survey of the state-of-the-art //arXiv preprint arXiv:2308.01222. – 2023.

[12] Vasilev R., D'yakonov A. Calibration of neural networks //arXiv preprint arXiv:2303.10761. – 2023.

[13] Niculescu-Mizil A., Caruana R. Obtaining Calibrated Probabilities from Boosting //UAI. – 2005. – Т. 5. – С. 413-20.

[14] Математическое ожидание https://ru.wikipedia.org/wiki/Математическое ожидание

[15] Градиентный спуск https://ru.wikipedia.org/wiki/Градиентный_спуск

Дюльдин Евгений Владимирович, Национальный Исследовательский Ядерный Университет МИФИ, аспирант, zhecos1@yandex.ru

Маканов Артём Жанович, Национальный Исследовательский Ядерный Университет МИФИ, бакалавр, artem.makanov@mail.ru

Боброва Елизавета Витальевна, Национальный Исследовательский Ядерный Университет МИФИ, аспирант, EVBobrova@mephi.ru

Зайцев Константин Сергеевич, Национальный Исследовательский Ядерный Университет МИФИ, профессор, KSZaytsev@mephi.ru

Гармаш Александр Александрович, Национальный Исследовательский Ядерный Университет МИФИ директор

инженерно-физического института биомедицины, AAGarmash@mephi.ru
Кузнецов Илья Александрович, Национальный Исследовательский Ядерный Университет МИФИ, магистрант, Il582936@mail.ru
Шарипов Данил Данисламович, Национальный Исследовательский Ядерный Университет МИФИ, магистрант, danildsharipov@yandex.ru
Основин Станислав Сергеевич, Национальный Исследовательский Ядерный Университет МИФИ, аспирант 1300stas1300@gmail.com

# Using an agent-based approach with deep learning models to process text and tabular data in diagnosing thyroid diseases

E.V. Diuldin, A.Z. Makanov, E.V. Bobrova, K.S. Zaytsev, A.A. Garmash,
D.D. Sharipov, I.A. Kuznetsov, S.S. Osnovin

**Abstract. The purpose of this work is to study systems for converting tabular data and algorithms for generating doctor's report labels based on nested data formats. As a result of studying data from the biomedical domain, a pipeline system was obtained for step-by-step conversion of tabular data into hidden attachments and generation of classification labels according to the Bethesda system. For model design, an agent-based approach and transformative methods were used based on boosting the output responses of solvers to form the resulting ensemble of machine learning models. The paper proposes methods for generating and sampling final data sets based on algorithms for generating medical data and conclusions according to the Bethesda Thyroid classification. The main result is a pipeline for generating label classes using the Bethesda system. When solving problems, approaches were chosen based on the ideas of autoencoders and their modifications for distilling knowledge based on the teacher-student approach; the auxiliary architecture is based on boosting and highlighting the most important features for constructing decision trees. The proposed solution is used within the framework of a smart medical assistant system to minimize decision-making time for high-level specialists and act as an assistant for doctors starting their careers. The system automates routine tasks and improves the quality of diagnostics in the cytological domain.**

**Keywords – agents model, deep learning, language model, transformer, Bethesda, generation, classification.**

REFERENCES

[1] Mikolov T. Efficient estimation of word representations in vector space //arXiv preprint arXiv:1301.3781. – 2013. – T. 3781.

[2] Juhlin C. C., Baloch Z. W. The 3rd edition of Bethesda system for reporting thyroid cytopathology: Highlights and comments // Endocrine Pathology. – 2024. – T. 35. – №. 1. – C. 77-79.

[3] Prokhorenkova L. et al. CatBoost: unbiased boosting with categorical features //Advances in neural information processing systems. – 2018. – T. 31.

[4] Kingma D. P. Auto-encoding variational bayes //arXiv preprint arXiv:1312.6114. – 2013.

[5] Generation of Medical Reports and Classification by Bethesda Using Deep Learning / E. V. Bobrova, A. Zh. Makanov, S. S. Osnovin [et al.] // International Journal of Open Information Technologies. - 2023. - Vol. 11, No. 10. - P. 119-129. - EDN WAVOVQ.

[6] Fuhrer B., Tessler C., Dalal G. Gradient Boosting Reinforcement Learning //arXiv preprint arXiv:2407.08250. – 2024.

[7] Louppe G. Understanding random forests: From theory to practice //arXiv preprint arXiv:1407.7502. – 2014.

[8] Sheridan R. P., Liaw A., Tudor M. Light gradient boosting machine as a regression method for quantitative structure-activity relationships //arXiv preprint arXiv:2105.08626. – 2021.

[9] Chen T., Guestrin C. Xgboost: A scalable tree boosting system //Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. – 2016. – C. 785-794.

[10] Lundberg S. A unified approach to interpreting model predictions //arXiv preprint arXiv:1705.07874. – 2017.

[11] Wang C. Calibration in deep learning: A survey of the state-of-the-art //arXiv preprint arXiv:2308.01222. – 2023.

[12] Vasilev R., D'yakonov A. Calibration of neural networks //arXiv preprint arXiv:2303.10761. – 2023.

[13] Niculescu-Mizil A., Caruana R. Obtaining Calibrated Probabilities from Boosting //UAI. – 2005. – T. 5. – C. 413-20.

[14] Mathematical expectation https://ru.wikipedia.org/wiki/Mathematical expectation

[15] Gradient descent https://ru.wikipedia.org/wiki/Gradient_descent