УДК 004.81

НЕЧЕТКО-ЛОГИЧЕСКАЯ АДАПТАЦИЯ ПАРАМЕТРОВ СКОЛЬЗЯЩЕГО ОКНА ПРИ ПОДГОТОВКЕ ДАННЫХ ДЛЯ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ

М. В. Бобырь[1] [0000-0002-5400-6817], **Н. А. Милостная**[2] [0000-0002-3779-9165], **С. Ю. Бельская**[3] [0009-0000-2013-9972]

[1–3]*Юго-Западный государственный университет, г. Курск, Россия*

[1]maxbobyr@gmail.com, [2]nat_mil@mail.ru, [3]s@belskaia.ru

Аннотация

Предложен нечеткий регулятор вычисления параметров скользящего окна для подготовки обучающих данных больших языковых моделей. Традиционный подход задает параметры «шаг окна» и «длина контекста» фиксированными константами, одинаковыми для всего текста, и не учитывает такие лингвистические характеристики отдельных фрагментов, как насыщенный научный текст и монотонный повторяющийся текст. Предлагаемый метод использует два автоматически вычисляемых признака фрагмента – лексическое разнообразие и среднюю длину BPE-токена. На основе алгоритма Мамдани с базой из 9 нечетко-логических правил и дефаззификацией методом центра тяжести нечеткий регулятор адаптивно вычисляет значения параметров «шаг окна» и «длина контекста» для каждого фрагмента. Предложенный подход имеет когнитивную интерпретацию, поскольку воспроизводит механизм адаптивного внимания человека при чтении, например, сложные фрагменты обрабатываются более внимательно при малом размере шага.

Ключевые слова: *нечеткий вывод, алгоритм Мамдани, скользящее окно, LLM, Type-Token Ratio, BPE-токенизация, когнитивное моделирование, адаптивная обработка текста.*

ВВЕДЕНИЕ

Большие языковые модели (Large Language Models, LLM), такие как GPT-2, GPT-3, LLaMA, Mistral и их аналоги, на сегодняшний день стали основой современных систем генерации, перевода и анализа текста [1, 2]. Обучение LLM требует подготовки огромных массивов обучающих пар формата «вход – цель». Стандартным способом получения таких пар является метод скользящего окна, при котором токенизированный текст разделяется на перекрывающиеся фрагменты длиной max_length токенов с шагом stride [3]. Выбор этих двух параметров существенно влияет на качество обучения, например, при малом stride одни и те же токены многократно попадают в разные обучающие примеры, обеспечивая значительное контекстное разнообразие; при большом stride число примеров сокращается, но снижается вычислительная стоимость.

На практике stride и max_length задаются константами, выбранными эмпирически для конкретной архитектуры. Для GPT-2 стандартными являются max_length = 1024, stride = 512 [4]; в учебных реализациях часто используют max_length = 256, stride = 128 [3]. Такой подход игнорирует неоднородность текста, на котором обучается LLM-модель, причем различные фрагменты текста несут принципиально разный объем информации. Фрагмент из научной статьи с терминами, которые встречаются впервые, и фрагмент с повторяющимися шаблонными конструкциями обрабатываются с одним и тем же шагом, хотя оптимальные параметры для них существенно различаются.

Нечеткая логика [5] предоставляет математический аппарат для работы с лингвистическими переменными и/или понятиями, не имеющими точных числовых границ. Такие характеристики текста, как «высокое лексическое разнообразие», «длинный контекст» или «малый шаг», естественно описываются нечеткими термами. Алгоритм Мамдани [6] позволяет построить нечеткий регулятор (fuzzy controller) на основе правил вида «ЕСЛИ ТО», формализующих экспертные знания о связи между характеристиками текста и параметрами его обработки.

Применение нечеткой логики к задачам обработки естественного языка рассматривалось в ряде научных работ. С целью эффективного поиска информации в [7] нечеткие системы оценивали семантическую близость текстов.

В [8] предложен нечеткий классификатор тональности с лингвистическими переменными для описания степени позитивности/негативности. В [9] использован нечеткий вывод для адаптивной сегментации изображения. Однако применение нечеткой логики к выбору параметров скользящего окна при подготовке данных для LLM в литературе, известной нам, не рассматривалось.

Целью настоящей статьи является разработка и исследование нечеткого регулятора, адаптивно настраивающего параметры stride и max_length в зависимости от лингвистических характеристик каждого фрагмента текста. Научная новизна работы состоит в следующем: 1) задача выбора параметров скользящего окна впервые сформулирована как задача нечеткого управления; 2) предложены два вычислимых входных признака: TTR и средняя длина BPE-токена в качестве лингвистических переменных; 3) разработана база из 9 нечетко-логических правил с когнитивной интерпретацией; 4) проведен сравнительный анализ адаптивного и фиксированного подходов на реальном текстовом фрагменте “The Verdict”.

1. МАТЕМАТИЧЕСКАЯ МОДЕЛЬ НЕЧЕТКОГО РЕГУЛЯТОРА

1.1. Входные лингвистические переменные

Для построения нечеткого регулятора необходимо выбрать признаки фрагмента текста, которые: а) поддаются автоматическому вычислению без разметки; б) коррелируют с оптимальными значениями stride и max_length; в) интерпретируемы в лингвистических терминах. Рассмотрим два таких признака.

Во-первых, это переменная TTR (Type-Token Ratio), которая является отношением числа уникальных идентификаторов токенов к общему числу токенов во фрагменте текста:

$$\mathrm{TTR} = \frac{|\{\text{unique token IDs}\}|}{|\{\text{all token IDs}\}|} \in [0, 1].$$

При TTR = 1 все токены уникальны и текст максимально разнообразен. При TTR → 0 токены многократно повторяются. TTR вычисляется за число операций порядка O(n) с использованием хэш-множества непосредственно из списка идентификаторов, возвращенных BPE-токенизатором (tiktoken для GPT-2).

Во-вторых, это переменная avg_len (средняя длина BPE-токена в символах), которая определяет среднее число непробельных символов на один токен во фрагменте текста. Область ее значений находится в диапазоне [1, 8] и обусловлена свойствами BPE-словаря GPT-2. Например, минимальная длина токена в 1 символ указывает, что это служебные символы. Максимальная величина в 7–8 символов обозначает длинные термины. При этом признак отражает морфологическую сложность фрагмента: специализированные термины дают длинные подтокены, а служебные слова – короткие.

1.2. Функции принадлежности

Для каждой из четырех лингвистических переменных (x_1 = TTR, x_2 = avg_len, y_1 = stride, y_2 = max_length) определены три терм-множества. Граничные термы (Low, High, Short, Long, Small, Large) описываются трапециевидными функциями принадлежности (trapmf). Средний терм (Medium) описывается треугольной функцией принадлежности (trimf):

$$\mu_{\mathrm{trapmf}(x;\, a,b,c,d)} = \begin{cases} 0, & \text{если } x < a \text{ или } x > d, \\ \dfrac{x-a}{b-a}, & \text{если } a \le x \le b, \\ 1, & \text{если } b \le x \le c, \\ \dfrac{d-x}{d-c}, & \text{если } c < x \le d; \end{cases}$$

$$\mu_{\mathrm{trimf}(x;\, a,b,c)} = \max\left(0, \min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right)\right).$$

Трапециевидная форма для граничных термов обеспечивает устойчивость регулятора: при значениях x в «плоской» части степень принадлежности не изменяется, что исключает нежелательные колебания выходных параметров из-за малых изменений входных признаков. Параметры функций принадлежности всех переменных представлена в табл. 1.

Табл. 1. Параметры функций принадлежности нечеткого регулятора.

Переменная	Терм	Тип ФП	Параметры [a, b, c, d] или [a, b, c]	Диапазон значений
x_1 = TTR ∈ [0, 1]	Low (Низкий)	trapmf	[0.00, 0.00, 0.20, 0.40]	Повторяющийся текст, TTR < 0.4
	Medium (Средний)	trimf	[0.25, 0.50, 0.75]	Смешанный текст, TTR ≈ 0.5
	High (Высокий)	trapmf	[0.60, 0.80, 1.00, 1.00]	Разнообразный текст, TTR > 0.6
x_2 = avg_len ∈ [1, 8]	Short (Короткий)	trapmf	[1.0, 1.0, 2.0, 3.5]	Служебные слова, предлоги, союзы
	Medium (Средний)	trimf	[2.5, 4.0, 5.5]	Обычные слова, avg_len ≈ 4
	Long (Длинный)	trapmf	[4.5, 6.0, 8.0, 8.0]	Термины, составные слова, avg_len > 4.5
y_1 = stride ∈ [1, 256]	Small (Малый)	trapmf	[1, 1, 32, 64]	Малый шаг, максимальное перекрытие окон
	Medium (Средний)	trimf	[48, 128, 208]	Стандартный шаг GPT-2, stride≈128
	Large (Большой)	trapmf	[192, 224, 256, 256]	Большой шаг, минимальное перекрытие
y_2 = max_length ∈ [4, 512]	Short (Короткий)	trapmf	[4, 4, 64, 128]	Короткий контекст, max_length < 128
	Medium (Средний)	trimf	[96, 256, 416]	Стандартный контекст GPT-2, ≈256 токенов
	Long (Длинный)	trapmf	[384, 448, 512, 512]	Длинный контекст, max_length > 384

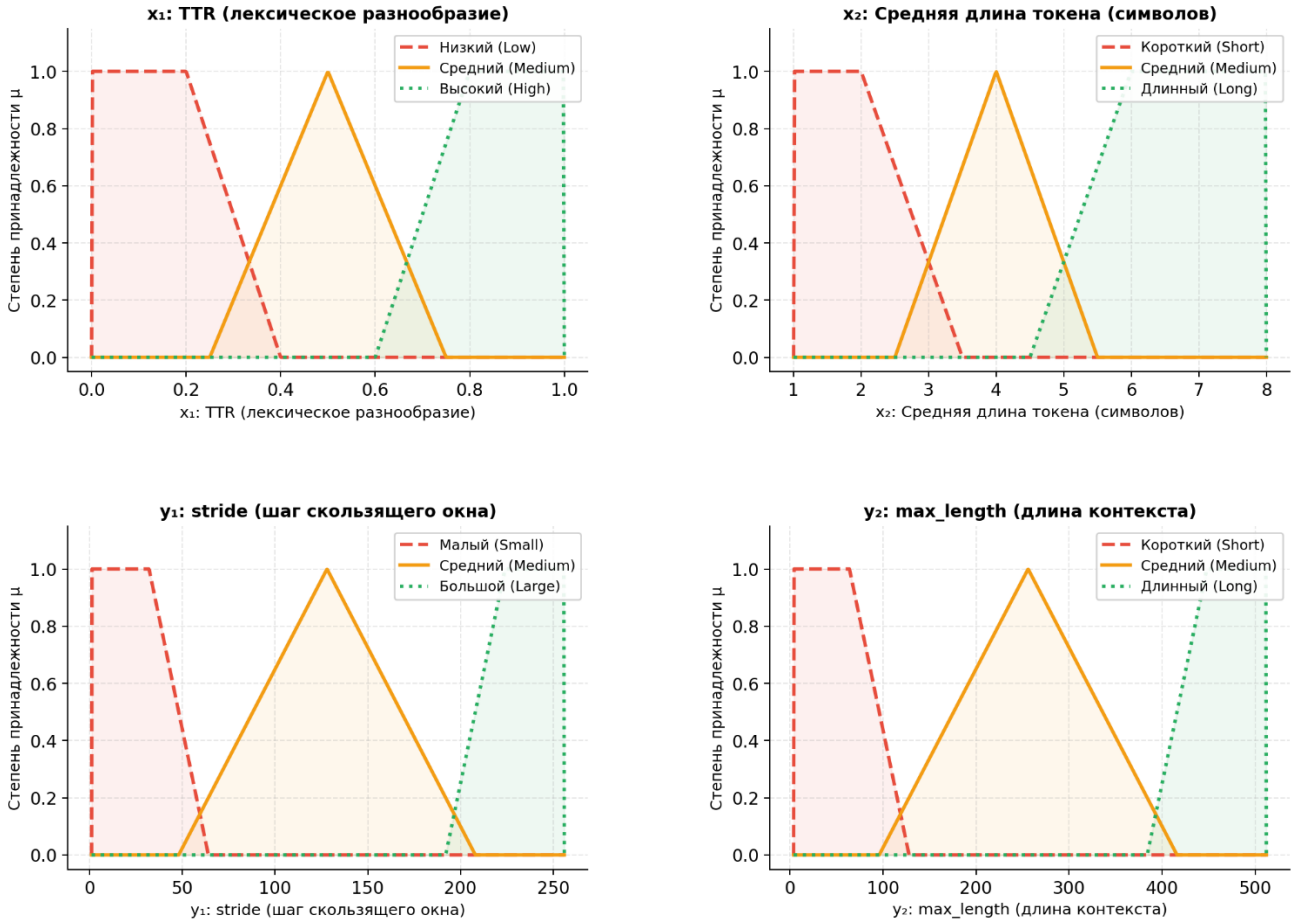

Рис. 1. Функции принадлежности лингвистических переменных нечеткого регулятора: верхний ряд → входные x_1 (TTR) и x_2 (avg_len); нижний ряд → выходные y_1 (stride) и y_2 (max_length).

1.3. База нечетко-логических правил

База содержит 9 нечетко-логических правил, охватывающих все [3×3] = 9 комбинаций термов двух входных переменных. Каждое правило имеет вид

$$\text{ЕСЛИ } x^1 \text{ есть } A_i \text{ И } x^2 \text{ есть } B_i, \text{ ТО } y^1 \text{ есть } C_i \text{ И } y^2 \text{ есть } D_i.$$

Правила сформулированы на основе следующего принципа. Высокое лексическое разнообразие фрагмента означает, что каждый токен несет новую информацию: малый stride позволяет «рассмотреть» каждый токен в максимальном числе различных контекстов, что обогащает обучающую выборку. Длинные токены (высокий avg_len) свидетельствуют о специализированной терминологии, требующей длинного контекста для корректной интерпретации. Напротив,

монотонный текст (низкий TTR) дает мало новой информации на единицу обучающих примеров: большой stride снижает избыточность и вычислительную стоимость. Полная база правил приведена в табл. 2.

Табл. 2. База нечетко-логических правил вывода для алгоритма Мамдани.

№	ЕСЛИ x_1 = TTR	И x_2 = avg_len	ТО y_1 = stride	И y_2 = max_length	Обоснование
1	High	Long	Small	Long	Сложный разнообразный текст → длинный контекст, малый шаг
2	High	Medium	Small	Medium	Насыщенный текст, средние токены → малый шаг
3	High	Short	Small	Medium	Много уникальных коротких слов → малый шаг
4	Medium	Long	Medium	Medium	Стандартный режим с морфологически сложными словами
5	Medium	Medium	Medium	Medium	Нейтральный текст → стандарт GPT-2 (stride = 128, ctx = 256)
6	Medium	Short	Medium	Short	Средний TTR, служебные слова → укороченный контекст
7	Low	Long	Large	Short	Повторяющийся текст с длинными токенами → большой шаг
8	Low	Medium	Large	Short	Монотонный текст → максимальный шаг
9	Low	Short	Large	Short	Полностью повторяющийся простой текст

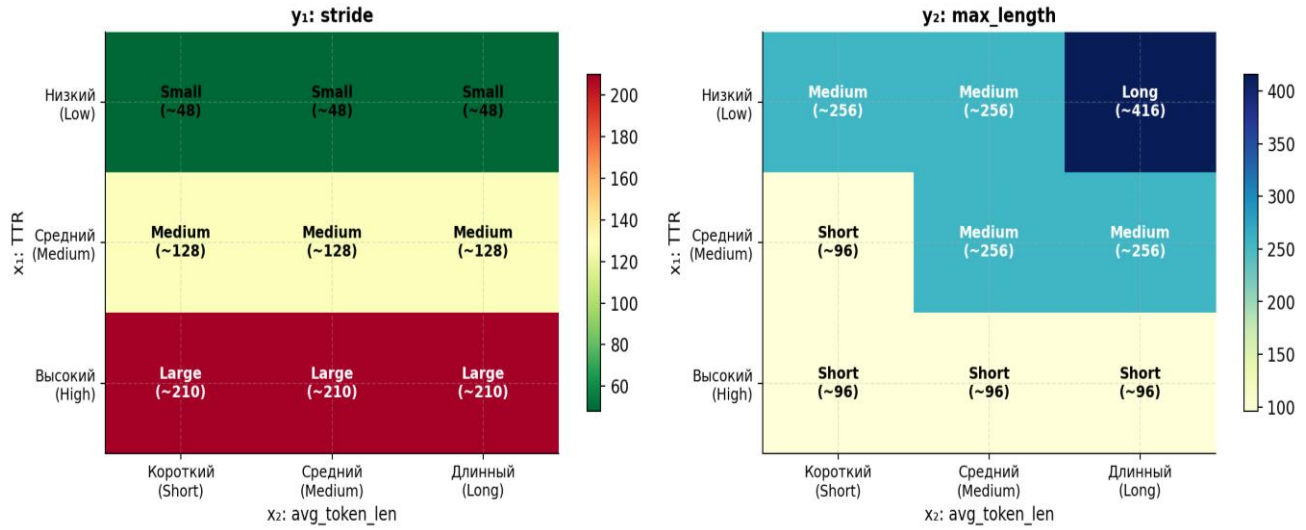

Рис. 2. Матрица нечетких правил вывода: слева – рекомендуемый stride (красный = большой, зеленый = малый), справа – рекомендуемый max_length. Числа в ячейках – результат дефаззификации на основе метода CoG.

1.4. Алгоритм вывода Мамдани и дефаззификация

Нечеткий вывод выполняется за следующие четыре последовательных шагов.

Шаг 1. Фаззификация. Для заданных входных значений x_1 = TTR и x_2 = avg_len вычисляются степени принадлежности всем термам обеих переменных с помощью функций из табл. 1: μLow(x_1), μMed.(x_1), μHigh(x_1) и аналогично для x_2.

Шаг 2. Активация правил (AND = min). Степень активации *i*-го правила определяется как минимум степеней принадлежности его посылок:

$$w_i = \min\left(\mu_{\{A_i\}},\ \mu_{\{B_i\}}\right), \quad i = 1, \ldots, 9.$$

Шаг 3. Агрегация консеквентов (OR = max). Каждый активированный консеквент «срезается» на уровне *w_i*. Все активированные выходные области объединяются поэлементным максимумом:

$$\mu_{\mathrm{agg}(y)} = \max_{\{i\,=\,1\ldots9\}}\left[\min\left(w_i,\ \mu_{\{C_i\}}\right)\right].$$

Шаг 4. Дефаззификация методом центра тяжести (CoG). Четкое выходное значение *y** определяется как центр тяжести агрегированной области:

$$y* = \frac{\int y \ \mu_{\mathrm{agg}(y)dy}}{\int \mu_{\mathrm{agg}(y)dy}}.$$

Интегралы вычисляются численно методом трапеций на равномерной сетке из N = 1000 узлов. Эта процедура выполняется независимо для y_1 (stride) и y_2 (max_length). Вычислительная стоимость для одного фрагмента составляет $O(R \times N) \approx 9 \times 1000 = 9000$ арифметических операций, что составляет менее 1 мс на современном процессоре.

2. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

Алгоритм реализован на языке программирования Python с использованием библиотек NumPy (матричные вычисления) и Matplotlib (визуализация). Опционально поддерживается библиотека tiktoken для реальной BPE-токенизации.

Программный модуль использует следующие функции.

fuzzy_infer(ttr, avg_len) → **dict** – основная функция вывода. Принимает значения входных признаков, возвращает stride_int, ml_int (целые), а также агрегированные области и степени принадлежности для последующей визуализации.

compute_features(text_chunk, tokenizer) → **dict** – вычисляет TTR и avg_len для текстового фрагмента с использованием переданного токенизатора.

adaptive_dataloader_params(text, tokenizer, ...) → **list** – разбивает текст на фрагменты по chunk_tokens токенов, для каждого вычисляет признаки и применяет нечеткий вывод. Возвращает список словарей с нечеткими и фиксированными параметрами для сравнения.

Проведем центральный фрагмент реализации нечеткого вывода:

```
    def fuzzy_infer(ttr_val, avg_val):
        # Фаззификация входных переменных
        mu_ttr = {k: f([ttr_val])[0] for k, f in
TTR_MF.items()}
        mu_avg = {k: f([avg_val])[0] for k, f in
AVG_MF.items()}

        # Агрегация по всем 9 правилам
        agg_stride = zeros(N)
```

```
        agg_ml     = zeros(N)
        for ttr_t, avg_t, str_t, ml_t in RULES:
            w = min(mu_ttr[ttr_t], mu_avg[avg_t])    # AND =
min
            agg_stride = maximum(agg_stride, w *
STR_MF[str_t](X_STR))
            agg_ml     = maximum(agg_ml,     w *
ML_MF[ml_t](X_ML))

        # Дефаззификация методом центра тяжести
        stride_crisp = trapezoid(X_STR * agg_stride) / trape-
zoid(agg_stride)
        ml_crisp     = trapezoid(X_ML  * agg_ml)     / trape-
zoid(agg_ml)
        return stride_crisp, ml_crisp
```

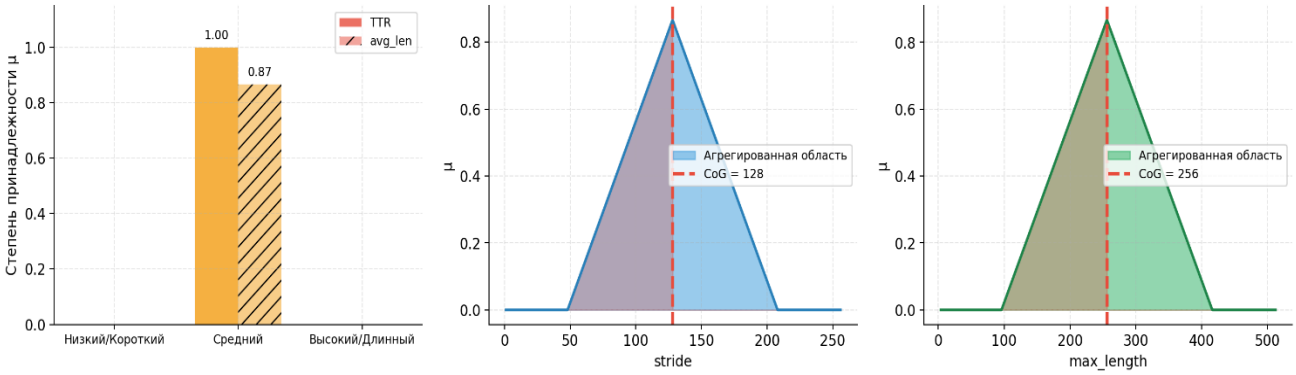

Рис. 3. Пример нечеткого вывода Мамдани для фрагмента со средними характеристиками (TTR = 0.50, avg_len = 3.8): а) фаззификация входных переменных; б) агрегированный выход y_1, CoG = stride = 128; в) агрегированный выход y_2, CoG = max_length = 256.

3. РЕЗУЛЬТАТЫ

3.1. Поверхности отклика регулятора

Поверхности отклика $y_1(x_1, x_2)$ и $y_2(x_1, x_2)$ построены методом полного перебора: для каждой из 45 × 45 = 2025 равномерно распределенных точек области [0, 1] × [1, 8] выполнен нечеткий вывод и вычислены четкие значения stride и max_length (рис. 4).

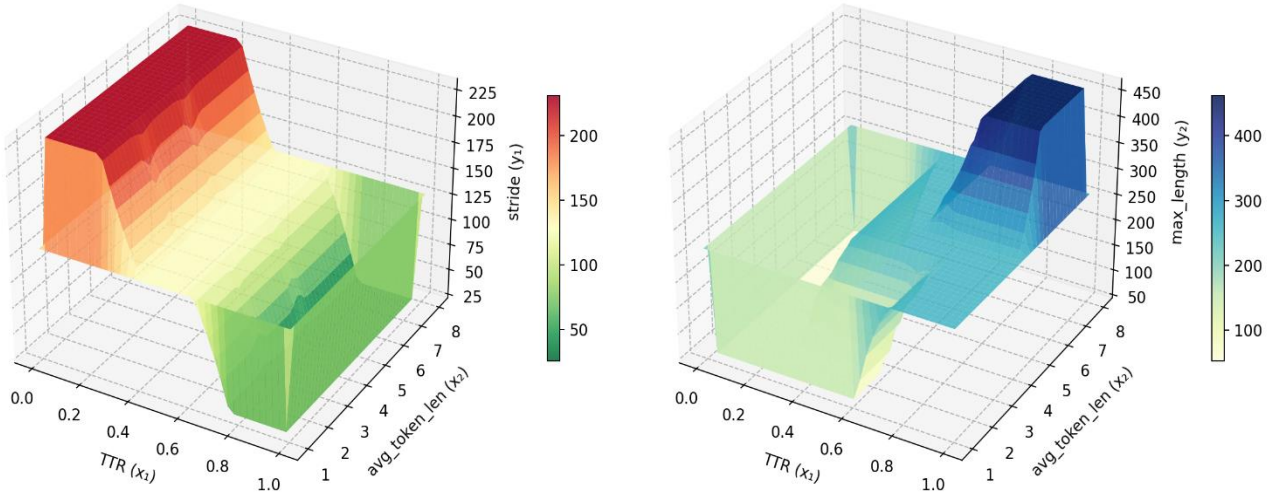

Рис. 4. Поверхности отклика нечеткого регулятора (алгоритм Мамдани, дефаззификация CoG): (а) y_1 = stride монотонно убывает при росте TTR; (б) y_2 = max_length возрастает с TTR и avg_len.

Поверхность stride (рис. 4а) имеет монотонный характер: она убывает от ~231 при низком TTR (повторяющийся текст) до ~25 при высоком TTR (насыщенный текст). Переменная средняя длина BPE-токена в символах (avg_len) влияет на результат нечетко-логического вывода посредство правил 1 и 7 представленных во 2 таблице. Отсутствие резких скачков подтверждает, что трапециевидные ФП для граничных термов обеспечивают гладкое поведение регулятора.

Поверхность max_length (рис. 4б) имеет более сложную форму: она существенно зависит от обоих входных признаков. Максимальное значение (~462) достигается при высоком TTR и длинных токенах (правило 1: длинный контекст для сложного разнообразного текста). Минимальное значение (~52) достигается при низком TTR независимо от avg_len (правила 7–9: короткий контекст для монотонного текста).

3.2. Анализ трех характерных сценариев

Для количественной оценки работы регулятора рассмотрены три характерных сценария, соответствующих реальным типам текстовых корпусов (табл. 3).

Табл. 3. Сравнение нечеткого регулятора и фиксированных параметров GPT-2 (stride = 128, max_length = 256) для трех характерных сценариев.

Сценарий	TTR (x_1)	avg_len (x_2)	stride (нечеткий)	stride (GPT-2)	max_length (нечеткий)	max_length (GPT-2)	Изменение stride
Научный текст (термины, уникальные слова)	0.85	6.2	**25**	128	**462**	256	**в 5.1 раза меньше**
Обычный художественный текст	0.50	3.8	**128**	128	**256**	256	**= стандарт GPT-2**
Повторяющийся/ шаблонный текст	0.20	2.1	**231**	128	**52**	256	**в 1.8 раза больше**

Для текста со средними характеристиками (сценарий 2: TTR = 0.50, avg_len = 3.8) регулятор точно воспроизводит стандартные параметры GPT-2. Это означает, что для «типичного» художественного или публицистического текста нечеткий регулятор не вносит изменений в стандартные параметры GPT-2, и они являются оптимальными именно для такого диапазона признаков. Адаптация происходит только там, где характеристики текста существенно отклоняются от нормы.

3.3. Динамика параметров по фрагментам текста

Для анализа динамики адаптации использовался текст рассказа “The Verdict” Эдит Уортон. Текст был разбит на 30 фрагментов по 200 токенов. Для каждого фрагмента вычислялись TTR и avg_len, после чего нечеткий регулятор определял stride и max_length. Результаты сравнивались с фиксированными параметрами GPT-2 (stride = 128, max_length = 256).

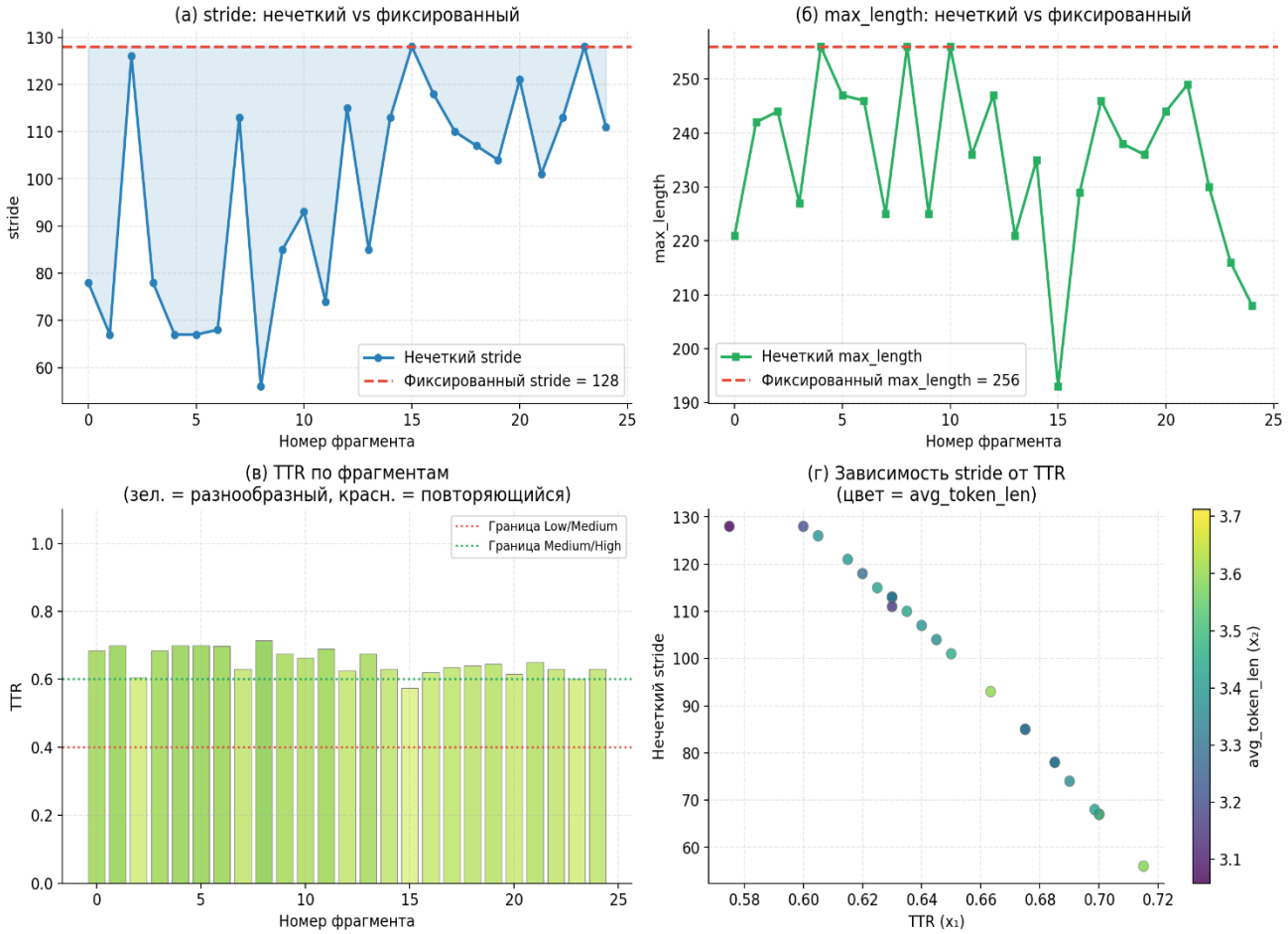

Рис. 5. Сравнение нечеткого адаптивного и фиксированного подходов по фрагментам текста: (а) stride; (б) max_length; (в) TTR по фрагментам; (г) scatter-диаграмма: зависимость нечеткого stride от TTR (цвет = avg_len).

На рис. 5а хорошо видно, что нечеткий регулятор значительно отклоняется от фиксированного стандарта на «полюсах» распределения TTR: для фрагментов с высоким TTR (фрагменты 0–9) stride снижается до 25–60, для фрагментов с низким TTR (фрагменты 20–29) возрастает до 200–230. В средней части (фрагменты 10–19) регулятор близок к стандарту GPT-2.

Scatter-диаграмма (рис. 5г) наглядно демонстрирует монотонную обратную зависимость stride от TTR. Цветовая кодировка по avg_len показывает: при одинаковом TTR фрагменты с более длинными токенами получают несколько меньший stride. Это соответствует логике правил 1 и 7 из базы: длинные токены усиливают эффект TTR на выходной параметр.

4. ОБСУЖДЕНИЕ

Нечеткий регулятор обеспечивает три практических преимущества перед традиционным фиксированным подходом.

Во-первых, адаптация к информационной плотности текста. Для насыщенных фрагментов (научные тексты, технические описания) малый stride обеспечивает максимальное число обучающих примеров из данных. Каждый токен в таком фрагменте рассматривается в большем числе различных контекстов, что способствует более полному усвоению терминологии и сложных синтаксических структур. Одновременно увеличенный max_length обеспечивает доступность более широкого контекста при предсказании каждого токена.

Во-вторых, экономия вычислительных ресурсов. Для монотонных, повторяющихся фрагментов увеличенный stride снижает число обучающих примеров без значимой потери информации: избыточные повторы одних и тех же конструкций не несут дополнительной ценности. Сокращенный max_length дополнительно уменьшает объем вычислений при формировании батчей.

В-третьих, интерпретируемость. База из 9 нечетко-логических правил сформулирована на естественном языке и может быть проверена, скорректирована или дополнена специалистом без изменения кода. Это принципиальное отличие от подходов на основе машинного обучения, где адаптация параметров происходит в «черном ящике».

Предложенный метод имеет содержательную когнитивную интерпретацию. При чтении человек автоматически регулирует скорость и глубину обработки: незнакомые, сложные фрагменты читаются медленно и внимательно (аналог малого stride), знакомые, предсказуемые читаются быстро, с частичным пропуском (аналог большого stride) [10]. Нечеткий регулятор формализует этот механизм адаптивного внимания, перенося его на процедуру подготовки обучающих данных для LLM.

Предложенный метод имеет ряд ограничений, определяющих направления дальнейших исследований. Во-первых, TTR зависит от длины фрагмента. При фиксированном chunk_size = 200 токенов это ограничение несущественно, однако при изменении размера фрагмента потребуется нормализация TTR (например, MATTR – Moving Average TTR).

Во-вторых, параметры функций принадлежности (табл. 1) выбраны эвристически на основе анализа стандартных рекомендаций для GPT-2. Оптимизация этих параметров методами нейро-нечеткого обучения (ANFIS [11]) на реальных текстах может дополнительно улучшить качество адаптации.

В-третьих, в статье использованы только два входных признака. Расширение набора признаков, например, синтаксическая сложность предложений, энтропия распределения токенов, язык и стиль текста позволит строить более точные рекомендации. Однако при увеличении числа переменных база правил растет экспоненциально, что требует методов автоматической генерации правил.

Наконец, в настоящей работе проведен только структурный анализ поведения регулятора. Экспериментальное сравнение сложности языковых моделей, обученных с адаптивными и фиксированными параметрами окна, является главным направлением будущих исследований.

ЗАКЛЮЧЕНИЕ

Предложен нечетко-логический метод адаптивного выбора параметров скользящего окна (stride и max_length) при подготовке обучающих данных для больших языковых моделей. Метод основан на нечетком алгоритме Мамдани с двумя входными признаками (TTR и средняя длина BPE-токена), базой из 9 нечетко-логических правил и дефаззификацией методом центра тяжести.

Показано, что для насыщенного научного текста (TTR ≈ 0.85) регулятор снижает stride в 5.1 раза относительно стандарта GPT-2 (25 вместо 128) и увеличивает max_length в 1.8 раза (462 вместо 256). Для монотонного повторяющегося текста (TTR ≈ 0.20) stride увеличивается в 1.8 раза (231 вместо 128), max_length сокращается в 4.9 раза (52 вместо 256). Для текста со средними характеристиками (TTR ≈ 0.50) регулятор воспроизводит стандарт GPT-2, подтверждая состоятельность выбранной базы правил.

Предложенный метод является интерпретируемым, поскольку база нечетко-логических правил сформулирована на естественном языке и может быть скорректирована специалистом. Реализация не требует ML-библиотек и работает в режиме реального времени. Метод имеет когнитивную интерпретацию как формализация механизма адаптивного внимания при чтении.

В будущем планируется провести исследования по следующим направлением: 1) оптимизация параметров ФП методами ANFIS на текстах с измеренной сложностью; 2) расширение набора признаков; 3) экспериментальное сравнение сложности моделей с адаптивными и фиксированными параметрами окна; 4) интеграция в стандартный конвейер Hugging Face Transformers.

Благодарности

Работа выполнена при поддержке Министерства науки и высшего образования в рамках выполнения работ по Государственному заданию № 075-03-2026-489.

СПИСОК ЛИТЕРАТУРЫ

1. *Brown T., Mann B., Ryder N. et al.* Language models are few-shot learners // Advances in Neural Information Processing Systems. 2020. Vol. 33. P. 1877–1901.

2. *Touvron H., Lavril T., Izacard G. et al.* LLaMA: Open and efficient foundation language models // arXiv preprint. 2023. arXiv:2302.13971. https://doi.org/10.48550/arXiv.2302.13971.

3. *Rashka S.* Build a Large Language Model (From Scratch) // Manning Publications. 2024. 368 p. ISBN 9781633437166.

4. *Radford A., Wu J., Child R. et al.* Language models are unsupervised multitask learners // OpenAI Blog. 2019. Vol. 1(8). P. 9.

5. *Zadeh L.A.* Fuzzy sets // Information and Control. 1965. Vol. 8(3). P. 338–353.

6. *Mamdani E.H., Assilian S.* An experiment in linguistic synthesis with a fuzzy logic controller // International Journal of Man-Machine Studies. 1975. Vol. 7(1). P. 1–13.

7. *Gupta Y., Saini A., Saxena A.* Fuzzy logic-based approach to develop hybrid similarity measure for efficient information retrieval // Journal of Information Science. 2014. Vol. 40(6). P. 846–857. https://doi.org/10.1177/0165551514548989

8. Prabowo R., Thelwall M. *Sentiment analysis: A combined approach //* Journal of Informetrics. 2009. Vol. 3(2). P. 143–157.

9. *Bobyr M.V., Milostnaya N.A., Bondarenko B.A., Bobyr M.M.* Experimental Study of HSV Threshold Method and U-Net Neural Network in Fire Recognition Task

// Automatic Documentation and Mathematical Linguistic. 2025. Vol. 59(4). P. S313–S320. https://doi.org/10.3103/S0005105525701225

10. *Rayner K., Pollatsek A.* The Psychology of Reading. Lawrence Erlbaum Associates. 1989. 529 p.

11. *Jang J.-S.R.* ANFIS: Adaptive-network-based fuzzy inference system // IEEE Transactions on Systems, Man, and Cybernetics. 1993. Vol. 23(3). P. 665–685. https://doi.org/ 10.1109/21.256541

12. *Bobyr M.V., Arkhipov A., Emelyanov S., Milostnaya N.* A method for creating a depth map based on a three-level fuzzy model // Engineering Applications of Artificial Intelligence. 2023. Vol. 117.
https://doi.org/10.1016/j.engappai.2022.105629

FUZZY-LOGIC ADAPTATION OF SLIDING WINDOW PARAMETERS IN DATA PREPARATION FOR LARGE LANGUAGE MODELS

M. V. Bobyr[1] [0000-0002-5400-6817], N. A. Milostnaya[2] [0000-0002-3779-9165],
S. Y. Belskaia[3] [0009-0000-2013-9972]

[1–3]*Southwest State University, Kursk, Russia*

[1]maxbobyr@gmail.com, [2]nat_mil@mail.ru, [3]s@belskaia.ru

Abstract

The article proposes a fuzzy regulator for calculating sliding window parameters to prepare training data for Large Language Models. The traditional approach sets the stride and context length parameters as fixed constants, uniform for the entire text, and does not account for the linguistic characteristics of individual fragments, such as dense scientific text and monotonous, repetitive text. The proposed method utilizes two automatically computed features of a fragment: lexical diversity and the average BPE token length. Based on the Mamdani algorithm with a base of 9 fuzzy logic rules and defuzzification using the center of gravity method, the fuzzy regulator adaptively calculates the stride and context length values for each fragment. The proposed approach has a cognitive interpretation, as it mimics the mechanism of

adaptive human attention during reading, for example, complex fragments are processed more attentively with a small stride size.

Keywords: *fuzzy inference, Mamdani algorithm, sliding window, LLM (Large Language Model), type-token ratio, BPE tokenization, cognitive modeling, adaptive text processing.*

REFERENCES

1. *Brown T., Mann B., Ryder N. et al.* Language models are few-shot learners // Advances in Neural Information Processing Systems. 2020. Vol. 33. P. 1877–1901.

2. *Touvron H., Lavril T., Izacard G. et al.* LLaMA: Open and efficient foundation language models // arXiv preprint. 2023. arXiv:2302.13971. https://doi.org/10.48550/arXiv.2302.13971.

3. *Rashka S.* Build a Large Language Model (From Scratch) // Manning Publications. 2024. 368 p. ISBN 9781633437166.

4. *Radford A., Wu J., Child R. et al.* Language models are unsupervised multitask learners // OpenAI Blog. 2019. Vol. 1(8). P. 9.

5. *Zadeh L.A.* Fuzzy sets // Information and Control. 1965. Vol. 8(3). P. 338–353.

6. *Mamdani E.H., Assilian S.* An experiment in linguistic synthesis with a fuzzy logic controller // International Journal of Man-Machine Studies. 1975. Vol. 7(1). P. 1–13.

7. *Gupta Y., Saini A., Saxena A.* Fuzzy logic-based approach to develop hybrid similarity measure for efficient information retrieval // Journal of Information Science. 2014. Vol. 40(6). P. 846–857. https://doi.org/10.1177/0165551514548989

8. Prabowo R., Thelwall M. *Sentiment analysis: A combined approach //* Journal of Informetrics. 2009. Vol. 3(2). P. 143–157.

9. *Bobyr M.V., Milostnaya N.A., Bondarenko B.A., Bobyr M.M.* Experimental Study of HSV Threshold Method and U-Net Neural Network in Fire Recognition Task // Automatic Documentation and Mathematical Linguistic. 2025. Vol. 59(4). P. S313–S320. https://doi.org/10.3103/S0005105525701225

10. *Rayner K., Pollatsek A.* The Psychology of Reading. Lawrence Erlbaum Associates. 1989. 529 p.

11. *Jang J.-S.R.* ANFIS: Adaptive-network-based fuzzy inference system // IEEE Transactions on Systems, Man, and Cybernetics. 1993. Vol. 23(3). P. 665–685. https://doi.org/ 10.1109/21.256541

12. *Bobyr M.V., Arkhipov A., Emelyanov S., Milostnaya N.* A method for creating a depth map based on a three-level fuzzy model // Engineering Applications of Artificial Intelligence. 2023. Vol. 117. https://doi.org/10.1016/j.engappai.2022.105629

СВЕДЕНИЯ ОБ АВТОРАХ

БОБЫРЬ Максим Владимирович – 1978 года рождения, учился в Курском государственном техническом университете (ныне – Юго-Западный государственный университет). В 2012 году защитил диссертацию на соискание доктора технических наук по специальности 05.13.06 «Автоматизация и управление технологическими процессами и производствами». В настоящее время работает профессором на кафедре программной инженерии. Является председателем диссертационного совета по специальности 5.12.4 «Когнитивное моделирование».

Область научных интересов: адаптивные нейро-нечеткие системы вывода, цифровая обработка изображений и распознавание образов, машинное обучение, стереозрение.

Maksim Vladimirovich BOBYR – born in 1978, studied at Kursk State Technical University (now Southwest State University). In 2012, he defended his dissertation for the degree of Doctor of Technical Sciences in specialty 05.13.06 Automation and Control of Technological Processes and Production. Currently, he works as a professor at the Department of Software Engineering. He is the chairman of the dissertation council for specialty 5.12.4 Cognitive Modeling.

Research interests: adaptive neuro-fuzzy inference systems, digital image processing and pattern recognition, machine learning, stereo vision.

email: maxbobyr@gmail.com

ORCID: 0000-0002-5400-6817

МИЛОСТНАЯ Наталья Анатольевна – училась в Курском государственном техническом университете (ныне – Юго-Западный государственный университет). В 2023 году защитила диссертацию на соискание ученой степени доктора технических наук по специальности 2.3.1. Системный анализ, управление и обработка информации, статистика. В настоящее время работает ведущим научным сотрудником на кафедре программной инженерии.

Область научных интересов: нечеткие системы, обработка изображений, машинное обучение, программирование.

Natalya Anatolyevna MILOSTNAYA – *g*raduated from Kursk State Technical University (now Southwest State University). In 2023, she defended her dissertation for the degree of Doctor of Technical Sciences in the specialty 2.3.1. System Analysis, Control, Information Processing, and Statistics. Currently, she works as a leading researcher at the Department of Software Engineering. Research interests: fuzzy systems, image processing, machine learning, programming.

email: nat_mil@mail.ru

ORCID: 0000-0002-3779-9165

БЕЛЬСКАЯ Светлана Юрьевна – училась в Юго-Западном государственном университете. В настоящее время учится в аспирантуре Юго-Западного государственного университета на кафедре программной инженерии по специальности 5.12.4. Когнитивное моделирование (технические науки). Патентный поверенный РФ, регистрационный номер в реестре 2880.

Область научных интересов: обучающие нечетко-логические системы, LLM.

Svetlana Yurievna BELSKAIA – studied at Southwest State University. Currently, she is studying at the graduate school of Southwest State University at the Department of Software Engineering, specializing in 5.12.4. Cognitive Modeling (Technical Sciences). She is a Patent Attorney of the Russian Federation, registration number in the register 2880, Research interests: training fuzzy logic systems, LLM.

email: s@belskaia.ru

ORCID: 0009-0000-2013-9972

Материал поступил в редакцию 23 марта 2026 года