Метрики корректности кода на основе логарифмов вероятностей токенов языковых моделей

Д. В. Федрушков, И. З. Смирнов, С. В. Ковальчук, А. А. Алиев

***Аннотация* — В данной работе исследуются вероятностные характеристики языковых моделей как инструмент оценки корректности программного кода. Предлагаемый подход основан на анализе логарифмов вероятностей токенов, вычисляемых моделью для сгенерированных решений, и не требует наличия референсных решений или выполнения кода. Исследуется связь вероятностных метрик с функциональной корректностью решений, определяемой автоматическим тестированием, с использованием ранговой корреляции, ROC-анализа и логистической регрессии. Рассматриваются различные способы агрегирования логарифмов вероятностей токенов, а также влияние длины решения и энтропии распределения токенов. Для контроля влияния длины используются парное сравнение решений близкой длины и частная ранговая корреляция. Эксперименты на наборе HumanEval-X для пяти языков программирования и пяти открытых моделей с числом параметров до 16 млрд показывают, что вероятностные метрики позволяют различать корректные и некорректные решения и имеют статистически значимую связь с показателем pass@1. После контроля длины разделяющая способность метрик сохраняется, а различие между самооценкой и перекрёстной оценкой существенно уменьшается, тогда как энтропия распределения токенов демонстрирует сопоставимую разделяющую способность.**

***Ключевые слова* — искусственный интеллект, языковые модели, генерация программного кода, метрики, распределение вероятностей, оценка качества, корректность кода.**

I. Введение

Большие языковые модели (БЯМ) широко применяются в задачах, связанных с разработкой, таких как генерация кода на основании описаний на естественном языке, дополнение фрагментов кода и исправление ошибок в коде [1]. Успешная генерация кода позволяет улучшить продуктивность и качество программной разработки [2]. Однако оценка генеративных моделей является сложной задачей, так как метрики, широко применяемые для автоматизированной оценки текстов, такие как BLEU [3], ROUGE [4], имеют низкую корреляцию с человеческой оценкой и требуют наличия референса для сравнения [5]. Одним из способов автоматизированной оценки БЯМ без необходимости наличия референса является оценка при помощи отдельной языковой модели-оценщика (llm-as-a-judge) [6], такой подход позволяет повышать точность ответов модели в разных областях знаний. Несмотря на это, модели-оценщики основываются на промптах (текстовых инструкциях), что делает их неустойчивыми к изменениям инструкций [5]. Также модели-оценщики нередко испытывают трудности при оценке семантической и синтаксической корректности кода [7]. Для анализа синтаксической и семантической корректности применяются такие метрики, как CodeBLEU [8], который является адаптацией тексто-ориентированного подхода и метрика на базе модели CodeBERT [9], обученная специально для работы с программным кодом. Однако сгенерированный код, несмотря на высокие показатели по специализированным метрикам, может не проходить тесты [10].

В настоящей работе предлагается исследовать вероятностные характеристики языковых моделей, не требующие отдельно сформулированных инструкций для оценивания, референсного решения, исполнения кода или дополнительных вызовов модели. Для этого рассматриваются логарифмы вероятностей, которые языковая модель присваивает токенам сгенерированного решения. В работе рассматриваются различные варианты агрегирования этих значений по последовательности токенов и анализируется их связь с результатами функционального тестирования.

II. Подходы к оценке корректности кода

Существующие подходы к оценке сгенерированного кода можно условно разделить на четыре группы: метрики текстового сходства, методы функционального тестирования, статический анализ кода и оценка с помощью моделей-экспертов.

Метрики текстового сходства (например, BLEU [3], ROUGE [4]) оценивают близость сгенерированного решения к референсному на уровне n-грамм. Расширенные подходы, такие как CodeBLEU [8], учитывают синтаксические структуры и зависимости данных, а методы на основе обученных представлений (например, CodeBERTScore [11]) используют контекстуальные эмбеддинги. Тем не менее, все такие метрики требуют референсного решения и не гарантируют функциональной корректности кода [12].

Функциональное тестирование определяет корректность на основе выполнения кода и является одной из наиболее надёжных метрик, поскольку

Статья получена 5 октября 2026.

напрямую отражает соответствие решения поставленной задаче. Однако наличие тестов не гарантировано, а их разработка и автоматическая генерация связаны с существенными затратами.

Статический анализ позволяет выявлять ошибки без выполнения кода, но требует настройки правил и плохо масштабируется на сложные проекты [13].

Альтернативой является подход LLM-as-Judge [14, 15], в котором корректность решения оценивается языковой моделью без использования тестов и референсов. Такой подход обеспечивает гибкую семантическую оценку, однако его результаты зависят от формулировки инструкций и параметров генерации, что снижает воспроизводимость.

В данной работе рассматривается подход, основанный на вероятностных характеристиках языковой модели. В отличие от LLM-as-Judge, он не требует отдельных инструкций и не зависит от параметров генерации: оценка определяется логарифмами вероятности последовательности, фиксированными параметрами модели. Это позволяет рассматривать logprob как устойчивый и интерпретируемый сигнал качества, не требующий референсов или выполнения кода.

III. Логарифмы вероятностей текстовых последовательностей

Для генерации текстовых последовательностей в языковых моделях используется механизм правдоподобия, определяющий вероятность появления следующего слова (токена) из словаря языковой модели в зависимости от последовательности предыдущих токенов. Пусть V обозначает словарь модели, x обозначает исходную инструкцию (условие задачи или текст запроса), предъявляемую языковой модели, а $y = (y_1, y_2, \dots, y_T)$ — последовательность токенов, сгенерированную моделью в качестве продолжения этой инструкции.

Вероятность появления очередного токена y_t определяется с помощью функции softmax, применяемой к «логитам» модели:

$$P(y_t|x, y_{<t}) = \frac{\exp(l_{y_t})}{\sum_{v \in V} \exp(l_v)} \quad (1)$$

где l_v обозначает «логит» токена v, то есть значение выхода модели до применения функции softmax, определяющее относительную вероятность выбора соответствующего токена, а $y_{<t} = (y_1, \dots, y_{t-1})$— префикс уже сгенерированной последовательности.

Вероятность всей последовательности токенов определяется как произведение условных вероятностей отдельных токенов:

$$P(y|x) = \prod_{t=1}^{T} P(y_t|x, y_{<t}) \quad (2)$$

где T — итоговая длина последовательности токенов.

На практике вероятности отдельных токенов могут принимать очень малые значения, поэтому при перемножении вероятностей возникает риск численного переполнения и потери точности вычислений. Для повышения устойчивости вычислений вероятности обычно рассматриваются в логарифмической форме. Логарифм вероятности токена определяется следующим образом:

$$\log P(y_t|x, y_{<t}) = l_{y_t} - \log \sum_{v \in V} \exp(l_v) \quad (3)$$

Логарифм вероятности всей последовательности выражается следующим образом:

$$S_{\text{sum}}(y) = \sum_{t=1}^{T} \log P(y_t|x, y_{<t}) \quad (4)$$

Как правило, более длинные последовательности имеют меньший суммарный логарифм вероятности. Использование нормализованного варианта логарифма вероятности позволяет частично компенсировать этот эффект и облегчает сопоставление последовательностей различной длины:

$$S_{\text{norm}}(y) = \frac{S_{\text{sum}}(y)}{T^{\alpha}} \quad (5)$$

Нормализованные вероятностные оценки широко используются при выборе наиболее правдоподобного ответа, генерируемого языковой моделью. При сравнении нескольких возможных последовательностей обычно рассматривается задача выбора последовательности с максимальным логарифмом вероятности (или его нормализованного варианта). Данный принцип лежит в основе алгоритмов декодирования языковых моделей, таких как жадный поиск (greedy search) и лучевой поиск (beam search), где итоговая последовательность выбирается на основе максимизации вероятности [16].

Из того же распределения можно получить и вторую характеристику. Выражение (3) использует вероятность одного токена — того, который фактически присутствует в оцениваемой последовательности. Само же распределение задано на всём словаре, и его неопределённость в позиции t описывается энтропией, усреднённой по последовательности:

$$H(y) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{v \in V} P(v|x, y_{<t}) \log P(v|x, y_{<t}) \quad (6)$$

Обе величины строятся из одних и тех же логарифмов, но усредняются по-разному: выражения (4) и (5) — по фактической последовательности токенов, выражение (6) — по распределению модели. Поэтому энтропия не зависит от того, какой токен стоит в решении, и характеризует не согласие ответа с ожиданием модели, а степень определённости самой модели относительно возможных продолжений в данном контексте.

Таким образом, вероятность всей последовательности в логарифмической форме отражает степень согласованности ответа с распределением, усвоенным языковой моделью, относительно поставленной задачи. Это позволяет выдвинуть предположение, что потенциальное решение какой-либо задачи может быть оценено языковой моделью на предмет корректности через логарифм вероятности появления данного решения. Далее вероятностные характеристики рассматриваются как возможные метрики для оценки корректности решений.

IV. Вероятность последовательности как метрика

A. *Описание эксперимента*

Для проверки гипотезы о связи вероятностных характеристик языковых моделей с корректностью решений был проведён эксперимент на наборе задач по программированию на нескольких языках. Для каждой

задачи генерировались решения с помощью нескольких языковых моделей, после чего корректность результатов проверялась с использованием автоматических тестов. Полный цикл эксперимента изображён на рис. 1.

Рис. 1. Схема экспериментального пайплайна оценки решений с использованием логарифмов вероятности и автоматического тестирования

Набор данных для генерации решений и проверки функциональной корректности HumanEval-X [17, 18] содержит по 164 задачи на программирование, представленных на пяти языках программирования — Python, Java, Go, JavaScript и C++.

Для генерации решений использовались пять языковых моделей: Qwen3 8B [19], Llama3.1 8B [20], DeepSeek-Coder-v2 16B [21], Gemma3 12B [22] и Phi-4 14B [23]. В эксперимент были включены открытые модели с числом параметров до 16 миллиардов, что соответствует распространённому классу моделей, доступных для практического использования без специализированных вычислительных ресурсов.

Для каждого сгенерированного решения были вычислены вероятностные характеристики последовательности токенов по отношению к исходной задаче. В частности, для каждой пары «задача–решение» рассчитывались логарифмы вероятностей токенов ответа, после чего агрегировались следующие метрики: суммарный логарифм вероятности S_{sum}, а также его нормализованные варианты S_{avg} и S_{norm}, получаемые из выражения (5) при различных значениях параметра α. В частности, при $\alpha = 1$ формула (5) соответствует среднему логарифму вероятности S_{avg}, тогда как при $\alpha = 0.5$ используется мягкая нормализация S_{norm}.

Для анализа связи между логарифмами вероятностей языковых моделей и фактической корректностью решений использовались следующие статистические методы:

— коэффициент ранговой корреляции Спирмена для оценки монотонной зависимости между значением вероятностной метрики и показателем корректности решений;

— модель логистической регрессии, позволяющая оценить влияние метрики на вероятность успешного прохождения тестов;

— ROC-анализ для анализа способности метрик различать корректные и некорректные решения.

Отдельную трудность представляет длина решения. Суммарный логарифм вероятности по построению убывает с ростом числа токенов, а сама длина связана с вероятностью прохождения тестов, поскольку длинные решения чаще соответствуют более сложным задачам. Поэтому часть наблюдаемой связи между метрикой и корректностью может принадлежать не метрике, а длине. Чтобы оценить влияние длины, все сравнительные величины вычислялись в двух вариантах: без контроля длины и с контролем.

Для ROC-анализа контроль выполняется парным сравнением. Рассматриваются пары решений на одном языке, из которых одно прошло тесты, а другое нет, и длины которых различаются не более чем на 5 %. Доля пар, в которых метрика выше у прошедшего решения, обозначается далее AUC_L и имеет тот же смысл, что и величина (10), но вычисляется внутри узкой полосы длины, где длина сама по себе уже не может служить признаком. Пары отбираются случайно, по миллиону попыток на оценку: последовательный перебор сместил бы выборку в сторону коротких решений.

Для корреляции контроль выполняется переходом к частному коэффициенту. Ранги метрики и ранги метки прохождения тестов освобождаются от линейной связи с рангом длины, после чего коэффициент вычисляется между остатками. В отличие от парной процедуры здесь используются все наблюдения, дополнительно контроль проверен расслоением корпуса по длине решения, чтобы с большей точностью исключить её влияние.

Коэффициент ранговой корреляции Спирмена определяется следующим образом:

$$\rho = \frac{\sum_{i=1}^{n}(R_i-\bar{R})(S_i-\bar{S})}{\sqrt{\left(\sum_{i=1}^{n}(R_i-\bar{R})^2\right)\left(\sum_{i=1}^{n}(S_i-\bar{S})^2\right)}} \quad (7)$$

Коэффициент корреляции отражает наличие монотонной зависимости между переменными, однако не позволяет количественно оценить связь между значением метрики и вероятностью корректного решения. Для такой оценки была использована модель логистической регрессии следующего вида:

$$P(z = 1|S) = \frac{1}{1+\exp(-(\beta_0+\beta_1 S))} \quad (8)$$

Для анализа разделяющей способности метрик использовался ROC-анализ, основанный на зависимости истинно положительной доли от ложноположительной при изменении порога классификации. ROC-кривая определяется следующим образом:

$$\text{ROC}(\tau) = \left(\frac{\text{FP}(\tau)}{\text{FP}(\tau)+\text{TN}(\tau)}, \frac{\text{TP}(\tau)}{\text{TP}(\tau)+\text{FN}(\tau)}\right) \quad (9)$$

Качество разделения классов обычно характеризуется площадью под ROC-кривой (AUC), которая может быть интерпретирована как вероятность того, что корректное решение получит более высокую оценку метрики, чем некорректное:

$$\text{AUC} = P(S(x^+) > S(x^-)) \quad (10)$$

Наконец, следует определить единицу наблюдения и способ оценки точности. Логарифмы вычисляются для пары «задача–решение», поэтому наблюдением является решение: ему отвечает одна метка прохождения тестов, общая для всех его оценок, и оценки одного решения независимыми наблюдениями не являются. Зависимы и сами решения: каждое условие задачи переведено на все рассматриваемые языки и предъявлено всем генераторам, поэтому его трудность переносится и между языками, и между генераторами. Единицей группировки поэтому выбрано условие задачи — верхний уровень вложенности, включающий язык, генератор и решение. Доверительные интервалы и уровни значимости для ранговых корреляций рассчитываются методом повторных выборок с извлечением кластеров целиком, для коэффициентов логистической регрессии — по робастной оценке ковариационной матрицы с той же кластеризацией; точечные оценки от выбора единицы группировки не зависят. Сопоставление самооценки с перекрёстной оценкой выполняется, напротив, на уровне отдельной оценки, поскольку это различие существует только там.

B. Результаты эксперимента

Доля решений, прошедших тестирование, представлена в табл. 1.

Представленные результаты демонстрируют заметные различия в качестве генерации между рассматриваемыми моделями. В частности, модель Gemma3 достигает наилучших значений показателя pass@1, тогда как менее производительные модели демонстрируют существенно более низкие результаты. Эти различия служат отправной точкой для анализа связи вероятностных оценок моделей с корректностью решений.

Таблица 1 — Доля корректных решений в зависимости от модели-генератора и языка программирования

Генератор	**Python**	**Java**	**Go**	**JavaScript**	**C++**	**Всего**
Qwen3	0,646	0,659	0,366	0,628	0,530	0,566
Llama3.1	0,329	0,280	0,220	0,250	0,256	0,267
DeepSeek	0,329	0,280	0,274	0,305	0,323	0,302
Gemma3	0,793	0,726	0,579	0,738	0,622	0,691
Phi-4	0,640	0,640	0,506	0,610	0,524	0,584

Средние значения указанных метрик в зависимости от языка программирования и используемой языковой модели приведены в табл. 2.

Как видно из табл. 2, значения логарифмов вероятности варьируются в зависимости как от используемой модели, так и от языка программирования. Наблюдаются различия в масштабе значений между языками, что может быть связано с особенностями распределения длины и структуры сгенерированных последовательностей.

Таблица 2 — Средние значения вероятностных метрик в зависимости от языка программирования и модели-оценщика

Метрика	**Язык**	**Qwen3**	**Llama3.1**	**DeepSeek**	**Gemma3**	**Phi-4**
Энтропия	Python	0,20	0,48	0,64	0,16	0,44
	Java	0,22	0,45	0,43	0,10	0,37
	Go	0,24	0,54	0,58	0,18	0,45
	JavaScript	0,31	0,54	0,55	0,14	0,49
	C++	0,29	0,54	0,42	0,15	0,48
S_{avg}	Python	-1,16	-1,33	-0,77	-1,52	-1,25
	Java	-0,46	-0,46	-0,40	-0,51	-0,50
	Go	-0,73	-0,74	-0,65	-0,91	-0,87
	JavaScript	-0,60	-0,63	-0,50	-0,77	-0,65
	C++	-0,54	-0,52	-0,36	-0,51	-0,55
S_{sum}	Python	-32,62	-43,08	-36,23	-45,73	-38,71
	Java	-46,00	-48,90	-42,19	-61,86	-49,19
	Go	-44,50	-45,87	-45,61	-54,17	-47,12
	JavaScript	-40,32	-43,07	-36,38	-55,09	-43,87
	C++	-42,66	-43,56	-33,00	-52,67	-44,25
S_{norm}	Python	-4,83	-5,98	-4,40	-6,34	-5,46
	Java	-4,26	-4,44	-3,85	-5,19	-4,59
	Go	-5,01	-4,97	-4,76	-5,80	-5,38
	JavaScript	-4,51	-4,71	-3,94	-5,81	-4,86
	C++	-4,50	-4,46	-3,21	-4,84	-4,59

Значения коэффициента корреляции между вероятностными метриками и показателем pass@1 представлены в табл. 3. В ячейках указаны обычный коэффициент и частный коэффициент при контроле длины решения. Звёздочкой отмечены значения, значимые на уровне $p < 0{,}001$

Полученные значения коэффициента корреляции указывают на наличие статистически значимой зависимости между логарифмами вероятностей языковых моделей и корректностью решений. Наиболее устойчивые значения корреляции наблюдаются для нормализованной метрики S_{norm},

что свидетельствует о важности учёта длины последовательности при анализе вероятностных характеристик ответов. При этом связь выражена слабее в случаях, когда языковая модель оценивает собственные решения; природа этого различия рассматривается далее.

Таблица 3 — Коэффициент ранговой корреляции Спирмена между значением метрики и показателем pass@1 в зависимости от модели-генератора и модели-оценщика.

Предиктор	**Оценщик**	**Qwen3**	**Llama3.1**	**DeepSeek**	**Gemma3**	**Phi-4**	**Всего**
Длина, токены	—	0,191* / —	0,295* / —	0,166* / —	0,302* / —	0,144* / —	0,177* / —
Энтропия	Qwen3	0,272* / 0,307*	0,224* / 0,277*	0,236* / 0,290*	0,071 / 0,162*	0,181* / 0,246*	0,280* / 0,333*
	Llama3.1	0,264* / 0,284*	0,090 / 0,250*	0,139* / 0,259*	0,022 / 0,190*	0,086 / 0,190*	0,152* / 0,238*
	DeepSeek	0,273* / 0,279*	0,125* / 0,273*	0,132* / 0,292*	0,045 / 0,241*	0,116 / 0,232*	0,158* / 0,249*
	Gemma3	0,361* / 0,340*	0,349* / 0,364*	0,333* / 0,365*	0,245* / 0,324*	0,281* / 0,317*	0,369* / 0,386*
	Phi-4	0,251* / 0,307*	0,152* / 0,288*	0,162* / 0,275*	0,012 / 0,208*	0,117 / 0,248*	0,208* / 0,312*
S_{avg}	Qwen3	0,085 / 0,283*	0,090 / 0,327*	0,140* / 0,338*	0,044 / 0,289*	0,146* / 0,329*	0,208* / 0,401*
	Llama3.1	0,186* / 0,265*	-0,106 / 0,186*	0,068 / 0,298*	-0,054 / 0,227*	0,027 / 0,199*	0,032 / 0,189*
	DeepSeek	0,294* / 0,268*	0,186* / 0,278*	0,221* / 0,323*	0,159* / 0,248*	0,208* / 0,271*	0,210* / 0,250*
	Gemma3	0,228* / 0,330*	0,092 / 0,341*	0,175* / 0,399*	-0,187* / 0,170*	0,103 / 0,293*	0,221* / 0,388*
	Phi-4	0,204* / 0,324*	0,028 / 0,327*	0,118 / 0,392*	-0,060 / 0,308*	-0,020 / 0,228*	0,137* / 0,359*
S_{sum}	Qwen3	0,218* / 0,113	0,428* / 0,329*	0,324* / 0,313*	0,417* / 0,301*	0,292* / 0,284*	0,389* / 0,364*
	Llama3.1	0,207* / 0,084	0,344* / 0,191*	0,293* / 0,271*	0,359* / 0,215*	0,214* / 0,165*	0,234* / 0,157*
	DeepSeek	0,201* / 0,078	0,376* / 0,258*	0,292* / 0,262*	0,376* / 0,241*	0,244* / 0,208*	0,258* / 0,192*
	Gemma3	0,279* / 0,219*	0,438* / 0,353*	0,379* / 0,389*	0,273* / 0,159*	0,307* / 0,289*	0,401* / 0,371*
	Phi-4	0,253* / 0,177*	0,437* / 0,347*	0,360* / 0,383*	0,402* / 0,279*	0,226* / 0,186*	0,352* / 0,346*
S_{norm}	Qwen3	0,240* / 0,273*	0,407* / 0,368*	0,378* / 0,377*	0,321* / 0,299*	0,379* / 0,376*	0,418* / 0,424*
	Llama3.1	0,301* / 0,250*	0,179* / 0,204*	0,314* / 0,341*	0,220* / 0,231*	0,237* / 0,243*	0,231* / 0,226*
	DeepSeek	0,253* / 0,174*	0,365* / 0,288*	0,362* / 0,338*	0,331* / 0,247*	0,304* / 0,274*	0,299* / 0,252*
	Gemma3	0,378* / 0,356*	0,403* / 0,380*	0,403* / 0,431*	-0,036 / 0,169*	0,324* / 0,350*	0,404* / 0,421*
	Phi-4	0,383* / 0,354*	0,379* / 0,369*	0,402* / 0,441*	0,250* / 0,310*	0,186* / 0,263*	0,390* / 0,406*

Оценённые коэффициенты логистической регрессии и соответствующие значения статистической значимости для рассматриваемых вероятностных метрик представлены в табл. 4. В ячейке указаны коэффициенты без контроля длины и с ее контролем.

Значения коэффициентов и p-value указывают на статистически значимую связь рассматриваемых вероятностных метрик с корректностью решений. Для метрик логарифма вероятности положительный коэффициент соответствует росту вероятности корректного решения при увеличении значения метрики. Для энтропии коэффициент отрицателен, что согласуется с тем, что более концентрированное распределение соответствует меньшей энтропии и, в среднем, большей вероятности фактического токена.

Таблица 4 — Коэффициенты логистической регрессии для вероятностных метрик.

Метрика	**Регрессия**	**Qwen3**	**Llama3.1**	**DeepSeek**	**Gemma3**	**Phi-4**
Длина, токены	коэффициент	-0,0075 / —	-0,0075 / —	-0,0085 / —	-0,0064 / —	-0,0075 / —
	p-value	10^{-15} / —	10^{-15} / —	10^{-17} / —	10^{-15} / —	10^{-15} / —
Энтропия	коэффициент	-4,4157 / -5,9820	-1,9386 / -3,3132	-1,9385 / -3,9085	-8,6474 / -9,3311	-2,7257 / -4,7536
	p-value	10^{-16} / 10^{-38}	10^{-9} / 10^{-18}	10^{-14} / 10^{-32}	10^{-37} / 10^{-45}	10^{-13} / 10^{-26}
S_{avg}	коэффициент	0,8706 / 3,5376	0,2962 / 1,7939	1,4380 / 2,0310	0,5459 / 2,4025	0,4769 / 3,0206
	p-value	10^{-4} / 10^{-20}	10^{-4} / 10^{-17}	10^{-20} / 10^{-28}	10^{-4} / 10^{-16}	10^{-3} / 10^{-19}
S_{sum}	коэффициент	0,0412 / 0,0563	0,0188 / 0,0242	0,0141 / 0,0161	0,0296 / 0,0367	0,0365 / 0,0579
	p-value	10^{-66} / 10^{-58}	10^{-13} / 10^{-11}	10^{-12} / 10^{-11}	10^{-49} / 10^{-43}	10^{-30} / 10^{-22}
S_{norm}	коэффициент	0,5113 / 0,5336	0,2591 / 0,2842	0,2042 / 0,1980	0,3669 / 0,4321	0,5402 / 0,6460
	p-value	10^{-61} / 10^{-75}	10^{-16} / 10^{-17}	10^{-14} / 10^{-15}	10^{-48} / 10^{-78}	10^{-26} / 10^{-41}

ROC-кривые для рассматриваемых вариантов логарифмов вероятностей представлены на рис. 2.

Полученные значения ROC-AUC демонстрируют способность вероятностных характеристик языковых моделей различать корректные и некорректные решения. Наиболее высокие значения ROC-AUC наблюдаются для метрик S_{sum} и S_{norm}, тогда как усреднённый логарифм S_{avg} демонстрирует более слабую разделяющую способность. Результаты показывают заметные различия между моделями-оценщиками. Наиболее высокие значения ROC-AUC наблюдаются для Qwen3, Gemma3 и Phi-4 (0,742, 0,734 и 0,725 для S_{norm}), тогда как Llama3.1 и DeepSeek-Coder-v2 заметно уступают (0,634 и 0,672). Ранговая связь между долей корректных решений модели и её разделяющей способностью в роли оценщика составляет 0,70. Это может указывать на то, что модели, демонстрирующие более высокое качество решения задач, также обладают более информативными вероятностными распределениями, что делает их вероятностные оценки более согласованными с функциональной корректностью решений.

C. Самооценка и перекрёстная оценка

Поскольку в эксперименте каждая модель оценивает в том числе собственные решения,

отдельного рассмотрения требует вопрос о том, сохраняется ли работоспособность метрики в этом режиме. Сопоставление без контроля длины и с контролем приведено в табл. 5.

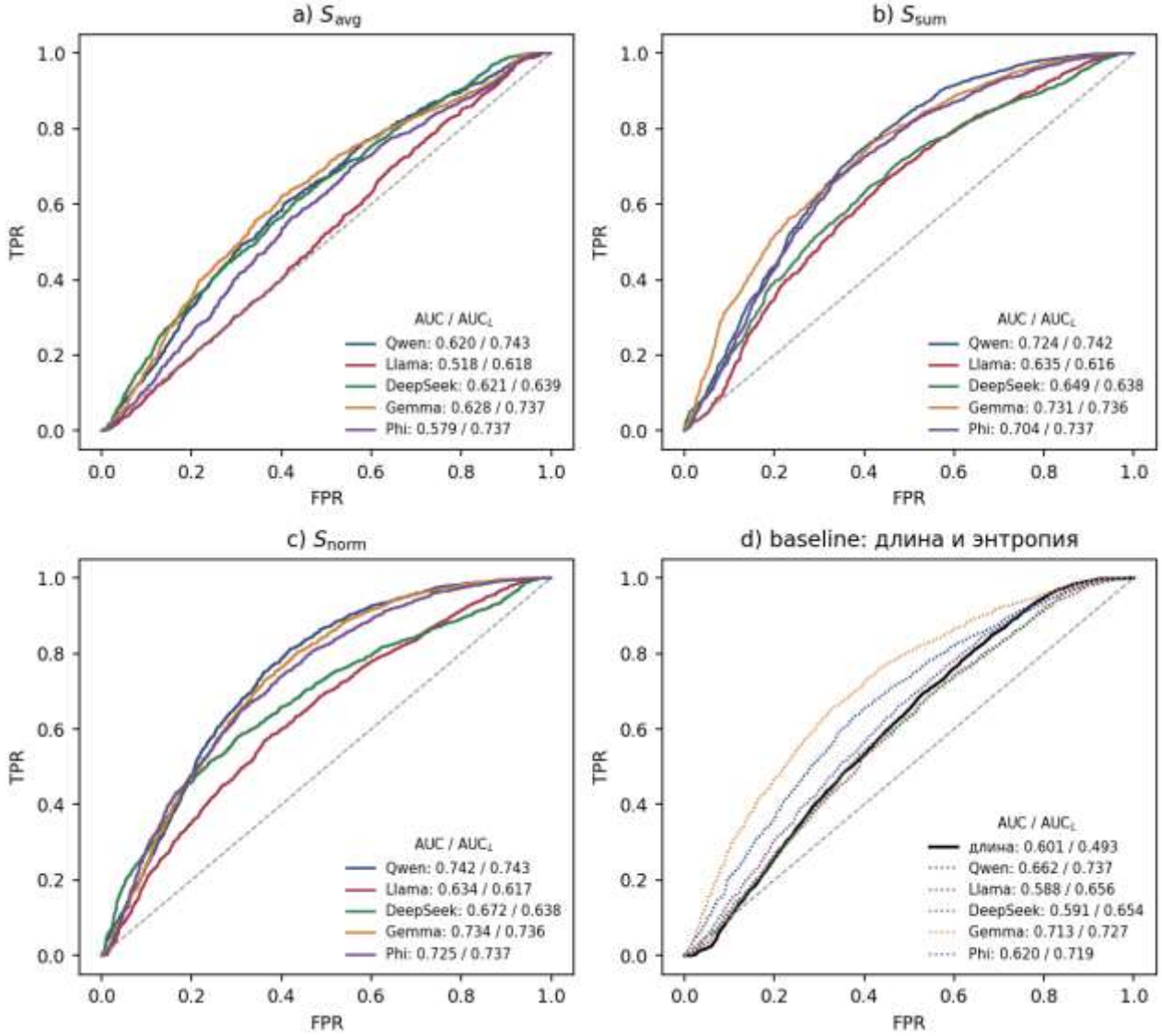

Рис. 2. ROC-кривые для различных вариантов агрегирования логарифмов вероятностей последовательности: a) средний логарифм вероятности, b) суммарный логарифм вероятности, c) нормализованный логарифм вероятности, d) простые альтернативы — длина решения и энтропия распределения токенов.

Таблица 5 — Связь метрики S_{NORM} с корректностью при самооценке и перекрёстной оценке

Режим оценивания	**n**	**ρ**	**частная ρ**	**ROC-AUC**	**AUC$_L$**
Вся выборка	20495	0,326*	0,315*	0,688	0,676
Самооценка	4099	0,255*	0,287*	0,647	0,684
Без самооценки	16396	0,363*	0,344*	0,710	0,692

Без контроля длины самооценка выглядит заметно слабее: ROC-AUC 0,647 против 0,710 при перекрёстной оценке. При сопоставимой длине разрыв практически исчезает — 0,684 против 0,692. Иными словами, в среднем различие между режимами объясняется длиной. При этом расслоение по длине показывает встречные тенденции: на коротких решениях лучше перекрёстная оценка, на длинных — самооценка.

Роль длины подтверждается в разрезе отдельных моделей (табл. 3). Для Gemma3, оценивающей собственные решения, обычный коэффициент корреляции отрицателен и статистически незначим (−0,036), а ROC-AUC составляет 0,478, то есть не отличается от случайного. При контроле длины частный коэффициент становится положительным и значимым (0,169), а AUC_L достигает 0,637. Для метрики S_{avg} знак меняется у трёх моделей из пяти. Таким образом, наблюдаемое в исходной постановке отсутствие связи при самооценивании оказывается больше следствием влияния длины, чем свойством самого режима.

Включение логарифма длины в логистическую регрессию вторым предиктором (табл. 4) приводит к тому же выводу: коэффициенты при вероятностных метриках сохраняют значимость для всех пяти моделей-оценщиков, то есть длина не исчерпывает наблюдаемую связь.

Отдельного внимания заслуживает сама длина

решения как возможный признак корректности: табл. 3 и 4 позволяют сопоставить её с вероятностными метриками напрямую. Связь с корректностью значима, но при сопоставимой длине разделяющая способность падает до 0,49, то есть до уровня случайного выбора. Длину, таким образом, следует рассматривать не как альтернативную метрику, а как величину, влияние которой необходимо устранять.

D. Соотношение логарифмов вероятности и энтропии как метрик

Энтропия (6) вычислялась по десяти наиболее вероятным токенам каждой позиции после нормировки их вероятностей к единице: полное распределение по словарю для этого не требуется. Как предиктор корректности она сопоставима с логарифмами вероятности: при контроле длины её разделяющая способность не падает, по отдельным моделям-оценщикам приближается к нормализованному логарифму (рис. 2, панель d), а в совместной логистической регрессии вместе с S_{norm} и логарифмом длины коэффициент при ней остаётся значимым для всех пяти моделей. Близость результатов обусловлена тем, что обе величины вычисляются из одного и того же распределения. В табл. 6 приведена ранговая корреляция энтропии с тремя вариантами агрегирования логарифма вероятности.

Таблица 6 — Ранговая корреляция предсказательной энтропии с логарифмами вероятности

Оценщик	**S_{avg}**	**S_{sum}**	**S_{norm}**
Qwen3	-0,535	-0,211	-0,491
Llama3.1	-0,702	-0,013	-0,519
DeepSeek	-0,883	-0,159	-0,545
Gemma3	-0,550	-0,468	-0,638
Phi-4	-0,747	-0,034	-0,629

Отрицательный знак закономерен: концентрированное распределение одновременно означает низкую энтропию и повышает вероятность того, что фактический токен окажется среди наиболее вероятных. Существенно, однако, что связь далека от функциональной и сама зависит от длины: от −0,11 на коротких решениях до −0,62 на длинных. Расхождение возникает в позициях, где модель определённа, а решение идёт вразрез с её ожиданием: энтропия там низка, а логарифм фактического токена мал. Именно такие позиции и представляют интерес при оценке корректности. Отдельно обращает на себя внимание столбец S_{sum}: с суммарным логарифмом энтропия почти не связана. Однако при фиксации длины коэффициент достигает −0,22…−0,57 во всех слоях, кроме самого короткого, то есть S_{sum} ведёт себя как остальные варианты агрегирования. Две характеристики, таким образом, описывают разные стороны одного распределения: логарифм — насколько модель уверена в тех токенах, которые предъявлены ей на оценку в составе уже сформированной последовательности, энтропия — насколько модель вообще определена относительно возможных продолжений в данном контексте.

V. Заключение

В работе исследована связь между логарифмами вероятностей, вычисляемых языковыми моделями для сгенерированных решений, и корректностью этих решений. В ходе экспериментов для набора задач на нескольких языках программирования были вычислены различные варианты агрегирования логарифмов вероятности последовательностей токенов. Полученные результаты показывают статистически значимую зависимость между вероятностными метриками и функциональной корректностью решений. Наиболее выраженная связь наблюдается для логарифма, учитывающего длину последовательности, а при контроле длины все варианты дают сопоставимый AUC. Модели с более высокой долей корректных решений дают вероятностные оценки, лучше согласованные с корректностью. Вероятностные характеристики языковых моделей, таким образом, могут служить сигналом корректности сгенерированных решений и, не требуя отдельных инструкций, дополнять подход LLM-as-Judge.

Контроль длины решения показывает, что сырые значения вероятностных метрик частично отражают длину ответа, и это необходимо учитывать при сопоставлении. После выравнивания длины разделяющая способность метрик сохраняется, а самооценка становится сопоставимой с перекрёстной оценкой, хотя её качество зависит от длины решения. Длина решения не является устойчивым признаком корректности, тогда как энтропия распределения сопоставима с логарифмами по разделяющей способности. Их совместное использование тем выгоднее, чем слабее связь между ними, и требует отдельного исследования.

Следует указать ограничения проведённого анализа. Энтропия вычислялась по усечённому распределению из десяти наиболее вероятных токенов, что, как правило, занижает значения, особенно в позициях с размытым распределением. Наблюдения не независимы: одни и те же условия задач повторяются на разных языках, генераторах и оценщиках, поэтому значимость оценена с кластеризацией по условию задачи. Парный контроль длины отвечает на вопрос о различении решений равной длины, но не даёт разложения вкладов. Кроме того, функциональная корректность в работе определяется результатами тестирования, поэтому полученные зависимости характеризуют только связь вероятностных метрик с прохождением тестов. Наконец, эксперимент проведён на ограниченном наборе задач HumanEval-X и пяти моделях, поэтому переносимость выявленных закономерностей требует отдельной проверки.

Перспективными направлениями представляются исследование подобных метрик на более широком классе задач программирования и моделях большего масштаба, изучение смещения оценок в зависимости от формулировки задачи и дополнительных инструкций по оценке, а также анализ влияния механизмов токенизации на связь с функциональной корректностью.

СПИСОК ЛИТЕРАТУРЫ

[1] Kaukaz S., Sarhan Q. A Systematic Survey on Large Language Models for Code Generation // ARO-The Scientific Journal of Koya University. 2025. Vol. 13. P. 83-99. DOI: 10.14500/aro.12159.

[2] Gulwani S., Polozov O., Singh R. Program Synthesis // Found. Trends Program. Lang. 2017. Vol. 4, No. 1–2. P. 1-119. DOI: 10.1561/2500000010.

[3] Papineni K., Roukos S., Ward T., Zhu W.-J. Bleu: a Method for Automatic Evaluation of Machine Translation // Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. 2002. P. 311-318. DOI: 10.3115/1073083.1073135.

[4] Lin C.-Y. ROUGE: A Package for Automatic Evaluation of Summaries // Text Summarization Branches Out. 2004. P. 74-81.

[5] Liu Y., Iter D., Xu Y., Wang S., Xu R., Zhu C. G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment. 2023. arXiv:2303.16634.

[6] Gu J., Jiang X., Shi Z., Tan H., Zhai X., Xu C., Li W., Shen Y., Ma S., Liu H., Wang S., Zhang K., Lin Z., Zhang B., Ni L., Gao W., Wang Y., Guo J. A survey on LLM-as-a-Judge // The Innovation. 2026. P. 101253. DOI: 10.1016/j.xinn.2025.101253.

[7] Pan R., Zhang H., Liu C. CodeCoR: An LLM-Based Self-Reflective Multi-Agent Framework for Code Generation. 2025. arXiv:2501.07811.

[8] Ren S., Guo D., Lu S., Zhou L., Liu S., Tang D., Sundaresan N., Zhou M., Blanco A., Ma S. CodeBLEU: a Method for Automatic Evaluation of Code Synthesis. 2020. arXiv:2009.10297.

[9] Feng Z., Guo D., Tang D., Duan N., Feng X., Gong M., Shou L., Qin B., Liu T., Jiang D., Zhou M. CodeBERT: A Pre-Trained Model for Programming and Natural Languages. 2020. arXiv:2002.08155.

[10] Fedrushkov D., Kovalchuk S., Aliev A. Natural language processing metrics efficiency for evaluating a generated code: facing the challenge // Scientific and Technical Journal of Information Technologies, Mechanics and Optics. 2026. Vol. 26. P. 135-144. DOI: 10.17586/2226-1494-2026-26-1-135-144.

[11] Zhou S., Alon U., Agarwal S., Neubig G. CodeBERTScore: Evaluating Code Generation with Pretrained Models of Code // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023. P. 13921-13937. DOI: 10.18653/v1/2023.emnlp-main.859.

[12] Kovalchuk S., Fedrushkov D., Lomshakov V., Aliev A. Test-based and metric-based evaluation of code generation models for practical question answering // 2023 International Conference on Code Quality (ICCQ). 2023. P. 73-86. DOI: 10.1109/ICCQ57276.2023.10114665.

[13] Shemetova E., Smirnov I., Alekseev A., Shenbin I., Rukhovich A., Nikolenko S., Lomshakov V., Piontkovskaya I. LAMeD: LLM-generated Annotations for Memory Leak Detection // Proceedings of the 29th International Conference on Evaluation and Assessment in Software Engineering. 2025. P. 1024-1034. DOI: 10.1145/3756681.3756999.

[14] He J., Shi J., Zhuo T. Y., Treude C., Sun J., Xing Z., Du X., Lo D. LLM-as-a-Judge for Software Engineering: Literature Review, Vision, and the Road Ahead // ACM Transactions on Software Engineering and Methodology. 2026. DOI: 10.1145/3797276.

[15] Fedrushkov D., He Y., Smirnov I., Aliev A., Kovalchuk S. Semantic vs. Structural Signals: Log-Probability and LLM-as-a-Judge for Reference-Free Code Evaluation // Proceedings of the Fifth Workshop on Generation, Evaluation and Metrics (GEM). 2026. P. 574-581. DOI: 10.18653/v1/2026.gem-main.55.

[16] Ott M., Auli M., Grangier D., Ranzato M. Analyzing Uncertainty in Neural Machine Translation // Proceedings of the 35th International Conference on Machine Learning. 2018. Vol. 80. P. 3956-3965.

[17] Chen M. et al. Evaluating Large Language Models Trained on Code. 2021. arXiv:2107.03374.

[18] Zheng Q., Xia X., Zou X., Dong Y., Wang S., Xue Y., Shen L., Wang Z., Wang A., Li Y., Su T., Yang Z., Tang J. CodeGeeX: A Pre-Trained Model for Code Generation with Multilingual Benchmarking on HumanEval-X // Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2023. P. 5673-5684. DOI: 10.1145/3580305.3599790.

[19] Yang A. et al. Qwen3 Technical Report. 2025. arXiv:2505.09388.

[20] Grattafiori A. et al. The Llama 3 Herd of Models. 2024. arXiv:2407.21783.

[21] Zhu Q. et al. DeepSeek-Coder-V2: Breaking the Barrier of Closed-Source Models in Code Intelligence. 2024. arXiv:2406.11931.

[22] Kamath A. et al. Gemma 3 Technical Report. 2025. arXiv:2503.19786.

[23] Abdin M. et al. Phi-4 Technical Report. 2024. arXiv:2412.08905.

Д. В. Федрушков, аспирант, Университет ИТМО (https://itmo.ru), Санкт-Петербург, email: dvfedrush@gmail.com, ORCID: 0009-0003-9088-4056, Scopus author ID: 57204596230

И. З. Смирнов, аспирант, Университет ИТМО (https://itmo.ru), Санкт-Петербург, email: izsmirnov@itmo.ru, ORCID: 0009-0002-3690-4733, Scopus author ID: 58059937000

С. В. Ковальчук, к.т.н., старший научный сотрудник, Университет ИТМО (https://itmo.ru), Санкт-Петербург, email: kovalchuk@itmo.ru, ORCID: 0000-0001-8828-4615, Scopus author ID: 55382199400.

А. А. Алиев, старший преподаватель, Санкт-Петербургский государственный университет (https://spbu.ru), Санкт-Петербург, email: artem.aliev@gmail.com, ORCID: 0000-0001-7984-4721, Scopus author ID: 58188719500.

Log Probability-Based Evaluation of Code Correctness

D. V. Fedrushkov, I. Z. Smirnov, S. V. Kovalchuk, A. A. Aliev

Abstract — This work investigates probabilistic characteristics of language models as a tool for assessing the correctness of program code. The proposed approach is based on the analysis of token-level log-probabilities assigned by the model to generated solutions and requires neither reference solutions nor code execution. We examine the relationship between log-probability values and functional correctness measured by automated testing, using rank correlation, ROC analysis and logistic regression, and we compare several strategies for aggregating token-level values into a sequence-level score. Because solution length is associated both with the metrics and with the probability of passing tests, every comparison is reported twice: without length control and under length control, implemented as paired comparison within a narrow length band and as partial rank correlation. Experiments on the HumanEval-X benchmark across five programming languages and five open-weight models of up to 16 billion parameters show that probabilistic metrics distinguish correct from incorrect solutions and exhibit a statistically significant association with pass@1. Under length control the gap between self- and cross-evaluation substantially decreases, solution length alone is a weak predictor of correctness, whereas predictive entropy of the token distribution remains competitive with log-probability-based metrics.

Keywords — artificial intelligence, language models, code generation, metrics, probability distribution, quality evaluation, code correctness

REFERENCES

[1] Kaukaz S., Sarhan Q. A Systematic Survey on Large Language Models for Code Generation // ARO-The Scientific Journal of Koya University. 2025. Vol. 13. P. 83-99. DOI: 10.14500/aro.12159.

[2] Gulwani S., Polozov O., Singh R. Program Synthesis // Found. Trends Program. Lang. 2017. Vol. 4, No. 1–2. P. 1-119. DOI: 10.1561/2500000010.

[3] Papineni K., Roukos S., Ward T., Zhu W.-J. Bleu: a Method for Automatic Evaluation of Machine Translation // Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. 2002. P. 311-318. DOI: 10.3115/1073083.1073135.

[4] Lin C.-Y. ROUGE: A Package for Automatic Evaluation of Summaries // Text Summarization Branches Out. 2004. P. 74-81.

[5] Liu Y., Iter D., Xu Y., Wang S., Xu R., Zhu C. G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment. 2023. arXiv:2303.16634.

[6] Gu J., Jiang X., Shi Z., Tan H., Zhai X., Xu C., Li W., Shen Y., Ma S., Liu H., Wang S., Zhang K., Lin Z., Zhang B., Ni L., Gao W., Wang Y., Guo J. A survey on LLM-as-a-Judge // The Innovation. 2026. P. 101253. DOI: 10.1016/j.xinn.2025.101253.

[7] Pan R., Zhang H., Liu C. CodeCoR: An LLM-Based Self-Reflective Multi-Agent Framework for Code Generation. 2025. arXiv:2501.07811.

[8] Ren S., Guo D., Lu S., Zhou L., Liu S., Tang D., Sundaresan N., Zhou M., Blanco A., Ma S. CodeBLEU: a Method for Automatic Evaluation of Code Synthesis. 2020. arXiv:2009.10297.

[9] Feng Z., Guo D., Tang D., Duan N., Feng X., Gong M., Shou L., Qin B., Liu T., Jiang D., Zhou M. CodeBERT: A Pre-Trained Model for Programming and Natural Languages. 2020. arXiv:2002.08155.

[10] Fedrushkov D., Kovalchuk S., Aliev A. Natural language processing metrics efficiency for evaluating a generated code: facing the challenge // Scientific and Technical Journal of Information Technologies, Mechanics and Optics. 2026. Vol. 26. P. 135-144. DOI: 10.17586/2226-1494-2026-26-1-135-144.

[11] Zhou S., Alon U., Agarwal S., Neubig G. CodeBERTScore: Evaluating Code Generation with Pretrained Models of Code // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023. P. 13921-13937. DOI: 10.18653/v1/2023.emnlp-main.859.

[12] Kovalchuk S., Fedrushkov D., Lomshakov V., Aliev A. Test-based and metric-based evaluation of code generation models for practical question answering // 2023 International Conference on Code Quality (ICCQ). 2023. P. 73-86. DOI: 10.1109/ICCQ57276.2023.10114665.

[13] Shemetova E., Smirnov I., Alekseev A., Shenbin I., Rukhovich A., Nikolenko S., Lomshakov V., Piontkovskaya I. LAMeD: LLM-generated Annotations for Memory Leak Detection // Proceedings of the 29th International Conference on Evaluation and Assessment in Software Engineering. 2025. P. 1024-1034. DOI: 10.1145/3756681.3756999.

[14] He J., Shi J., Zhuo T. Y., Treude C., Sun J., Xing Z., Du X., Lo D. LLM-as-a-Judge for Software Engineering: Literature Review, Vision, and the Road Ahead // ACM Transactions on Software Engineering and Methodology. 2026. DOI: 10.1145/3797276.

[15] Fedrushkov D., He Y., Smirnov I., Aliev A., Kovalchuk S. Semantic vs. Structural Signals: Log-Probability and LLM-as-a-Judge for Reference-Free Code Evaluation // Proceedings of the Fifth Workshop on Generation, Evaluation and Metrics (GEM). 2026. P. 574-581. DOI: 10.18653/v1/2026.gem-main.55.

[16] Ott M., Auli M., Grangier D., Ranzato M. Analyzing Uncertainty in Neural Machine Translation // Proceedings of the 35th International Conference on Machine Learning. 2018. Vol. 80. P. 3956-3965.

[17] Chen M. et al. Evaluating Large Language Models Trained on Code. 2021. arXiv:2107.03374.

[18] Zheng Q., Xia X., Zou X., Dong Y., Wang S., Xue Y., Shen L., Wang Z., Wang A., Li Y., Su T., Yang Z., Tang J. CodeGeeX: A Pre-Trained Model for Code Generation with Multilingual Benchmarking on HumanEval-X // Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2023. P. 5673-5684. DOI: 10.1145/3580305.3599790.

[19] Yang A. et al. Qwen3 Technical Report. 2025. arXiv:2505.09388.

[20] Grattafiori A. et al. The Llama 3 Herd of Models. 2024. arXiv:2407.21783.

[21] Zhu Q. et al. DeepSeek-Coder-V2: Breaking the Barrier of Closed-Source Models in Code Intelligence. 2024. arXiv:2406.11931.

[22] Kamath A. et al. Gemma 3 Technical Report. 2025. arXiv:2503.19786.

[23] Abdin M. et al. Phi-4 Technical Report. 2024. arXiv:2412.08905.

D. V. Fedrushkov, PhD Student, ITMO University (https://itmo.ru), Saint Petersburg, email: dvfedrush@gmail.com, ORCID: 0009-0003-9088-4056, Scopus Author ID: 57204596230

I. Z. Smirnov, PhD Student, ITMO University (https://itmo.ru), Saint Petersburg, email: izsmirnov@itmo.ru, ORCID: 0009-0002-3690-4733, Scopus Author ID: 58059937000

S. V. Kovalchuk, PhD in Engineering, Senior Researcher, ITMO University (https://itmo.ru), Saint Petersburg, email: kovalchuk@itmo.ru, ORCID: 0000-0001-8828-4615, Scopus Author ID: 55382199400

A. A. Aliev, Senior Teacher, Saint Petersburg State University (https://spbu.ru), Saint Petersburg, email: artem.aliev@gmail.com, ORCID: 0000-0001-7984-4721, Scopus Author ID: 58188719500