ОБ АТАКАХ НА БОЛЬШИЕ ФУНДАМЕНТАЛЬНЫЕ МОДЕЛИ

Грибунин В. Г.[1], Майоров С. А.[2], Мурашко А. А.[3]

DOI: 10.21681/2311-3456-2025-4-30-34

Цель исследования: *изучить возможности и ограничения нарушителя безопасности информации по организации атаки на большие фундаментальные модели, предназначенные для работы с программным кодом.*

Методы исследования: *сравнение и сопоставление, системный анализ.*

Результаты исследования: *в статье представлены особенности больших фундаментальных моделей, как объектов защиты информации, принципы реализации наиболее релевантных атак на большие фундаментальные модели, предназначенных для работы с программным кодом, приведены метрики, позволяющие сравнивать эффективность различных подходов к реализации атак, указаны проблемы, существующие в данной области для нарушителя безопасности информации.*

Научная новизна: *большие фундаментальные модели в мире только начинают использовать для работы с программным кодом. В статье описаны угрозы безопасности информации и возможные атаки на системы, использующие данные модели. Также представлены проблемные вопросы, требующие дальнейших исследований.*

Ключевые слова: *глубокое обучение, нарушители, угрозы, бэкдор, отравление данных, отравление модели, состязательные атаки.*

Введение

Достижения последних лет в области искусственного интеллекта во многом связаны с созданием больших фундаментальных моделей (БФМ). Это понятие было введено в новой редакции Национальной стратегии развития искусственного интеллекта на период до 2030 года[4] [1]. Под БФМ понимаются «модели искусственного интеллекта, являющиеся основой для создания и доработки различных видов программного обеспечения, обученные распознаванию определенных видов закономерностей, содержащие не менее 1 млрд параметров и применяемые для выполнения большого количества различных задач».

Как отмечено в Национальной стратегии, «большие фундаментальные модели уже сейчас способны писать программные коды по техническим заданиям...». Конечно, в настоящее время генерация программного кода весьма ограничена размером контекстного окна БФМ (в настоящее время около 130 000 токенов) [1] и, следовательно, объемом генерируемого кода (по нашему опыту, надежно генерируемый объем кода составляет в настоящее время 200–400 строк, в зависимости от языка программирования). Кроме того, БФМ предъявляют существенные требования к используемому оборудованию для их локального запуска. Например, Deepseek v3 требует для своего инференса наличия 80 Гбайт памяти на видеокарте типа NVIDIA A100 или H100. Однако, бурный прогресс в этом направлении позволяет надеяться на то, что уже в ближайшее время БФМ займут достойное место в арсенале всех разработчиков.

Помимо генерации кода, БФМ могут быть использованы для решения таких задач, как поиск кода, завершение/генерация кода и реферирование кода. Вместе с тем эти модели подвержены угрозам безопасности информации, так как в их основе лежат глубокие нейронные сети. Уязвимости и угрозы для глубоких нейронных сетей приведены, например, в [2]. Присущие БФМ уязвимости могут стать преградой для их использования в критически важных приложениях, таких как сетевая безопасность или обнаружение вредоносного программного обеспечения. Как будет показано далее, также, как и другие технологии искусственного интеллекта, БФМ подвержены атакам отравления данных и модели, то есть внедрения скрытых бэкдоров (триггеров) во время обучения [2], а также состязательным атакам на этапе инференса. В результате проведенных нарушителем атак БФМ могут генерировать вредоносный код [3].

Специфика БФМ связана со сложностью контроля за входными данными (человеческий язык) и ее

1 Грибунин Вадим Геннадьевич, доктор технических наук, доцент, главный научный сотрудник АНО «Институт инженерной физики», г. Серпухов Московской области, Россия. E-mail: wavelet2@mail.ru

2 Майоров Сергей Алексеевич, кандидат технических наук, старший научный сотрудник научно-методического управления АНО «Институт инженерной физики», г. Серпухов Московской области, Россия. E-mail: oniokr@iifmail.ru

3 Мурашко Александр Анатольевич, доктор технических наук, доцент, старший научный сотрудник научно-методического управления АНО «Институт инженерной физики», г. Серпухов Московской области, Россия. E-mail: oniokr@iifmail.ru

4 Национальная стратегия развития искусственного интеллекта на период до 2030 года (с дополнениями Указа Президента РФ от 15.02.2024 г. № 124. Режим доступа: http://www.kremlin.ru/acts/bank/44731 – Время доступа: 31.03.2025.

функционированием в режиме псевдореального времени (например, в чат-ботах). Обычно же при использовании моделей глубокого обучения форматы ввода и сценарии использования бывают, как правило, более контролируемы.

С другой стороны, код имеет четко выраженную структуру и синтаксис, должен не только соответствовать строгим грамматическим правилам, но и обеспечивать логическую точность и исполняемость. Даже небольшие ошибки или изменения во время генерации и понимания кода могут привести к сбою программы или получению неожиданных результатов. Это усложняет задачу нарушителя безопасности, так как при выполнении атаки ему необходимо сохранить корректность кода.

Описание атак на большие фундаментальные модели

Также, как и по отношению к другим моделям машинного обучения, на БФМ возможно осуществление двух классов атак: атаки бэкдора (триггера), заключающиеся в отравлении данных или отравлении модели, и состязательные атаки, заключающиеся в добавлении к входным данным модели небольших, специальным образом рассчитанных возмущений. Эти возмущения могут быть рассчитаны на основе знания внутренней структуры модели («белый ящик» и путем анализа ответов на запросы к модели (атаки «черного ящика»).

При отравлении данных в них внедряются бэкдоры, которые во время инференса модели, обученной на этом датасете, выступают в качестве триггера, приводя к неверной классификации входных образцов либо к генерации вредоносной информации [4].

Пусть исходный обучающий датасет $D = \{\mathcal{X},\mathcal{Y}\}$, где $x = \{x_i\}_{i=1}^{n} \in \mathcal{X}$ – последовательность из n токенов, $y \in \mathcal{Y}$ — соответствующие для них метки (для задач классификации кода).

Нарушитель создает скрытые бэкдоры с m токенами $\{t_i^*\}_{i=1}^{m}$, внедряет их в некоторое множество образцов датасета, в результате чего получается новый датасет $\mathcal{D}_p = \{\mathcal{D} \cup \mathcal{D}^*\}$, где $\mathcal{D}^* = \{\mathcal{X}^*,y^*\}$ – образцы данных с внедренными бэкдорами.

Для задач генерации кода соответствующая «истина» может быть представлена как $\{y_1,y_2,\ldots,y_n\} \in \mathcal{Y}$, а целевая метка как

$$y^* = \{y_1,y_2,\ldots,y_t,y_n\},\ x^* = \{x_i\}_{i=1}^{n} \oplus \{t_i\}_{i=1}^{m} \in \mathcal{X}^*.$$

Далее атакующий должен создать условия для того, чтобы жертва-разработчик скачал отравленные образцы $\mathcal{D}_p$ исходного кода (например, с GitHub).

При отравлении модели вначале вышеописанным способом выполняется процесс отравления данных. Далее модель обучается на этом отравленном датасете $f(\theta^*)$ таким образом, чтобы она оказалась связанной с некоторой секретной последовательностью m токенов бэкдора t^*, появление которой вызовет классификацию к целевой метке y^* (в случае выполнения задачи классификации). Отравление модели минимизирует потери обучения [5]:

$$\mathcal{L}_{D_p}(\theta^*) = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}'} \mathcal{L}(f(x;\theta^*),y) + \mathop{\mathbb{E}}_{(x^*,y^*)\sim\mathcal{D}^*} \mathcal{L}(f(x^*;\theta^*),y^*), \quad (1)$$

где $\mathcal{L}(.,.)$ – функция потерь (перекрестная энтропия).

Суть состязательных атак заключается в генерации определенной шумовой последовательности и добавления ее к исходным входным примерам с целью обмануть модель во время инференса [6].

Пусть x и y представляют собой входные данные и прогнозируемые выходные данные БФМ, соответственно. Состязательная выборка x' создается путем применения небольших специально рассчитанных возмущений к входным данным x. В результате проведения состязательной атаки БФМ демонстрирует высокую уверенность в своем неправильном предсказании для x'. Это может быть описано следующими выражениями [6]:

$$\begin{aligned} x' &= x + \eta \\ f(x) &= y \\ f(x') &\neq y \\ f(x') &= y^{\wedge\prime},\ y^{\wedge\prime} \neq y, \end{aligned} \quad (2)$$

где η есть специальным образом рассчитанное возмущение. Для входных данных x, представляющих собой фрагмент кода, в качестве η могут быть возмущения на уровне токена, оператора или блока. Целью состязательной атаки может быть либо отклонение прогноза модели от правильной метки $f(x') \neq y$, либо склонение решения модели к определенной метке $f(x') = y'$ (целевая атака).

Против различных БФМ показывают эффективность как атаки «белого», так и «черного» ящика. Методы атак «белого ящика» основываются на доступе к градиентам целевой модели для создания состязательных примеров. Эти примеры формируются путем внесения изменений, не затрагивающих семантику программы и позволяющих ее успешно скомпилировать. Наиболее распространенными операциями для таких атак являются подстановки, вставки и удаления элементов на уровне идентификаторов и операторов. Примерами являются переименование переменных, параметров функций, классов или структур, замена булевых выражений эквивалентными вариантами, а также вставка и удаление бесполезного кода или пустых операторов. Важно, чтобы такие примеры оставались семантически эквивалентными оригиналу, сохраняя при этом атаку незаметной для человека.

Современные исследования состязательных атак преимущественно сосредоточены на методах

«черного ящика». В отличие от атак «белого ящика», этот подход не требует знания градиентов модели. Вместо этого создаются состязательные примеры на основании анализа связи между входными и выходными данными модели. В методах «черного ящика» используются эволюционные или генетические алгоритмы, методы «запрос-ответ» для поиска оптимальных изменений в примерах. По вполне понятным причинам атаки «черного ящика» требуют большего времени и, как правило, менее успешны, чем атаки «белого ящика».

Метрики, используемые в исследованиях атак и защиты на большие фундаментальные модели

Для оценки эффективности методов атаки и защиты БФМ в различных работах предложено множество метрик оценки [5,7]. Причем эти метрики отличаются от метрик, используемых в исследованиях методов атак и защиты технологий искусственного интеллекта, работающих с другими модальностями данных. Рассмотрим метрики для БФМ.

1. Ложная тревога, например, о наличии бэкдоров (FPR). FPR есть отношение положительных образцов с неверными прогнозами ко всем положительным образцам:

$$FPR = \frac{FP}{TN + FP}. \quad (3)$$

2. Средний нормализованный ранг (ANR) используется для оценки эффективности атак бэкдора против моделей, предназначенных для извлечения нужного кода (из больших репозиториев). ANR показывает, насколько атака может повысить ранг извлечения для отравленных образцов. Меньшее значение ANR означает более эффективный метод атаки. В приведенном ниже выражении s' обозначает фрагмент кода бэкдора, $|S|$ – длину полного ранжированного списка, $|Q|$ – объем множества запросов:

$$ANR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{\mathrm{Rank}(Q_i,s')}{|S|}. \quad (4)$$

Среднее количество успешных запросов (промтов) к целевой модели в состязательной атаке. Важность этого показателя состоит в том, что для методов состязательной атаки «черного ящика» промты являются единственным способом доступа к целевой модели. В нижеприведенном выражении q_i – это количество запросов для i-й успешной атаки, $i \in \{j | f(j) = 1\}$:

$$\text{Запрос} = \frac{\sum q_i}{\sum f(i)}$$

$$f(j) = \begin{cases} 1, & \text{если } M(C_j^{adv}) \neq y_j \wedge M(C_j) = y_j, \\ 0, & \text{иначе.} \end{cases} \quad (5)$$

4) Коэффициент возмущения (Pert) показывает долю возмущения (шума), которому подвергнут исходный код во время состязательной атаки. Более низкий коэффициент возмущения указывает на то, что сгенерированные состязательные образцы имеют меньше возмущений. C_i^{adv} есть состязательный пример для C_i. $t(\cdot)$ обозначает количество токенов:

$$\mathrm{Pert} = \frac{\sum t(C_i^{adv}) - t(C_i^{adv} \cap C_i)}{\sum t(C_i)}. \quad (6)$$

5) Относительная деградация (R_d) используется для оценки ухудшения производительности модели при воздействии атаки. «refs» обозначает комментарий к ссылке, y обозначает исходный вывод, y' обозначает вывод возмущенной программы:

$$R_d = \frac{\mathrm{BLEU}(y,\mathrm{refs}) - \mathrm{BLEU}(y',\mathrm{refs})}{\mathrm{BLEU}(y,\mathrm{refs})}, \quad (7)$$

где BLEU (bilingual evaluation understudy) – известная метрика оценки расстояния между текстами.

6) Коэффициент валидности (V_r) определяется как отношение числа состязательных образцов, которые могут быть скомпилированы ($\mathrm{Count}_{\mathrm{valid}}$), к общему числу состязательных образцов ($\mathrm{Count}_{\mathrm{all}}$):

$$V_r = \frac{\mathrm{Count}_{\mathrm{valid}}}{\mathrm{Count}_{\mathrm{all}}}. \quad (8)$$

7) Коэффициент успеха (S_r) – комплексный показатель эффективности атаки и качества сгенерированных образцов. Он определяется как произведение относительной деградации (R_d) и коэффициента валидности (V_r):

$$S_r = R_d * V_r. \quad (9)$$

8) Доля измененных переменных (VCR) показывает, сколько злоумышленнику переменных необходимо изменить для успешной атаки. В нижеприведенном выражении для вычисления VCR n_i – это число переменных, измененных злоумышленником в i-й состязательной выборке, m_i – общее число переменных в i-й выборке:

$$VCR = \frac{\sum i\ n_i}{\sum i\ m_i}. \quad (10)$$

Проблемы, связанные с атаками на большие фундаментальные модели

Анализ известных работ по проведению атак на БФМ показал, что для нарушителя существуют следующие проблемы.

1) Скрытность триггеров бэкдора. В атаках бэкдора на БФМ скрытность внедряемых триггеров является основой успеха. Поэтому наблюдается эволюция методов создания триггеров, начиная от самых первых, таких как мертвый код, до методов модификации имен переменных/функций и даже внедрения адаптивных триггеров [8]. Однако всесторонняя оценка скрытности триггеров остается сложной задачей. В известных методах оцениваются какие-либо

частные показатели, например, синтаксическая или семантическая видимость, или заметность для человека.

2) Методы внедрения бэкдора для БФМ. Известные методы внедрения бэкдора плохо работают с программным кодом. В большинстве исследований рассматриваются, как правило, два сценария, в зависимости от того, могут или нет нарушители контролировать процесс обучения модели. В любом случае считается, что отравить обучающий датасет они могут. Однако коммерческие БФМ, такие как, например, Codex и GPT-4, имеют закрытый исходный код. Следовательно, злоумышленники не могут контролировать процесс обучения или отслеживать данные обучения. Для БФМ с открытым исходным кодом стоимость внедрения бэкдоров во время обучения или тонкой настройки очень велика. Кроме того, поскольку БФМ становятся все более сложными и надежными, злоумышленникам становится все труднее скрытно внедрять бэкдоры, и эффективность этих бэкдоров имеет тенденцию к снижению по мере роста размера модели.

3) Синтаксическая правильность и семантическое сохранение состязательных образцов. В состязательных атаках на БФМ сгенерированные состязательные образцы должны обладать синтаксической правильностью и сохранять семантику кода. Однако, даже если состязательные образцы сохраняют семантику кода, они могут вносить синтаксические или логические ошибки во время выполнения. Поэтому достижение синтаксической правильности и семантической согласованности состязательных образцов является сложной задачей.

4) Скрытность состязательных возмущений. Когда мы говорим о скрытности, то возникает вопрос: «Скрытность для кого?» Одно дело – скрытность для различных анализаторов программного кода, другое – для человеческого глаза опытного аналитика. Например, при оценке качества атак иногда используются метрики, основанные на сходстве текстов (например, вышеупомянутая CodeBLEU), однако эти метрики не всегда коррелируют с особенностями человеческого восприятия. Разработка метрик для оценки скрытности, учитывающих человеческое восприятие, остается сложной и пока не решенной до конца задачей.

5) Глубокое понимание принципов, лежащих в основе атак на КБФМ. Развитие объяснимости может помочь лучше понять основные принципы бэкдор-атак и состязательных атак. Для глубоких нейронных сетей известна проблема низкой интерпретируемости. Небольшие изменения параметров модели могут существенно повлиять на результаты прогнозирования, и редко возможно понять причину. Поэтому в последние годы в мире значительное внимание уделяется объяснимости глубокого обучения. Объяснимость может не только повысить безопасность БФМ, но и раскрыть ее «тайны», упростив понимание ее механизмов для злоумышленника.

Выводы

Таким образом, как показано в статье, применение БФМ для решения задач, связанных с обработкой и генерацией программного кода, сопряжено с определенными проблемами безопасности, хотя задача нарушителя по организации атак существенно усложнена по сравнению с другими модальностями систем глубокого машинного обучения.

Для противодействия атакам на БФМ были разработаны различные методы защиты, по аналогии с тем, как это делается для иных технологий искусственного интеллекта. Защита заключается в обнаружении аномалий в обучающих данных, в проведении состязательного обучения для повышения устойчивости модели к атакам.

Литература

1. Azim M. Best Open Source LLMs for Code Generation in 2025. – Интернет-ресурс. – Режим доступа: https://www.cubix.co/blog/best-open-source-llms-for-code-generation-in-2025. – Время доступа: 31.03.2025.
2. Грибунин В. Г., Кондаков С. Е. К вопросу о защите информации в интеллектуализированных образцах вооружения // Вопросы кибербезопасности. – 2021. – № 5(45). – Стр. 5–11. DOI:10.21681/2311-3456-2021-5-5-11.
3. Schuster R., Song C., Tromer E., Shmatikov V. You Autocomplete Me: Poisoning Vulnerabilities in Neural Code Completion // Proceedings of the 30th USENIX Security Symposium. USENIX Association, Canada. – 2021, pp. 1559–1575.
4. Gu T., Dolan-Gavitt B., Garg S. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain // Режим доступа: arXiv abs/1708.06733. DOI:10.48550/arXiv.1708.06733.
5. Chen Y. и др. Security of Language Models for Code: A Systematic Literature Review // Режим доступа: https://arxiv.org/pdf/2410.15631. – Время доступа: 31.03.2025.
6. Alzantot M., Sharma Y., Elgohary A. и др. Generating Natural Language Adversarial Examples // Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Brussels, Belgium. – Pp. 2890–2896. DOI:10.48550/arXiv.1804.07998.
7. Wan Y., Zhang S., Zhang H. и др. You see what I want you to see: poisoning vulnerabilities in neural code search // Proceedings of the 30th ACM foint European Software Engineering Conference and Symposium on the Foundations of Software Engineering. ACM, Singapore. – Pp.1233–1245. DOI:10.1145/3540250.3549153.
8. Ramakrishnan G., Albarghouthi A. Backdoors in Neural Models of Source Code // Proceedings of the 26th International Conference on Pattern Recognition. IEEE, Canada. – Pp.2892–2899. DOI:10.1109/ICPR56361.2022.9956690.

ABOUT ATTACKS ON LARGE FUNDAMENTAL MODELS

Gribunin V. G.[5], Mayorov S. A.[6], Murashko A. A.[7]

Keywords: *deep learning, intruders, threats, backdoor, data poisoning, model poisoning, adversarial attacks.*

Purpose of the study: *to study the possibilities and limitations of an information security attacker in organizing an attack on large fundamental models designed to work with program code.*

Methods of research: *comparison and juxtaposition, system analysis.*

Results: *the article presents the features of large fundamental models as objects of information protection, the principles of implementing the most relevant attacks on large fundamental models designed to work with program code, provides metrics that allow comparing the effectiveness of various approaches to attacks, and identifies the problems that exist in this area for attackers*

Scientific novelty: *large fundamental models in the world are just beginning to be used for working with program code. The article systematically describes information security threats and possible attacks on systems using these models. Problematic issues requiring further research are also presented.*

References

1. Azim M. Best Open Source LLMs for Code Generation in 2025. – Internet-resurs. – Rezhim dostupa: https://www.cubix.co/blog/best-open-source-llms-for-code-generation-in-2025. – Vremja dostupa: 31.03.2025.
2. Gribunin V. G., Kondakov S. E. K voprosu o zashhite informacii v intellektualizirovannyh obrazcah vooruzhenija // Voprosy kiberbezopasnosti. – 2021. – № 5(45). – Str. 5–11. DOI:10.21681/2311-3456-2021-5-5-11.
3. Schuster R., Song C., Tromer E., Shmatikov V. You Autocomplete Me: Poisoning Vulnerabilities in Neural Code Completion // Proceedings of the 30th USENIX Security Symposium. USENIX Association, Canada. – 2021, pp. 1559–1575.
4. Gu T., Dolan-Gavitt B., Garg S. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain // Rezhim dostupa: arXiv abs/1708.06733. DOI:10.48550/arXiv.1708.06733.
5. Chen Y. i dr. Security of Language Models for Code: A Systematic Literature Review // Rezhim dostupa: https://arxiv.org/pdf/2410.15631. – Vremja dostupa: 31.03.2025.
6. Alzantot M., Sharma Y., Elgohary A. i dr. Generating Natural Language Adversarial Examples // Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Brussels, Belgium. – Pp.2890–2896. DOI:10.48550/arXiv.1804.07998.
7. Wan Y., Zhang S., Zhang H. i dr. You see what I want you to see: poisoning vulnerabilities in neural code search // Proceedings of the 30th ACM foint European Software Engineering Conference and Symposium on the Foundations of Software Engineering. ACM, Singapore. – Pp.1233–1245. DOI:10.1145/3540250.3549153.
8. Ramakrishnan G., Albarghouthi A. Backdoors in Neural Models of Source Code // Proceedings of the 26th International Conference on Pattern Recognition. IEEE, Canada. – Pp.2892–2899. DOI:10.1109/ICPR56361.2022.9956690.

5 Vadim G. Gribunin, Dr.Sc. (Tech.), Associate Professor, Chief Researcher, Institute of Engineering Physics, Serpukhov, Moscow Region, Russia. E-mail: wavelet2@mail.ru
6 Sergey A. Mayorov, Ph.D. (Tech.), Senior Researcher, Scientific and Methodological Department, Institute of Engineering Physics, Serpukhov, Moscow Region, Russia. E-mail: oniokr@iifmail.ru
7 Alexander A. Murashko, Dr.Sc. (Tech.), Associate Professor, Senior Researcher of the Scientific and Methodological Department of the Institute of Engineering Physics, Serpukhov, Moscow Region, Russia. E-mail: oniokr@iifmail.ru