https://doi.org/10.31891/2307-5732-2025-355-107
УДК 004.738.5

ХОМУТНИК ДМИТРО
Національний університет "Львівська Політехніка"
https://orcid.org/0000-0002-3130-8839
e-mail: dmytro.y.khomutnyk@lpnu.ua
ПУКАЧ АНДРІЙ
Національний університет "Львівська Політехніка"
https://orcid.org/0009-0001-8563-3311
e-mail: andriy.i.pukach@lpnu.ua

МЕТОД ГЕНЕРАЦІЇ ХМАРНОЇ ІНФРАСТРУКТУРИ НА ОСНОВІ ВИСОКОРІВНЕВОГО ОПИСУ ПРИРОДНОЮ МОВОЮ ТА ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

У статті розглядається проблему автоматизованої генерації описів хмарної інфраструктури на основі вхідних запитів, сформульованих природною мовою. Традиційні підходи до реалізації подібних задач базуються на використанні засобів інфраструктури як коду (Infrastructure-as-Code, IaC), зокрема мови Terraform, яка вимагає дотримання точного синтаксису, володіння спеціалізованими знаннями та розуміння великої кількості параметрів низького рівня. Унаслідок цього, навіть створення відносно простої інфраструктури є складним і схильним до помилок процесом, особливо для користувачів без глибокої технічної підготовки.

У межах дослідження запропоновано метод, який поєднує можливості великих мовних моделей (ВММ) із проміжною високорівневою мовою опису, що дозволяє зменшити обсяг коду, підвищити читабельність та зменшити кількість помилок при генерації інфраструктури. Побудовано формальну модель процесу, яка охоплює формулювання запиту від користувача природною мовою, генерацію структурованого опису та його трансляцію в код Terraform.

Для оцінювання ефективності запропонованого методу проведено порівняльний експеримент, що включає три тестові сценарії зростаючої складності: від розгортання одного приватного сервера до масштабованого набору публічних серверів за балансувальником навантаження. Кожен сценарій реалізовано двома способами — шляхом прямої генерації коду Terraform та через проміжну мову. Порівняння здійснювалося за такими критеріями, як кількість синтаксичних і семантичних помилок, довжина згенерованого коду та кількість визначених ресурсів.

Результати експерименту свідчать, що використання проміжної мови суттєво зменшує ймовірність синтаксичних і семантичних помилок, скорочує обсяг опису та підвищує надійність і зрозумілість кінцевих конфігурацій. Крім того, високий рівень абстракції опису полегшує процес навчання для мовних моделей, дозволяючи їм генерувати більш коректний код з меншою кількістю інструкцій.

Запропонований метод є актуальним у контексті спрощення DevOps-процесів, тестування хмарної інфраструктури та інтеграції генеративних моделей у середовище розробки. Перспективи подальших досліджень включають донавчання моделей на доменно-специфічних наборах даних та розширення проміжної мови новими типами хмарних ресурсів.

Ключові слова: хмарна інфраструктура, нейромережі, великі мовні моделі, інфраструктура як код, генерація коду.

KHOMUTNYK DMYTRO
PUKACH ANDRII
Lviv Polytechnic National University

A METHOD FOR GENERATING CLOUD INFRASTRUCTURE BASED ON HIGH-LEVEL NATURAL LANGUAGE DESCRIPTIONS AND LARGE LANGUAGE MODELS

This article investigates the problem of automated generation of cloud infrastructure descriptions from natural language user input. Traditional approaches rely on Infrastructure-as-Code (IaC) tools such as Terraform, which require precise syntax, domain-specific knowledge, and familiarity with numerous low-level configuration parameters. As a result, defining even relatively simple infrastructure components can be complex and error-prone, especially for users without deep technical expertise.

The study proposes a method that combines the capabilities of large language models (LLMs) with an intermediate high-level description language. This language reduces code volume, improves readability, and decreases the number of errors during infrastructure generation. A formal model of the process is developed, encompassing natural language query formulation, generation of a structured infrastructure description, and its translation into Terraform code.

A comparative evaluation is conducted using three test scenarios of increasing complexity, ranging from the deployment of a single private server to a scalable group of public servers behind a load balancer. The two approaches — direct Terraform generation and generation via the intermediate language — are compared based on several criteria: number of syntax and semantic errors, code length, and resource count. The results show that the intermediate representation reduces errors, shortens the code, and increases generation reliability.

The experimental results demonstrate that using the intermediate language significantly reduces the likelihood of both syntactic and semantic errors, lowers code verbosity, and produces more reliable and interpretable configurations. The use of a high-level abstraction also simplifies the learning process for language models, enabling them to generate valid infrastructure code with fewer instructions.

The proposed method is relevant for simplifying DevOps processes, infrastructure testing, and integrating generative models into development environments. Future research may include domain-specific model fine-tuning and expansion of the intermediate language with new resource types.

Keywords: cloud infrastructure, neural networks, LLM, infrastructure as code, code generation.

Стаття надійшла до редакції / Received 29.05.2025
Прийнята до друку / Accepted 26.06.2025

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями

Проєктування розподілених систем і створення ресурсів хмарних інфраструктур, що їм відповідають, є одним з основних завдань сучасної веб-розробки. Зі зростанням вимог до масштабованості, доступності та

надійності, розробники все частіше звертаються до розподілених обчислювальних моделей, які забезпечують ефективну обробку даних та надання послуг у глобальному масштабі. Це підтверджується дослідженнями, які підкреслюють важливість інтеграції розподілених систем у хмарні обчислення для задоволення сучасних потреб [1].

Підхід "Інфраструктура як код" спростив процес управління хмарними ресурсами, дозволяючи описувати інфраструктуру за допомогою декларативного коду. Однак, навіть з використанням таких інструментів, як Terraform, процес створення інфраструктури залишається складним та вимагає глибоких технічних знань. Це пов'язано з необхідністю точного визначення параметрів ресурсів та їх взаємозв'язків, що може призводити до помилок та знижувати ефективність розробки [2].

В роботі [3] було запропоновано спосіб опису ресурсів хмарних інфраструктур і їхніх взаємозв'язків за допомогою високорівневої абстрактної мови опису. Однак навіть такі абстракції вимагають певних технічних знань від користувача для ефективного використання.

З огляду на це, виникає потреба у пошуку нових підходів, які б дозволили спростити процес створення хмарної інфраструктури. Одним з перспективних напрямів є використання великих мовних моделей (ВММ), які здатні інтерпретувати природну мову та генерувати відповідний код. Завдяки навчанню на великих обсягах даних, ВММ можуть містити інформацію про хмарних провайдерів, ресурси хмарної інфраструктури та їхнє створення, що відкриває можливості для автоматизації процесу опису інфраструктури [3].

Таким чином, ВММ можуть виступати в ролі інтерфейсу між намірами користувача та програмною реалізацією бажаної хмарної інфраструктури, що дозволяє зменшити технічні бар'єри та підвищити ефективність розробки.

Аналіз останніх досліджень і публікацій

Останні наукові роботи на тему використання ВММ для генерування коду опису інфраструктури досліджують можливість та доцільність різноманітних підходів вирішення цієї задачі.

У роботі [4] досліджується революційність підходу "інфраструктура як код" та наголошується вимогливість до спеціалізованих знань і навичок користувачів таких інструментів, а також необхідність великої кількості ручної роботи для оркестрації такої інфраструктури. Застосування можливостей ВММ з розуміння природної мови та слідування інструкціям, а також можливість розуміти і генерувати програмний код, розглядаються як потенційне полегшення і покращення роботи з автоматизації опису і роботи з ресурсами хмарної інфраструктури. У публікації також досліджуються завади для подальшого розвитку такого використання ВММ, де однією з головних проблем є обмежена кількість коду мовою Terraform в наборі тренувальних даних нейромережі. Це призводить до того, що ВММ може згенерувати синтаксично правильний опис з семантичними і логічними помилками.

Публікація [5] досліджує ефективність згенерованого опису хмарної інфраструктури з використанням ВММ. Автори використовували підхід циклу зі зворотнім зв'язком, у якому нейромережа враховувала помилки при інтерпретації згенерованого коду опису хмарної інфраструктури і генерувала виправлений код. Автори дійшли висновку, що з часом зворотній зв'язок стає все менш ефективним і в якийсь момент зовсім перестає приносити позитивні результати.

Робота [6] представляє технологічно незалежну платформу для автоматичного виправлення скриптів IaC. Вона дозволяє використовувати різні джерела інформації для керування процесом автоматичного виправлення, що підвищує ефективність та універсальність підходу. Автори демонструють високу успішність підходу на великому наборі сценаріїв виправлення. Однак ця задача стосується виправлення вже створеного опису, а не його коректна генерація.

Таким чином, аналіз останніх досліджень вказує на те, що генерація опису хмарної інфраструктури із застосуванням існуючих мов IaC залишається складною та трудомісткою задачею, яка потребує високого рівня знань і досвіду. Використання ВММ для вирішення цієї задачі має великий потенціал, але виявляє ряд суттєвих недоліків, які потребують подальших досліджень та нових підходів для підвищення точності і надійності згенерованих конфігурацій хмарних інфраструктур.

Формулювання цілей статті

Метою роботи є: представлення моделі генерації хмарних інфраструктур на основі великих мовних моделей, та дослідження використання проміжної мови опису для кращої реалізації запропонованої моделі. Зокрема, стаття зосереджується на використанні високорівневої абстрактної мови опису як проміжного представлення між природною мовою та низькорівневим кодом Terraform. Дослідження спрямоване на оцінку того, чи сприяє такий метод зменшенню кількості логічних і семантичних помилок у згенерованому коді.

Виклад основного матеріалу

В якості моделі генерації опису хмарної інфраструктури створено модель на основі штучних нейронних мереж та формалізованої форми опису, що дозволяє генерувати опис хмарної інфраструктури виходячи з запиту користувача природною мовою. Генерація відповідного опису здійснюється за допомогою великих мовних моделей, що реалізують стохастичне відображення з вхідного опису до структурованої синтаксично коректної конфігурації.

Нехай L_{desc} – мова опису хмарної інфраструктури, задана граматикою:

$$G = (N, \Sigma, P, S) \quad (1)$$

де:

— N – скінченна множина нетермінальних символів;
— Σ – скінченна множина термінальних символів;
— P – множина правил продукції у форматі $A \to \alpha$, де $A \in N, \alpha \in (N \cup \Sigma)^*$;
— $S \in N$ – початковий символ.

Приклад тексту граматики проміжної мови, описаної в роботі [3], представлений на рис. 1. Ця граматика визначає синтаксичні правила побудови опису, що допускає автоматичну валідацію, трансляцію та подальшу компіляцію у низькорівневу мову (наприклад, Terraform).

```
<description> ::= <resource>+
<resource> ::= <vpc> | <subnet> | <ec2>
<vpc> ::= "VPC" <id> "{" <vpc-attr>* <subnet>* <ec2>* "}"
<subnet> ::= "Subnet" <id> "=" <cidr>
<ec2> ::= "EC2" <id> "{" <ec2-attr>* "}"
```

Рис. 1. Приклад граматики проміжної мови у формі БНФ

Нехай $x \in L_{nat}$ – опис хмарної інфраструктури природною мовою, сформульований користувачем. Модель генерації базується на генеративній штучній нейронній мережі, яка реалізує функцію:

$$f_\theta(x) = arg\ max\ \ P(y \mid x; \theta)$$

де:
— θ – параметри нейронної мережі;
— f_θ – стохастична функція, яку реалізує нейромережева модель;
— $y \in L_{desc}$– опис інфраструктури, згенерований згідно з граматикою G;
— $P(y \mid x; \theta)$ – умовна ймовірність того, що правильний опис інфраструктури – це y, якщо на вхід подано запит x, за поточними параметрами моделі θ;
— arg max – оператор, який вибирає такий y, що максимізує умовну ймовірність.

Алгоритм генерації опису з використанням проміжної мови

Процес генерації опису хмарної інфраструктури з використанням проміжної мови реалізується за допомогою наступного алгоритму:

1. Отримання вхідних даних від користувача. Це загальний опис інфраструктури, яку хоче створити користувач.
2. Генерування опису інфраструктури проміжною мовою. Штучна нейронна мережа генерує формалізований опис хмарної інфраструктури виходячи з опису користувача.
3. Транслювання проміжної мови в Terraform. Сама хмарна інфраструктура створюється за допомогою Terraform і його низькорівневої мови опису.

Блок-схема описаного алгоритму представлена на рис. 2.

Рис. 2. Алгоритм генерації опису хмарної інфраструктури

В якості проміжної використовується мова, що оперує абстракціями більш високого рівня, ніж Terraform. Це дозволяє більш інтуїтивно структурувати елементи хмарних інфраструктур в самому описі, а також використовувати значно менше деталей для їхнього коректного опису. Детальний опис мови представлений в роботі [3]. На рис. 3 представлено опис базової інфраструктури, у якій визначено віртуальний сервер, що розміщується в одній підмережі. На рис. 3.а представлено опис проміжною мовою, а на рис. 3.б відповідний опис мовою Terraform.

```
VPC MyVPC {
  cidr_block = "192.12.0.0/16"
  region = "eu-west-1"

  Subnet MySubnet {
    EC2 MyServer {
      ami = "ami-0f29c8402f8cce65c"
      instanceType = "t3.micro"

      networking {
        ingress = all
      }
    }
  }
}
```

а)

```
resource "aws_vpc" "MyVPC" {
  cidr_block = "192.12.0.0/16"
}

resource "aws_subnet" "MySubnet" {
  vpc_id            = aws_vpc.MyVPC.id
  cidr_block        = "192.12.1.0/24"
  availability_zone = "eu-west-1a"
}

resource "aws_security_group" "MySecurityGroup" {
  name   = "my-security-group"
  vpc_id = aws_vpc.MyVPC.id

  ingress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

resource "aws_network_interface" "MyNetworkInterface" {
  subnet_id       = aws_subnet.MySubnet.id
  security_groups = [aws_security_group.MySecurityGroup.id]
  private_ips     = ["192.12.1.1"]
}

resource "aws_instance" "MyInstance" {
  ami           = "ami-0f29c8402f8cce65c"
  instance_type = "t3.micro"

  network_interface {
    network_interface_id = aws_network_interface.MyNetworkInterface.id
    device_index         = 0
  }
}
```

б)

Рис. 3. Опис хмарної інфраструктури з одним віртуальним сервером, що знаходиться в одній підмережі: а) - опис проміжною мовою; б) - опис мовою Terraform

Порівняння ефективності методів генерації описів

Для експериментальної перевірки переваг використання проміжної мови опису над генеруванням опису мовою Terraform напряму було сформовано три тестових сценарії. Кожен сценарій представляє собою узагальнений і не деталізований опис хмарної інфраструктури, який надається ВММ як запит для початку генерації формального опису. Усі запити надаються англійською мовою, так як ВММ надають кращі результати з використанням саме англійської мови [7].

1. “Create a simple server in Frankfurt (eu-central-1). It should run Linux and be accessible via SSH. No internet access needed. Place it in a private subnet.” Результатом виконання цього запиту має бути згенерована хмарна інфраструктура з одним віртуальним сервером зі встановленою операційною системою Linux та коректно налаштованим зовнішнім доступом через SSH Інтернет-порти.
2. “Create a public web server in us-east-1 using Amazon Linux. It should support HTTP and HTTPS traffic and have a public IP address.” Результатом виконання цього запиту має бути згенерована хмарна інфраструктура з одним віртуальним сервером з публічним доступом до нього через веб-протоколи HTTP і HTTPS. Відповідно, має бути коректно налаштований доступ через публічну IP-адресу до цього сервера.
3. “Create three Ubuntu servers in eu-west-1 behind a load balancer. They should support HTTP and be automatically distributed across availability zones.” Результатом виконання цього запиту має бути згенерована хмарна інфраструктура з віртуальним сервером, що автоматично реплікується в одному регіоні через динамічного розподілювача навантаження. Хмарний провайдер надає змогу автоматично збільшувати кількість реплік веб-сервера в залежності від кількості зайнятих ресурсів вже наявних серверів.

Сценарії відсортовані в порядку зростання складності їхньої реалізації. На їх основі було проведено тестування обох методів генерування опису інфраструктур.

Для реалізації запропонованого методу генерування було використано велику мовну модель з відкритими вагами DeepSeek R1 з 7 мільярдами параметрів. Дана модель використовує підхід “chain-of-thought”, описаний в роботі [8], для самокорекції результатів генерування.

Для порівняння і аналізу обох методів було обрано наступні критерії оцінювання:

1. Кількість рядків згенерованого опису.
2. Кількість явно описаних ресурсів.

3. Кількість синтаксичних помилок. Без їх виправлення Terraform не дозволить створення хмарної інфраструктури, а компілятор проміжної мови не транслює такий опис в мову Terraform. Сюди входять некоректні назви ресурсів чи їхніх конфігурацій.
4. Кількість семантичних помилок. Такі помилки включають відсутність необхідних чи наявність зайвих ресурсів, хибність їхніх конфігурацій тощо.

Для попереднього навчання ВММ використовувались інструкції, що включають в себе документацію необхідних хмарних ресурсів та їх параметрів для обох мов опису. Виходячи з припущення, що запропонована проміжна мова повинна полегшувати процес генерування коректних описів для ВММ, інструкції для обох методів були обмежені до однакової кількості символів. Таким чином частково перевіряється більша простота "пояснення" принципів використання проміжної мови для генеративних нейромереж.

Для більшої надійності результатів, тестування було проведено по 10 разів для кожного сценарію. Медіанні значення результатів обраних критеріїв оцінки для обох методів представлені в таблиці 1.

Таблиця 1

Результати генерування описів хмарних інфраструктур з тестових сценаріїв (медіанні значення)

Сценарій	Підхід	Кількість рядків	Кількість ресурсів	Кількість синтаксичних помилок	Кількість семантичних помилок
1	Terraform	33	5	2	3
1	Проміжна мова	14	3	1	1
2	Terraform	41	6	3	3
2	Проміжна мова	15	3	1	1
3	Terraform	69	9	8	4
3	Проміжна мова	41	6	2	2

На рис. 3 представлено графічне порівняння результатів генерування описів хмарних інфраструктур для тестових сценаріїв окремо для кожного критерію тестування.

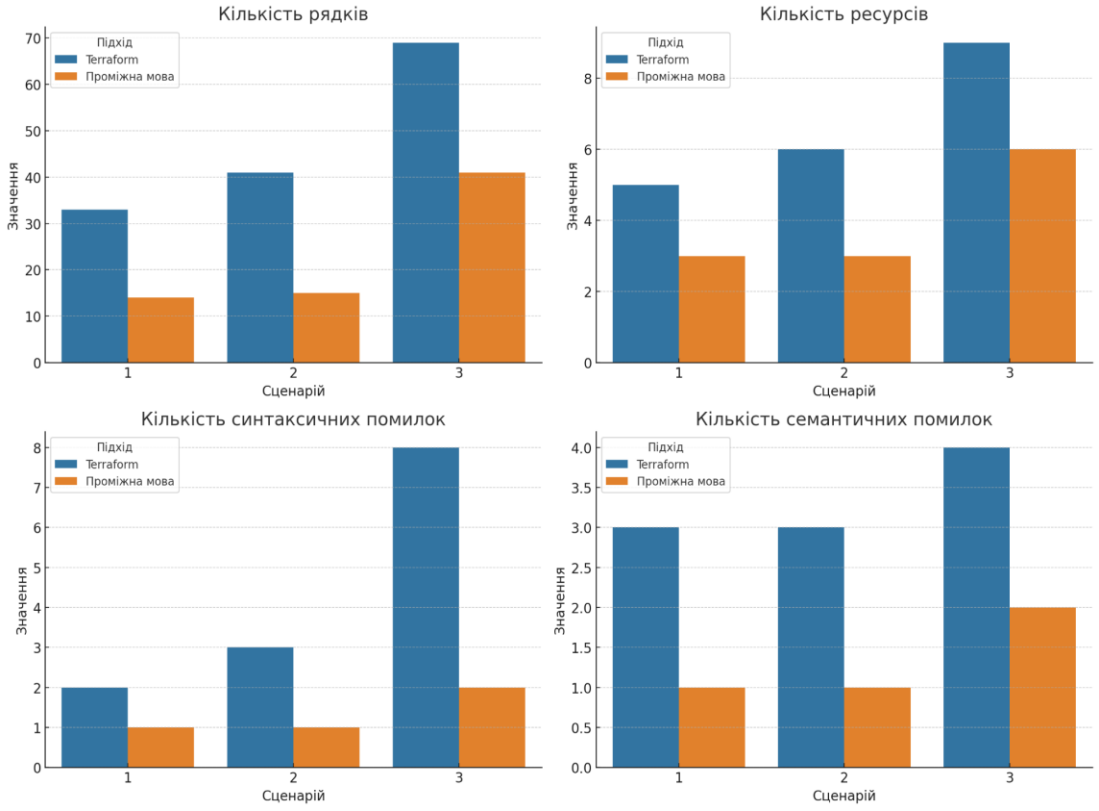

Рис. 3. Графічне порівняння результатів генерації описів двома методами

Висновки

У статті запропоновано метод генерування хмарної інфраструктури на основі загального опису природною мовою з використанням великих мовних моделей та проміжної високорівневої мови опису. Проведений експеримент показав, що застосування проміжної мови значно знижує кількість синтаксичних і семантичних помилок у згенерованому коді та зменшує кількість необхідних рядків опису. Варто зазначити, що експериментальна перевірка проводилась на базових прикладах хмарних інфраструктур, і результати свідчать про доцільність використання абстрактних описів як ефективного інтерфейсу між природною мовою та інструментами типу Terraform.

Крім того, дослідження підтвердило, що високорівнева структура опису спрощує навчання нейромереж і дозволяє досягти вищої точності генерації навіть із менш деталізованими інструкціями. Зокрема, медіанні значення ключових показників — таких як кількість помилок та обсяг згенерованого коду — продемонстрували значні переваги підходу на основі проміжної мови у порівнянні з прямою генерацією Terraform-конфігурацій.

Враховуючи обмеження експериментального дизайну, перспективними напрямками подальших досліджень є донавчання моделей на спеціалізованих наборах даних (мови опису та приклади), розширення граматики проміжної мови новими типами ресурсів та інтеграція методу в автоматизацію створення розподілених систем.

Таким чином, запропонований метод є вагомим внеском у галузь автоматизації розгортання хмарних інфраструктур, особливо у контексті зростаючого застосування генеративного ШІ в інженерних практиках.

Література

1. Ageed Z. Distributed Systems Meet Cloud Computing: A Review of Convergence and Integration / Ageed Zainab, Zeebaree Subhi // International Journal of Intelligent Systems and Applications in Engineering. – 2024. – Vol. 12, No. 11s. – P. 469–490.

2. Pathak, Avinash. Automating Infrastructure Management: Benefits and Challenges of Ansible and Terraform Implementation Across Sectors // International Journal of Scientific Research in Computer Science, Engineering and Information Technology. – 2024. – Vol. 10, No. 5. – P. 1032–1037. – DOI: https://doi.org/10.32628/CSEIT241051032

3. Хомутник Д. Ю. Високорівневий спосіб опису ресурсів хмарної інфраструктури / Дмитро Хомутник, Олександр Марченко // Комп'ютерно-інтегровані технології: освіта, наука, виробництво. – 2022. – №48. – С. 117–123. – DOI: https://doi.org/10.36910/6775-2524-0560-2022-48-18

4. Srivatsa K. G. A Survey of Using Large Language Models for Generating Infrastructure as Code / Kalahasti Ganesh Srivatsa, Sabyasachi Mukhopadhyay, Ganesh Katrapati, Manish Shrivastava // arXiv preprint. – 2024. – arXiv:2404.00227. – DOI: https://doi.org/10.48550/arXiv.2404.00227

5. Palavalli M. A. Using a Feedback Loop for LLM-based Infrastructure as Code Generation / Mayur Amarnath Palavalli, Mark Santolucito // arXiv preprint. – 2024. – arXiv:2411.19043. – DOI: https://doi.org/10.48550/arXiv.2411.19043

6. Saavedra N. InfraFix: Technology-Agnostic Repair of Infrastructure as Code / Nuno Saavedra, João F. Ferreira, Alexandra Mendes // arXiv preprint. – 2025. – arXiv:2503.17220. – DOI: https://doi.org/10.48550/arXiv.2503.17220

7. Julen E. Do Multilingual Language Models Think Better in English? / Julen Etxaniz, Gorka Azkune, Aitor Soroa, Oier Lopez de Lacalle, Mikel Artetxe // arXiv preprint. – 2023. – arXiv:2308.01223. – DOI: https://doi.org/10.48550/arXiv.2308.01223

8. DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning // arXiv preprint. – 2025. – arXiv:2501.12948. – DOI: https://doi.org/10.48550/arXiv.2501.12948

References

1. Ageed Z. Distributed Systems Meet Cloud Computing: A Review of Convergence and Integration / Ageed Zainab, Zeebaree Subhi // International Journal of Intelligent Systems and Applications in Engineering. – 2024. – Vol. 12, No. 11s. – P. 469–490.

2. Pathak, Avinash. Automating Infrastructure Management: Benefits and Challenges of Ansible and Terraform Implementation Across Sectors // International Journal of Scientific Research in Computer Science, Engineering and Information Technology. – 2024. – Vol. 10, No. 5. – P. 1032–1037. – DOI: https://doi.org/10.32628/CSEIT241051032

3. D. Khomutnyk. High-level technique for description of cloud infrastructure resources / Dmytro Khomutnyk, Oleksandr Marchenko // Computer-Integrated Technologies: Education, Science, Production. – 2022. – №48. – P. 117–123. – DOI: https://doi.org/10.36910/6775-2524-0560-2022-48-18

4. Srivatsa K. G. A Survey of Using Large Language Models for Generating Infrastructure as Code / Kalahasti Ganesh Srivatsa, Sabyasachi Mukhopadhyay, Ganesh Katrapati, Manish Shrivastava // arXiv preprint. – 2024. – arXiv:2404.00227. – DOI: https://doi.org/10.48550/arXiv.2404.00227

5. Palavalli M. A. Using a Feedback Loop for LLM-based Infrastructure as Code Generation / Mayur Amarnath Palavalli, Mark Santolucito // arXiv preprint. – 2024. – arXiv:2411.19043. – DOI: https://doi.org/10.48550/arXiv.2411.19043

6. Saavedra N. InfraFix: Technology-Agnostic Repair of Infrastructure as Code / Nuno Saavedra, João F. Ferreira, Alexandra Mendes // arXiv preprint. – 2025. – arXiv:2503.17220. – DOI: https://doi.org/10.48550/arXiv.2503.17220

7. Julen E. Do Multilingual Language Models Think Better in English? / Julen Etxaniz, Gorka Azkune, Aitor Soroa, Oier Lopez de Lacalle, Mikel Artetxe // arXiv preprint. – 2023. – arXiv:2308.01223. – DOI: https://doi.org/10.48550/arXiv.2308.01223

8. DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning // arXiv preprint. – 2025. – arXiv:2501.12948. – DOI: https://doi.org/10.48550/arXiv.2501.12948