Вот таблица с популярными датасетами для суммаризации кода, их характеристиками, метриками и источниками:

### Ключевые метрики:

1. \*\*BLEU (Bilingual Evaluation Understudy)\*\* — измеряет n-граммное совпадение с эталоном .

2. \*\*METEOR\*\* — учитывает синонимы и порядок слов .

3. \*\*ROUGE (Recall-Oriented Understudy)\*\* — фокусируется на полноте совпадений .

4. \*\*CodeBLEU\*\* — адаптация BLEU для кода с учетом синтаксиса .

5. \*\*Человеческая оценка\*\* — точность, читаемость, консистентность .

### Рекомендации по выбору:

- Для Java-проектов: \*\*FunCom\*\* или \*\*TL-CodeSum\*\* .

- Для Python: \*\*Python Docstrings\*\* или \*\*PYMT5\*\* .

- Для кросс-языковых задач: \*\*CodeSearchNet\*\* .

Большинство датасетов доступны на платформах вроде GitHub, Hugging Face или через официальные публикации. Примеры использования и предобработки можно найти в статьях .



1. **CodeSearchNet**:

- **Описание**: Создан компанией Microsoft Research, этот датасет содержит около 6 миллионов функций на шести языках программирования (Python, Java, JavaScript, Go, PHP, Ruby). Каждая функция сопровождается описанием на естественном языке.

- **Источник**: https://github.com/github/codesearchnet

- **Метрика**: BLEU, ROUGE

2. **FunCom**:

- **Описание**: Этот датасет предназначен для генерации комментариев к функциям на Python. Содержит реальные функции и соответствующие им комментарии, написанные людьми.

- **Источник**: https://zenodo.org/record/3240524#.Yy8qJXbMLIU

- **Метрика**: BLEU, ROUGE

3. **CoNaLa**:

- **Описание**: CoNaLa (Conversational Network Analysis and Learning) – это набор данных, предназначенный для изучения разговорных взаимодействий между программистами. Включает примеры кода и их описания, а также обсуждения о решении задач.

- **Источник**: https://conala-corpus.github.io/

- **Метрика**: BLEU, ROUGE

4. **Docstring Dataset**:

- **Описание**: Содержит пары «код-докстринг» на Python. Докстринги представляют собой комментарии, описывающие назначение и использование функции или метода.

- **Источник**: https://arxiv.org/pdf/2009.08378.pdf

- **Метрика**: BLEU, ROUGE

5. **JavaDoc Dataset**:

- **Описание**: Аналогичный датасет для языка Java, включающий пары «метод-документация».

- **Источник**: https://huggingface.co/datasets/javadoc

- **Метрика**: BLEU, ROUGE

6. **StackOverflow Code Snippets**:

- **Описание**: Хотя StackOverflow не является специализированным датасетом для суммаризации кода, он может служить источником примеров кода и связанных с ними вопросов и ответов. Эти данные могут быть использованы для создания обучающих выборок.

- **Источник**: https://archive.org/download/stackexchange

- **Метрика**: BLEU, ROUGE

В задачах суммаризации кода используются различные метрики для оценки качества сгенерированных описаний. Помимо стандартных NLP-метрик (например, ROUGE, BLEU, METEOR), применяются специализированные подходы, учитывающие структуру кода и семантику. Вот основные категории метрик и примеры их использования:

---

### 1. \*\*Традиционные метрики NLP\*\*

Используются для оценки текстовых суммаризаций, но адаптированы для анализа комментариев к коду:

- \*\*ROUGE (Recall-Oriented Understudy for Gisting Evaluation)\*\*

Оценивает полноту совпадения n-грамм между сгенерированным и эталонным текстом. Например, ROUGE-L анализирует наибольшую общую подпоследовательность .

- \*\*BLEU (Bilingual Evaluation Understudy)\*\*

Измеряет точность совпадения n-грамм. Часто применяется BLEU-4 для учета 4-грамм .

- \*\*METEOR\*\*

Учитывает синонимы, порядок слов и морфологию, что полезно для оценки естественности описаний .

---

### 2. \*\*Специализированные метрики для кода\*\*

Учитывают синтаксис и семантику программного кода:

- \*\*CodeBLEU\*\*

Комбинирует BLEU с анализом абстрактных синтаксических деревьев (AST) и совпадением ключевых слов кода (например, переменных, операторов) .

- \*\*Структурное сходство\*\*

Оценивает соответствие структуры сгенерированного описания логике кода (например, через сравнение AST) .

- \*\*Семантическое сходство\*\*

Использует векторные представления кода и текста (например, Code2Vec) для анализа смысловой близости .

---

### 3. \*\*Метрики качества кода\*\*

Применяются для оценки исходного кода, что косвенно влияет на сложность его суммаризации:

- \*\*Цикломатическая сложность\*\*

Определяет количество независимых путей в коде. Высокая сложность может требовать более детальных описаний .

- \*\*Индекс сопровождаемости (Maintainability Index)\*\*

Оценивает легкость поддержки кода (0–100). Низкие значения сигнализируют о необходимости более точных комментариев .

- \*\*Глубина наследования и связывание классов\*\*

Показывают сложность архитектуры, что влияет на ясность описаний .

---

### 4. \*\*Человеческая оценка\*\*

Эксперты анализируют:

- \*\*Точность\*\* — насколько описание соответствует коду.

- \*\*Полнота\*\* — охват ключевых функций.

- \*\*Читаемость\*\* — естественность формулировок .

Примеры критериев из автоматизированных систем верификации:

- \*\*Correctness\*\* — отсутствие ошибок в описании.

- \*\*Relevance\*\* — соответствие контексту кода .

---

### 5. \*\*Гибридные метрики\*\*

Комбинируют несколько подходов:

- \*\*CodeROUGE\*\*

Объединяет ROUGE с анализом AST и данных потока управления .

- \*\*Semantic F1\*\*

Оценивает точность совпадения семантических сущностей (например, функций, переменных) между кодом и описанием .

---

### Примеры датасетов и их метрики

| Датасет | Основные метрики | Источник |

|-----------------------|---------------------------------------|------------------------|

| \*\*CoDesc\*\* | BLEU-4, METEOR, ROUGE-L | [CoDesc, 2021] |

| \*\*Python Docstrings\*\* | BLEU, ROUGE, точность сем. сходства | [Wan et al., 2018] |

| \*\*CodeSearchNet\*\* | CodeBLEU, ROUGE, BLEU | [CodeSearchNet, 2019] |

---

### Рекомендации по выбору метрик

- Для \*\*синтаксической точности\*\*: CodeBLEU, структурное сходство.

- Для \*\*семантической адекватности\*\*: векторные модели (Code2Vec) или человеческая оценка.

- Для \*\*быстрой автоматической проверки\*\*: ROUGE, BLEU .

Подробнее о метриках качества кода можно узнать из статей по анализу ПО , а о подходах к суммаризации — в исследованиях NLP .

```  
from transformers import pipeline

text = '''

What this code is doing?

s = 0

for i in range(10):

s+=i

print(s)

'''

messages = [

{"role": "user", "content": text},

]

pipe = pipeline("text-generation", model="Qwen/Qwen2.5-Coder-1.5B")

print(pipe(messages))  
  
```

```  
from datasets import load\_dataset

import pprint

ds = load\_dataset("codeparrot/xlcost-text-to-code", "C++-program-level")

dataset = ds["train"]

pprint.pprint(dataset[0])

```