AI and FOSS: looking for founding principles

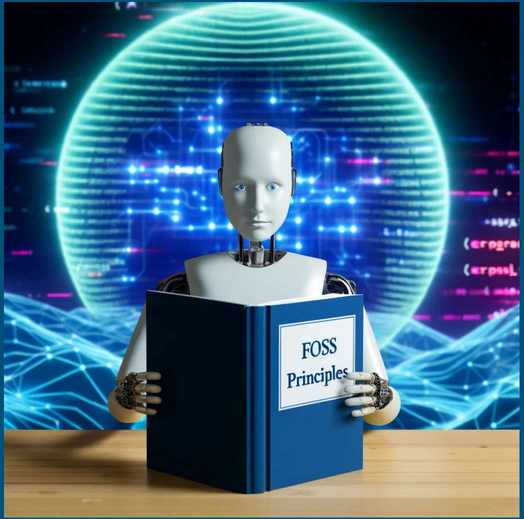


Image created with DALL-E

Roberto Di Cosmo

Computer Science Professor

Director, Software Heritage

https://dicosmo.org

@rdicosmo

Perspectives on AI and code

1. The user
2. The software commons
3. ~~The developers~~ (next time :-))

Preserving user freedom in an AI world

Image created with DALL-E

From the April 2018 LLW workshop in Barcelona

(thanks to Andy Wilson)

How can we preserve the "user freedom", in the spirit of FOSS, in an "automated decision making world"?

- Well into the debate about "ethical principles" for AI
- Way before the Generative AI explosion of interest

Path 1: Explainable AI

Principle: “Any end user that is the object of an automated decision should be given a *human understandable explanation* of why the decision was made.”

It’s the same that we expect from a judgement: not the sequence of neuron firings in the head of a judge, but the “rationale” of the decision.

This seems to be the golden standard *from a user point of view*.

UPDATE: **needs to be refined** with generative AI (what is an *explanation* here?)

Path 2: Accountability and Transparency

Principle: “*When a human understandable explanation cannot be obtained*, it is in general not possible to assess the outcome of an automated decision just by looking at it. Hence we believe that *the whole decision making process should be transparent, and accountable*.”

In the case of machine learning, this includes, in particular,

- the (source code of the) **software** used for the processing
- the trained machine **model**(s)
- the **data** used in the training
- and all information necessary to perform **independent experiments** using all the above

This is a *plan B for a user*, but seems to be the gold standard *for research in ML*.

UPDATE: **very hot topic** with LLMs, see later (also AI act)

Path 3: Ethics

Principle: “When *we cannot control* the outcome of the use of a technology, or we fear *we do not sufficiently understand the consequences*, we should just *refrain from using it.*”

This is a *last resort for a user*, but has been the gold standard *for research in biotechnology, for decades*.

Big question: who makes the decisions?

See e.g. the UNESCO recommendation on ethics for AI

Preserving software commons in a (Gen)AI world

Image created with DALL-E

Software commons ***are massively used*** *for building Large Language Models.*

Independently of what we do, there is no turning back.

The **real question** is *how* they should be built and *whom* they should benefit.

Let's have a candid look around us.

Closed model APIs

OpenAI

ANTHROP\C

Open model weights

Mistral.AI

deepseek

This slide is courtesy of Leandro Von Werra and Harm de Vries

Closed model APIs

✗ Model weights not available

- Can’t run the model locally
- Can’t inspect the model’s representations
- Limits fine-tuning abilities

And more:

- limits user freedom (personal data leakage)

Open model weights

This slide is courtesy of Leandro Von Werra and Harm de Vries

Closed model APIs

✗ Model weights not available

- Can’t run the model locally
- Can’t inspect the model’s representations
- Limits fine-tuning abilities

And more:

- limits user freedom
  (personal data leakage)

Open model weights

✗ Training data is not disclosed

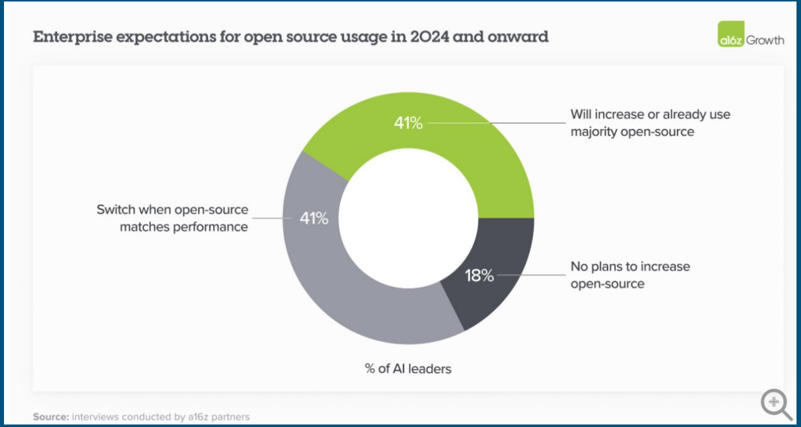
- Content creators don’t know if their data is used
- There’s no way to remove it
- Can’t inspect data for biases
- Potential benchmark contamination

This is not what “open” should mean.

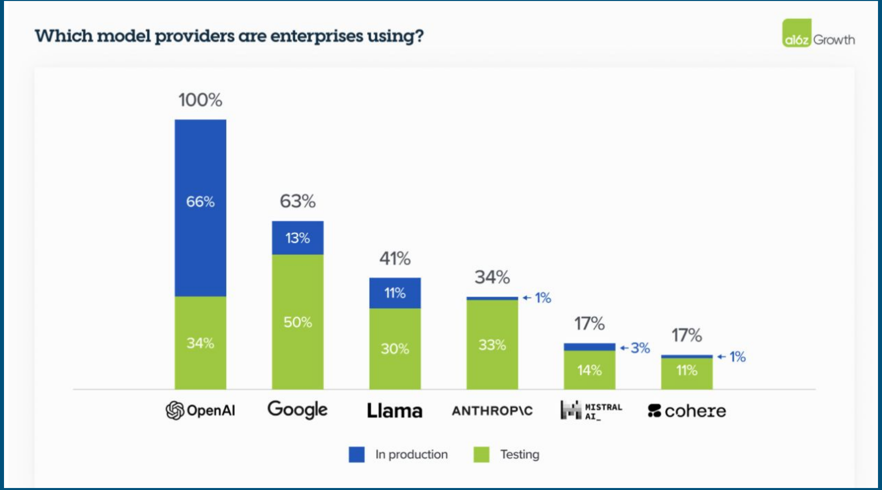
Can we change all this? How?

This slide is courtesy of Leandro Von Werra and Harm de Vries

A window of opportunity



https://a16z.com/generative-ai-enterprise-2024/



Companies want "open source" models (sorry Stefano, not my words!), but…

LLMs follow a winners take all dynamics!

This slide is courtesy of Leandro Von Werra and Harm de Vries

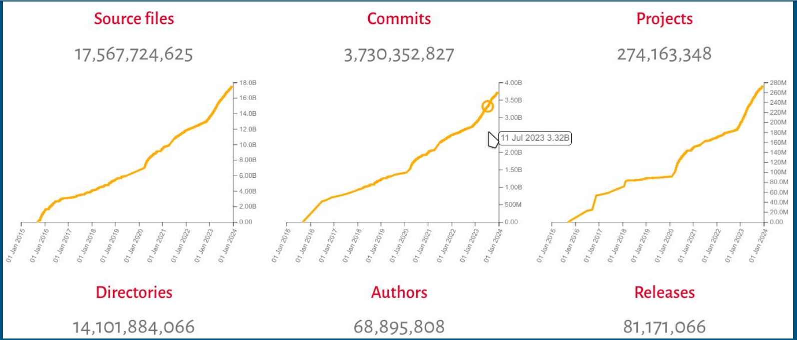
The source code opportunity

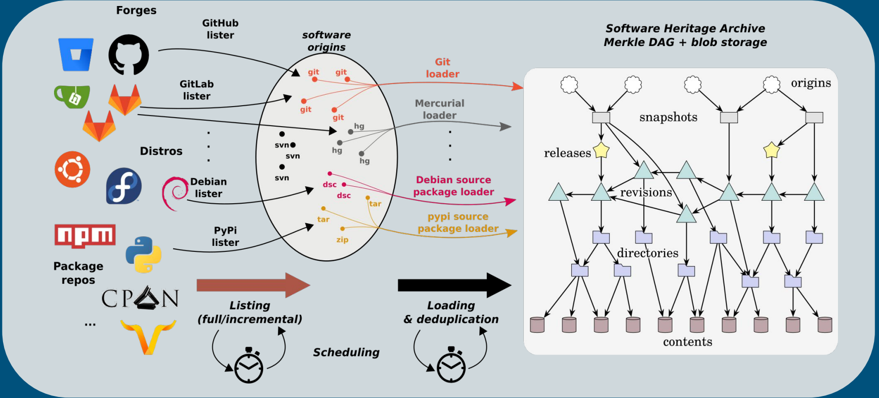
Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Largest archive of source code

digital commons built since 2015

Cultural Heritage | Industry | Research | Public Administration





500+ code hosting platforms

All versions, full development history
In a single giant Merkle Graph

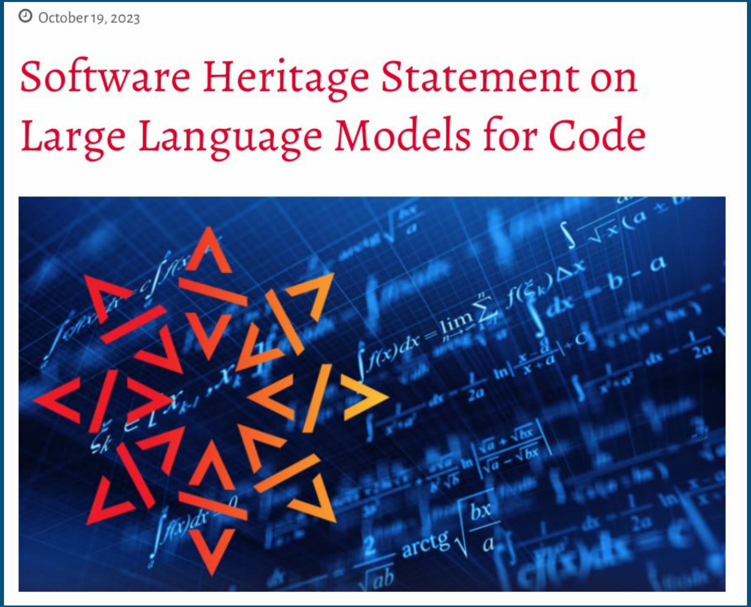
- 35×10^9 nodes
- 500×10^9 edges
~ **2 PB** storage

Ensures **availability**
Guarantees **integrity**
Enables **traceability**

} of all source code

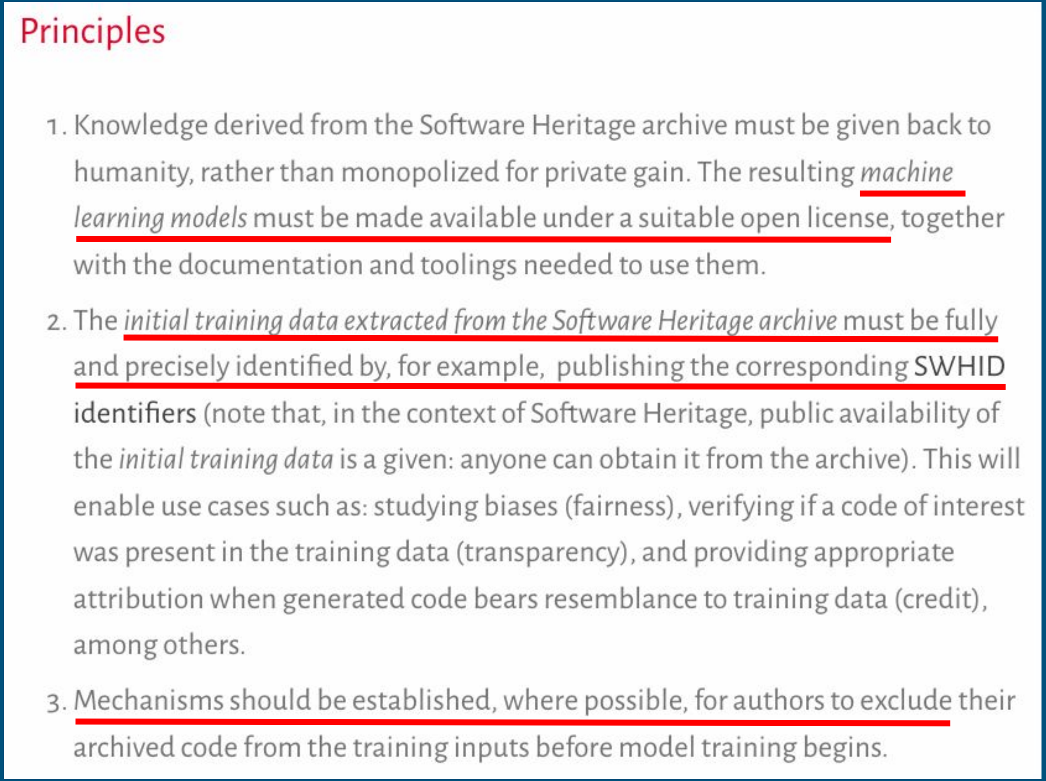
Unique dataset for machine learning,
an infrastructure for transparency and accountability

Looking for founding principles at Software Heritage



October 19, 2023

Software Heritage Statement on Large Language Models for Code

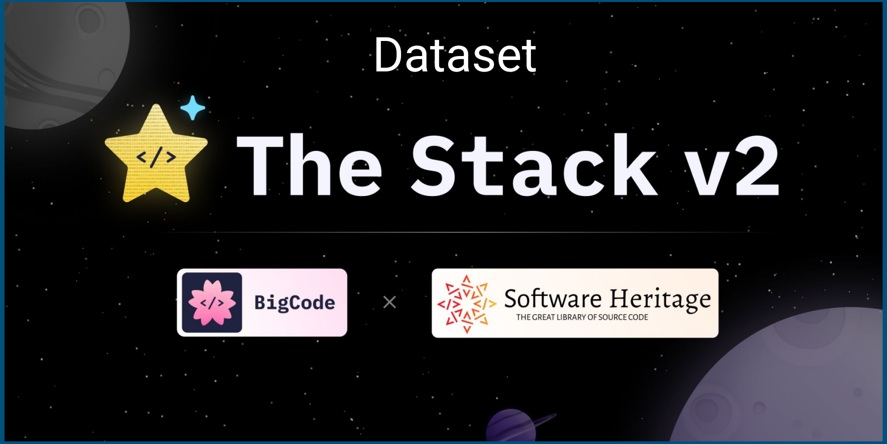


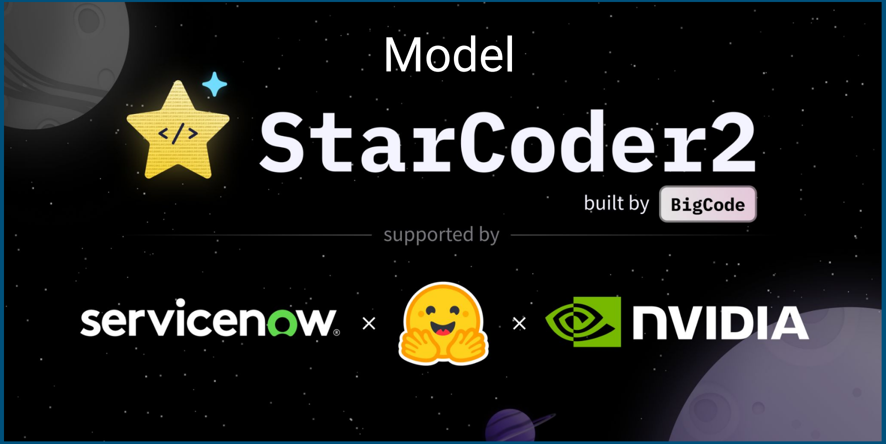
Principles

1. Knowledge derived from the Software Heritage archive must be given back to humanity, rather than monopolized for private gain. The resulting *machine learning models* must be made available under a suitable open license, together with the documentation and toolings needed to use them.
2. The *initial training data extracted from the Software Heritage archive* must be fully and precisely identified by, for example, publishing the corresponding SWHID identifiers (note that, in the context of Software Heritage, public availability of the *initial training data* is a given: anyone can obtain it from the archive). This will enable use cases such as: studying biases (fairness), verifying if a code of interest was present in the training data (transparency), and providing appropriate attribution when generated code bears resemblance to training data (credit), among others.
3. Mechanisms should be established, where possible, for authors to exclude their archived code from the training inputs before model training begins.

Question: are we asking too much?

Findings from BigCode: The Stack v2 and StarCoder2



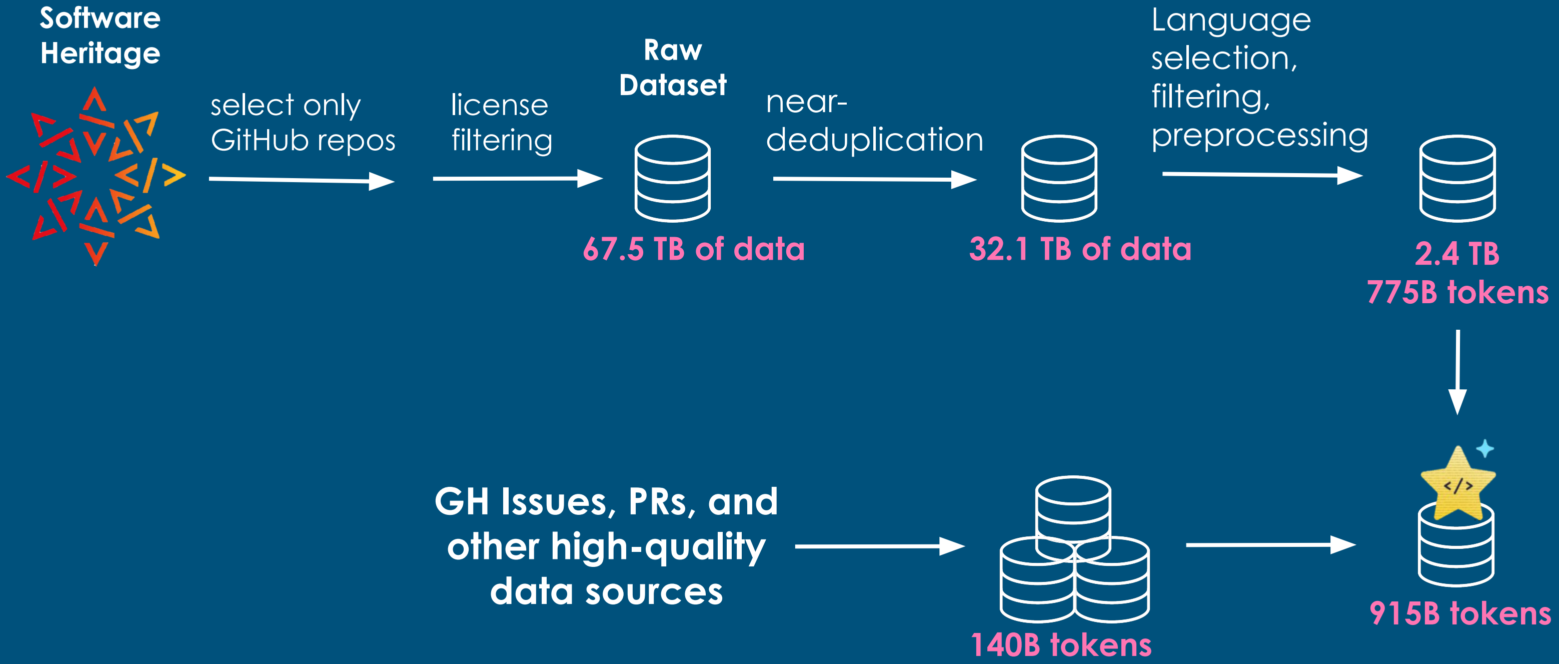


Released February 28th 2024

Yes one can build **the best open LLM for code available** while fully adhering to the Software Heritage principles for responsible LLMs, …
and even more: the full training pipeline is made public too!

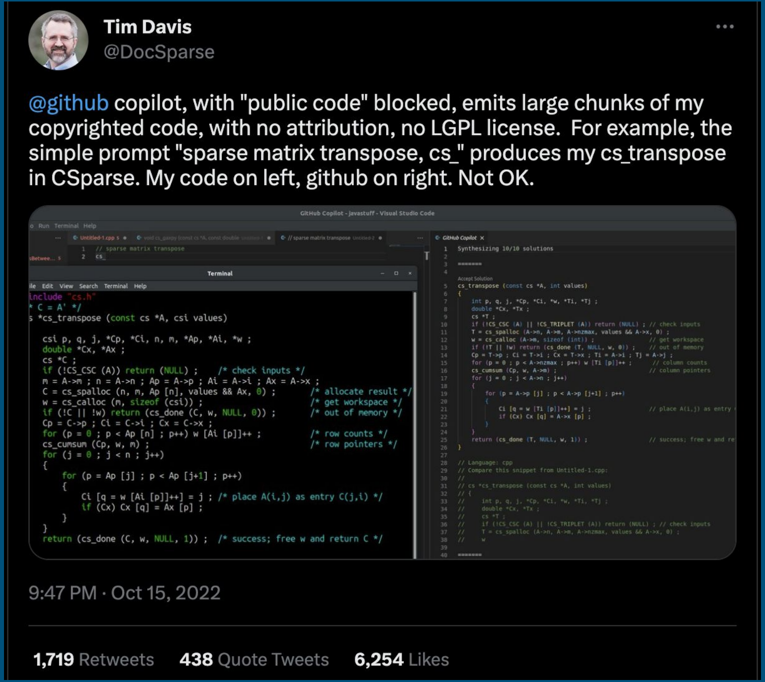
The Stack v2

Data collection pipeline fully open and transparent built by BigCode



This slide is courtesy of Leandro Von Werra and Harm de Vries

I found my (L)GPL code in your dataset!



Tim Davis
@DocSparse

@github copilot, with "public code" blocked, emits large chunks of my copyrighted code, with no attribution, no LGPL license. For example, the simple prompt "sparse matrix transpose, cs_" produces my cs_transpose in CSparse. My code on left, github on right. Not OK.

GitHub Copilot - javastuff - Visual Studio Code
Untitled-1.cpp
// sparse matrix transpose
Terminal
Accept Solution
GitHub Copilot

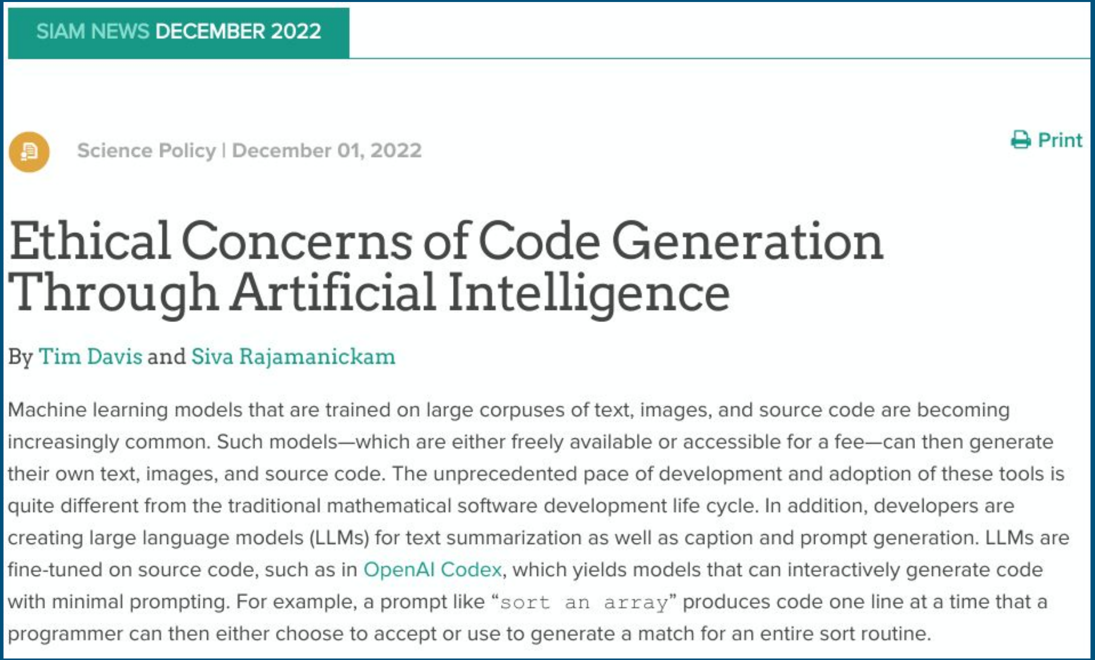
```
// sparse matrix transpose
cs_
```

```
include "cs.h"
* C = A' */
s *cs_transpose (const cs *A, csi values)

   csi p, q, j, *Cp, *Ci, n, m, *Ap, *Ai, *w ;
   double *Cx, *Ax ;
   cs *C ;
   if (!CS_CSC (A)) return (NULL) ;    /* check inputs */
   m = A->m ; n = A->n ; Ap = A->p ; Ai = A->i ; Ax = A->x ;
   C = cs_spalloc (n, m, Ap [n], values && Ax, 0) ;      /* allocate result */
   w = cs_calloc (m, sizeof (csi)) ;                      /* get workspace */
   if (!C || !w) return (cs_done (C, w, NULL, 0)) ;       /* out of memory */
   Cp = C->p ; Ci = C->i ; Cx = C->x ;
   for (p = 0 ; p < Ap [n] ; p++) w [Ai [p]]++ ;          /* row counts */
   cs_cumsum (Cp, w, m) ;                                 /* row pointers */
   for (j = 0 ; j < n ; j++)
   {
       for (p = Ap [j] ; p < Ap [j+1] ; p++)
       {
           Ci [q = w [Ai [p]]++] = j ; /* place A(i,j) as entry C(j,i) */
           if (Cx) Cx [q] = Ax [p] ;
       }
   }
   return (cs_done (C, w, NULL, 1)) ;  /* success; free w and return C */
```

```
Synthesizing 10/10 solutions

=======

cs_transpose (const cs *A, int values)
{
    int p, q, j, *Cp, *Ci, *w, *Ti, *Tj ;
    double *Cx, *Tx ;
    cs *T ;
    if (!CS_CSC (A) || !CS_TRIPLET (A)) return (NULL) ; // check inputs
    T = cs_spalloc (A->n, A->m, A->nzmax, values && A->x, 0) ;
    w = cs_calloc (A->m, sizeof (int)) ;                  // get workspace
    if (!T || !w) return (cs_done (T, NULL, w, 0)) ;    // out of memory
    Cp = T->p ; Ci = T->i ; Cx = T->x ; Ti = A->i ; Tj = A->j ;
    for (p = 0 ; p < A->nzmax ; p++) w [Ti [p]]++ ;      // column counts
    cs_cumsum (Cp, w, A->m) ;                             // column pointers
    for (j = 0 ; j < A->n ; j++)
    {
        for (p = A->p [j] ; p < A->p [j+1] ; p++)
        {
            Ci [q = w [Ti [p]]++] = j ;                   // place A(i,j) as entry
            if (Cx) Cx [q] = A->x [p] ;
        }
    }
    return (cs_done (T, NULL, w, 1)) ;                    // success; free w and re
}

// Language: cpp
// Compare this snippet from Untitled-1.cpp:
//
// cs *cs_transpose (const cs *A, int values)
// {
//     int p, q, j, *Cp, *Ci, *w, *Ti, *Tj ;
//     double *Cx, *Tx ;
//     cs *T ;
//     if (!CS_CSC (A) || !CS_TRIPLET (A)) return (NULL) ; // check inputs
//     T = cs_spalloc (A->n, A->m, A->nzmax, values && A->x, 0) ;
//     w

=======
```

9:47 PM · Oct 15, 2022

1,719 Retweets **438** Quote Tweets **6,254** Likes



SIAM NEWS DECEMBER 2022

Science Policy | December 01, 2022

Print

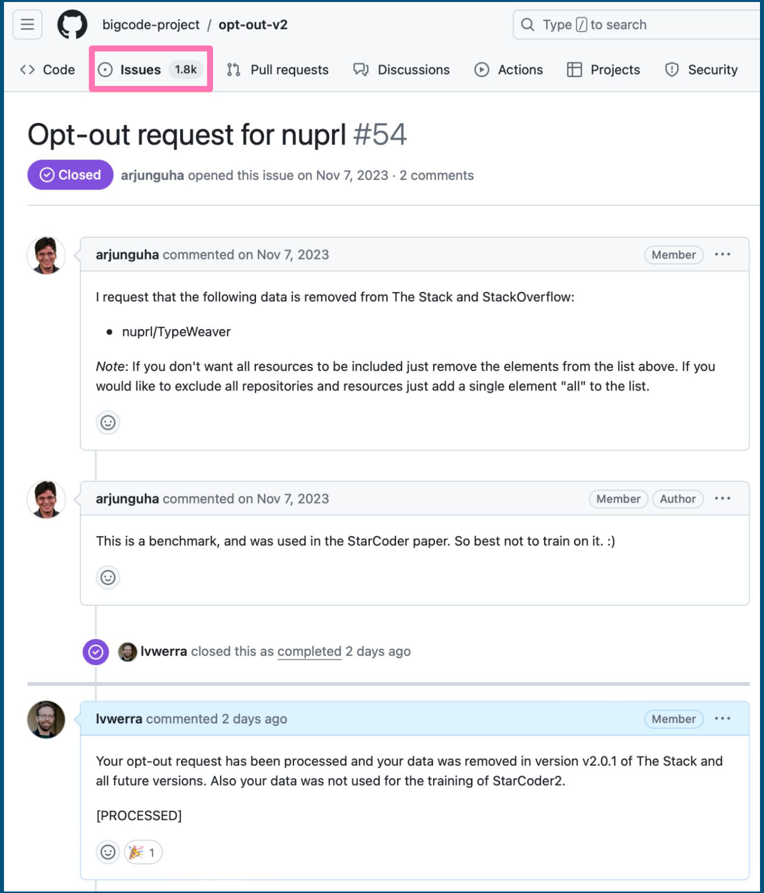
Ethical Concerns of Code Generation Through Artificial Intelligence

By Tim Davis and Siva Rajamanickam

Machine learning models that are trained on large corpuses of text, images, and source code are becoming increasingly common. Such models—which are either freely available or accessible for a fee—can then generate their own text, images, and source code. The unprecedented pace of development and adoption of these tools is quite different from the traditional mathematical software development life cycle. In addition, developers are creating large language models (LLMs) for text summarization as well as caption and prompt generation. LLMs are fine-tuned on source code, such as in OpenAI Codex, which yields models that can interactively generate code with minimal prompting. For example, a prompt like "`sort an array`" produces code one line at a time that a programmer can then either choose to accept or use to generate a match for an entire sort routine.

https://sinews.siam.org/Details-Page/ethical-concerns-of-code-generation-through-artificial-intelligence

The BigCode approach: data inspection a



Opt-out request for nuprl #54

Closed arjunguha opened this issue on Nov 7, 2023 · 2 comments

arjunguha commented on Nov 7, 2023 · Member

I request that the following data is removed from The Stack and StackOverflow:

- nuprl/TypeWeaver

Note: If you don't want all resources to be included just remove the elements from the list above. If you would like to exclude all repositories and resources just add a single element "all" to the list.

arjunguha commented on Nov 7, 2023 · Member · Author

This is a benchmark, and was used in the StarCoder paper. So best not to train on it. :)

lvwerra closed this as completed 2 days ago

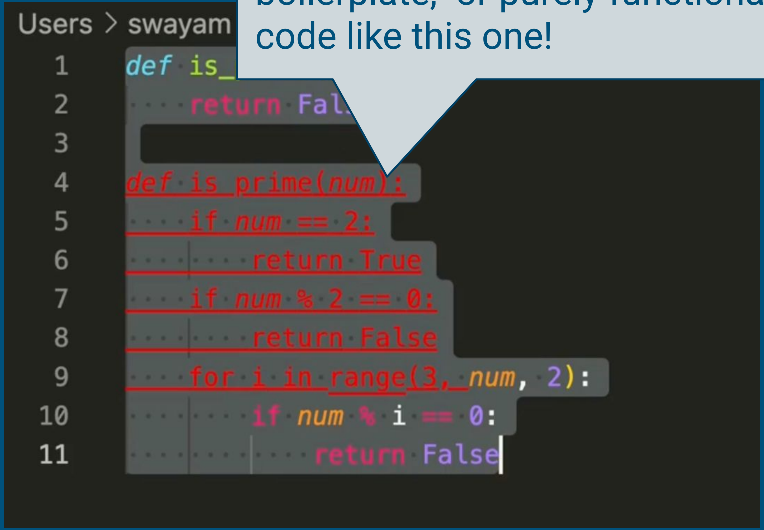
lvwerra commented 2 days ago · Member

Your opt-out request has been processed and your data was removed in version v2.0.1 of The Stack and all future versions. Also your data was not used for the training of StarCoder2.

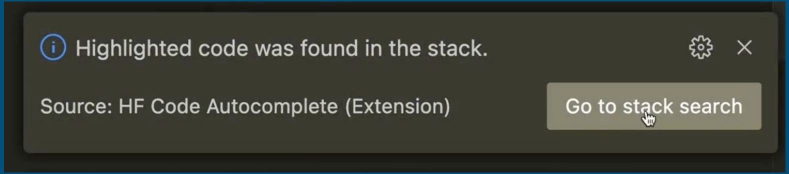
[PROCESSED]

Members

Beware of false positives: *not everything is copyrightable*, e.g. boilerplate, or purely functional code like this one!



```
def is_
    return Fal

def is_prime(num):
    if num == 2:
        return True
    if num % 2 == 0:
        return False
    for i in range(3, num, 2):
        if num % i == 0:
            return False
```



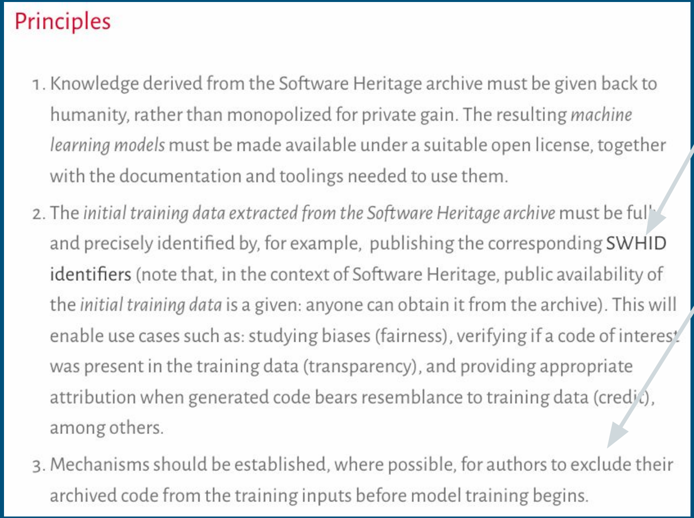
Highlighted code was found in the stack.

Source: HF Code Autocomplete (Extension)

https://marketplace.visualstudio.com/items?itemName=HuggingFace.huggingface-vscode

This slide is courtesy of Leandro Von Werra and Harm de Vries

Lessons learned



Principles

1. Knowledge derived from the Software Heritage archive must be given back to humanity, rather than monopolized for private gain. The resulting *machine learning models* must be made available under a suitable open license, together with the documentation and toolings needed to use them.
2. The *initial training data extracted from the Software Heritage archive* must be fully and precisely identified by, for example, publishing the corresponding SWHID identifiers (note that, in the context of Software Heritage, public availability of the *initial training data* is a given: anyone can obtain it from the archive). This will enable use cases such as: studying biases (fairness), verifying if a code of interest was present in the training data (transparency), and providing appropriate attribution when generated code bears resemblance to training data (credit), among others.
3. Mechanisms should be established, where possible, for authors to exclude their archived code from the training inputs before model training begins.

Transparency is easy: SWHID (undergoing ISO standardisation) and Software Heritage
N.B. : may be mandated by regulations!

Opt out is complex: who is *the real right owner*? (similar issues to license compliance)

+

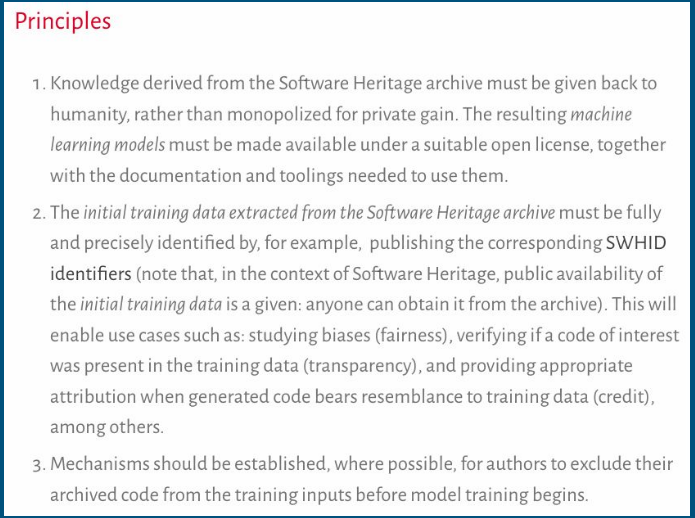
- **Building the training set** is complex: e.g. includes **license compliance** alike work **at massive scale**
- Generating **attribution information** on model output **is more complex** than license compliance

We need **a global mutualised effort** to ensure fully open models will succeed!

Spoiler alert: Software Heritage is engaging in this effort.

Ok for StarCoder2 and OlmO, …
what about all the others?

Could we ask for more?



Principles

1. Knowledge derived from the Software Heritage archive must be given back to humanity, rather than monopolized for private gain. The resulting *machine learning models* must be made available under a suitable open license, together with the documentation and toolings needed to use them.
2. The *initial training data extracted from the Software Heritage archive* must be fully and precisely identified by, for example, publishing the corresponding SWHID identifiers (note that, in the context of Software Heritage, public availability of the *initial training data* is a given: anyone can obtain it from the archive). This will enable use cases such as: studying biases (fairness), verifying if a code of interest was present in the training data (transparency), and providing appropriate attribution when generated code bears resemblance to training data (credit), among others.
3. Mechanisms should be established, where possible, for authors to exclude their archived code from the training inputs before model training begins.

Is this in the interest of FLOSS?

Pre GenAI principles

- the (source code of the) **software** used for the processing
- the trained machine **model**(s)
- the **data** used in the training
- and all information necessary to perform **independent experiments** using all the above

Other ideas…

Limit use to specific licences?

Tagging / attribution on gen code?

Lessons learned, cont'd

Sounds familiar?

Fully open models… **are not for free**

Pros *Full transparency on all stages of model development*

- External inputs to the project
- Scientific reproducibility

Cons

Resource overhead

- **Legal risks** of data transparency
- Giving away **competitive edge**
- **Code and data maintenance**

The ***playing field today is tilted in favour of closed/open weights models***….

We need **a global effort** to make open models succeed !

A plurality of actors we can engage with

European Commission

European AI Office

and more …

GPAI / THE GLOBAL PARTNERSHIP ON ARTIFICIAL INTELLIGENCE

The AI Alliance

United Nations | AI Advisory Body

unesco

Global AI Ethics and Governance Observatory