First Year Review

Chris Cummins

machine learning for compilers

$$y = f(x)$$

Optimisations

CPU or GPU

Workgroup size

Cflags

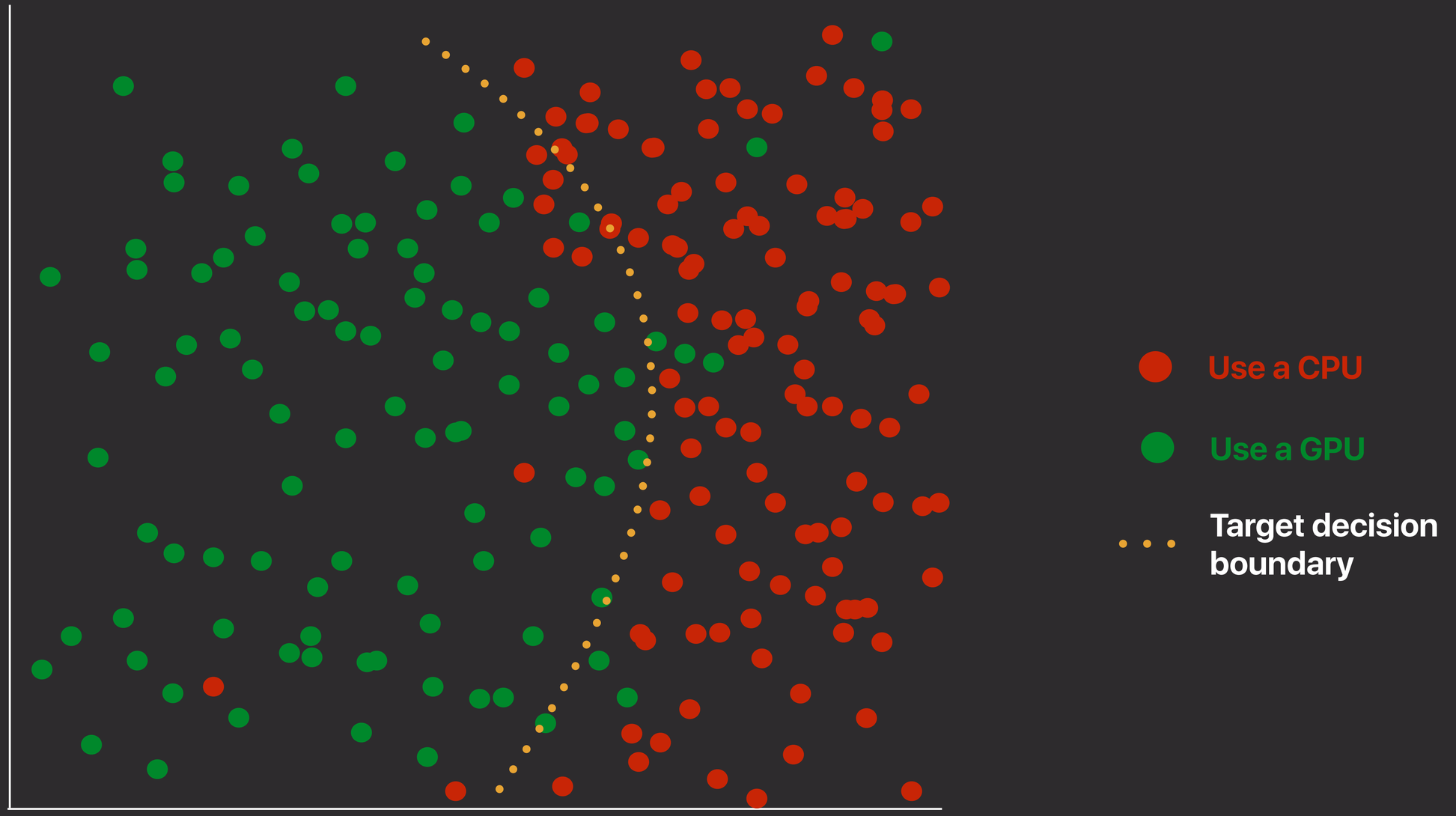
Features

instructions

Arithmetic density

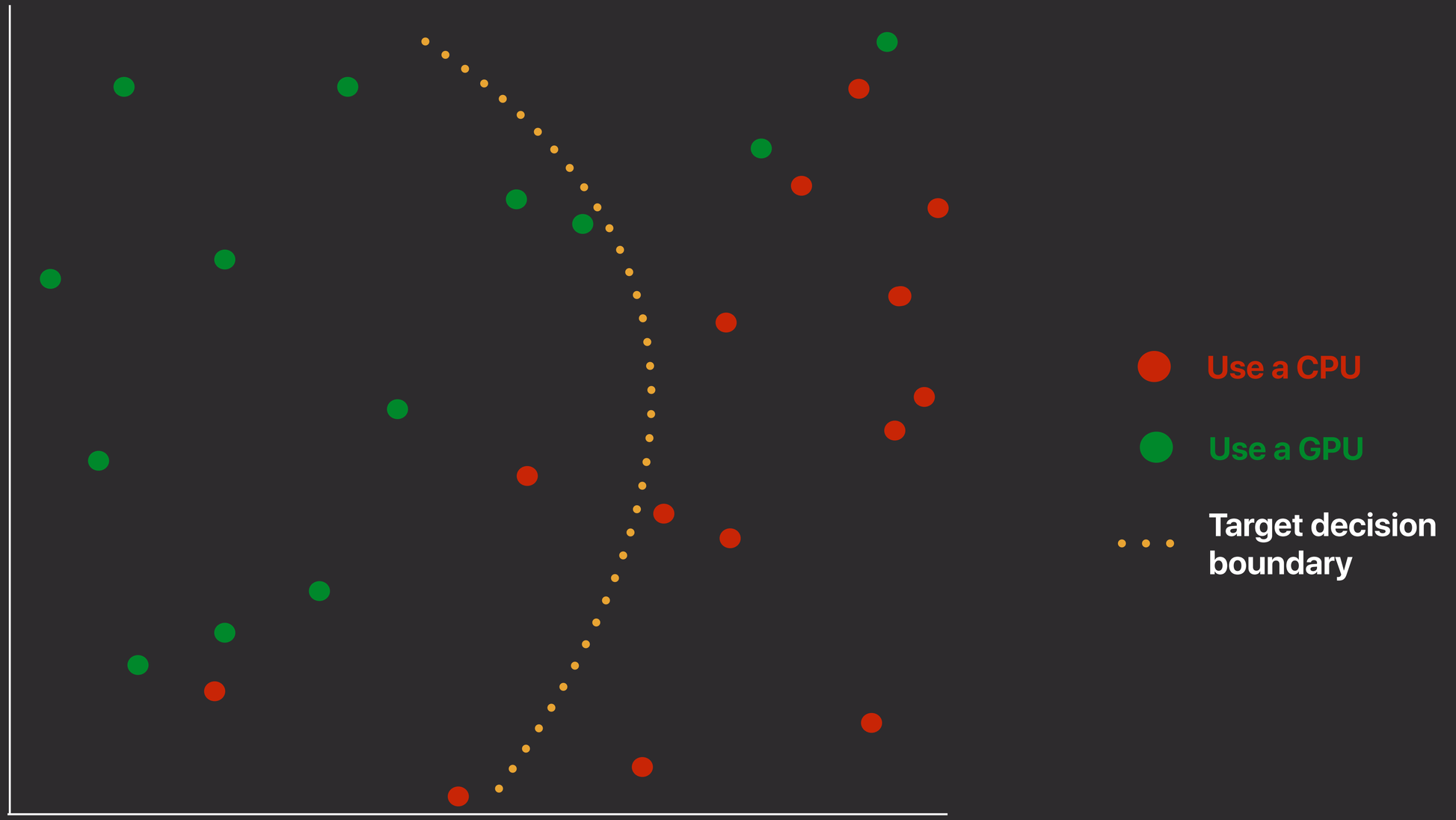
Dataset size

machine learning for compilers



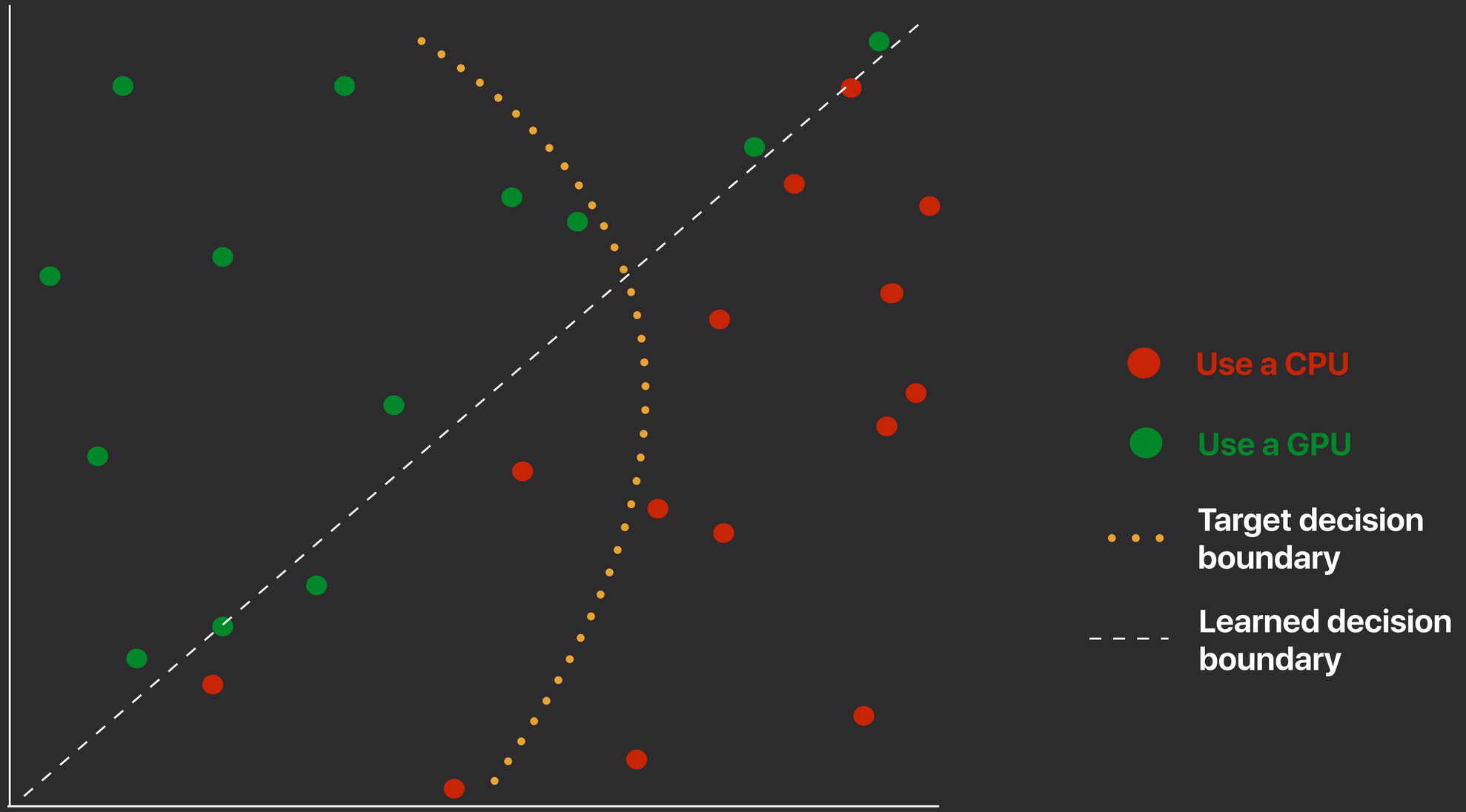
The idea.

machine learning for compilers



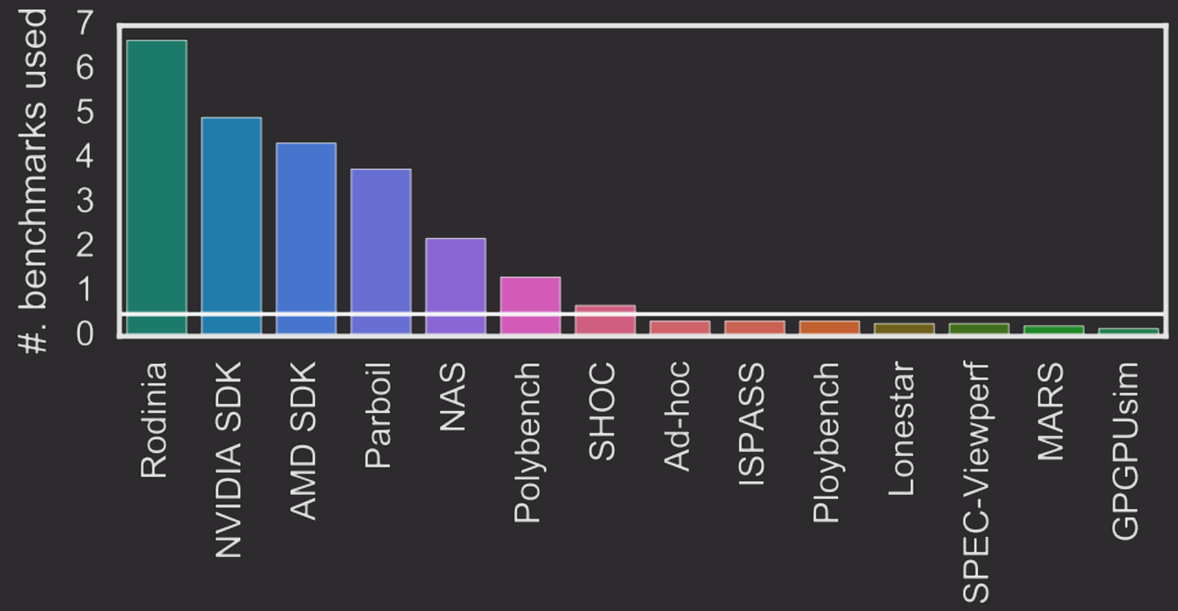
The reality.

machine learning for compilers

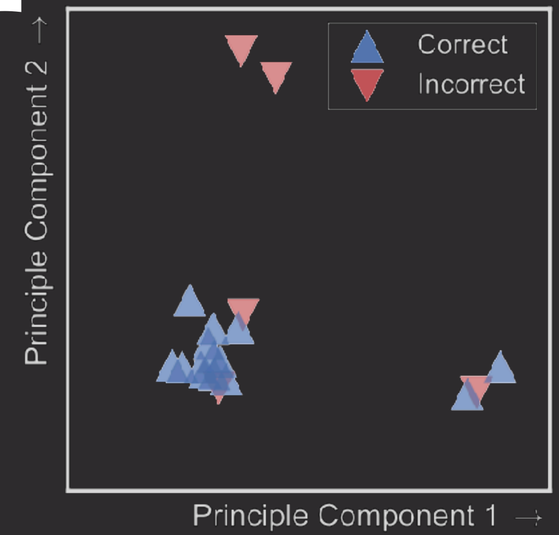


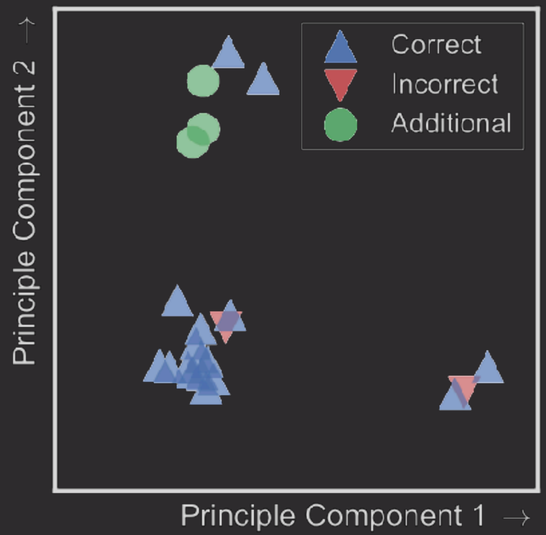
The reality.

1. there aren't enough benchmarks



2. more benchmarks = better models



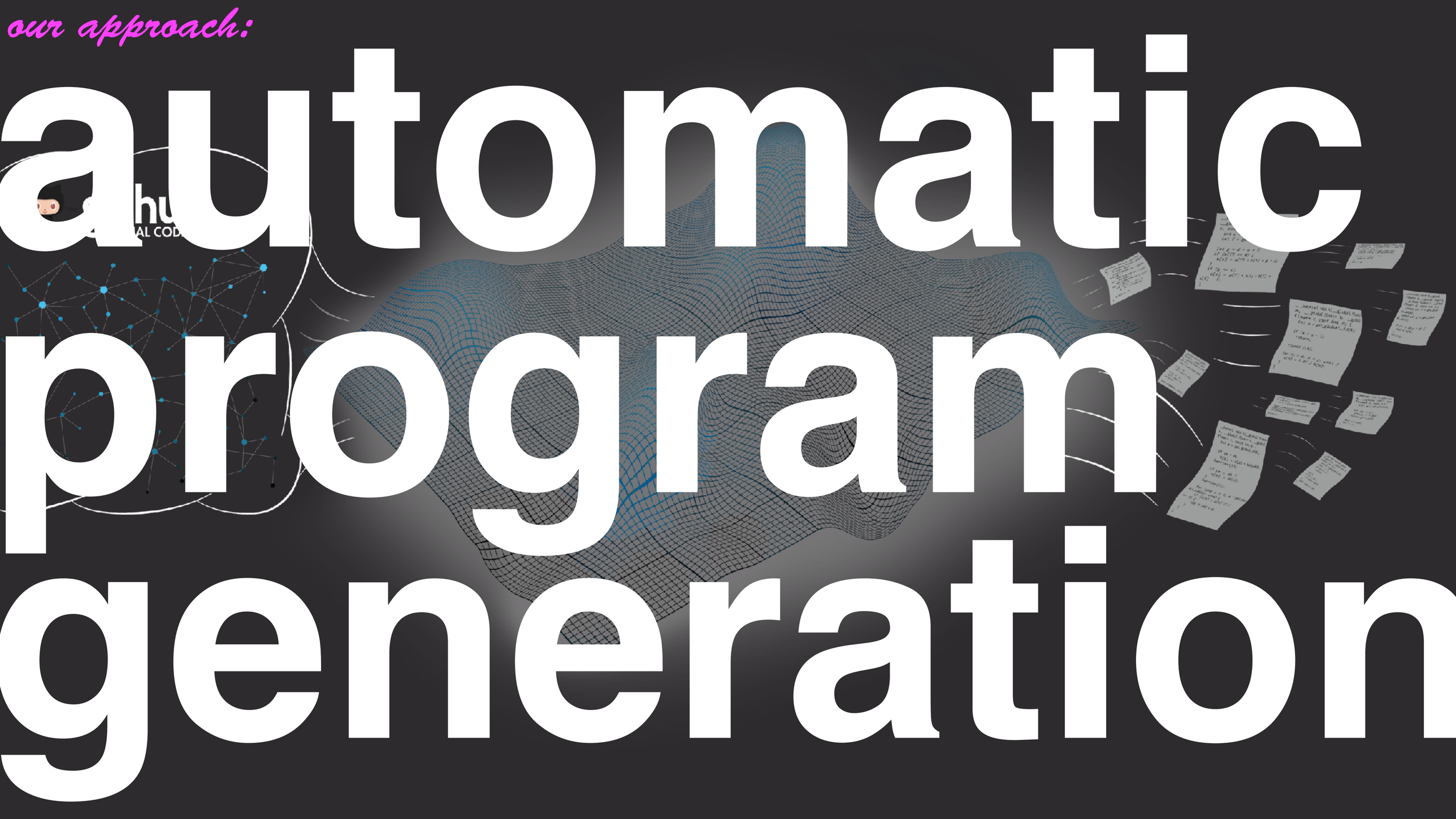


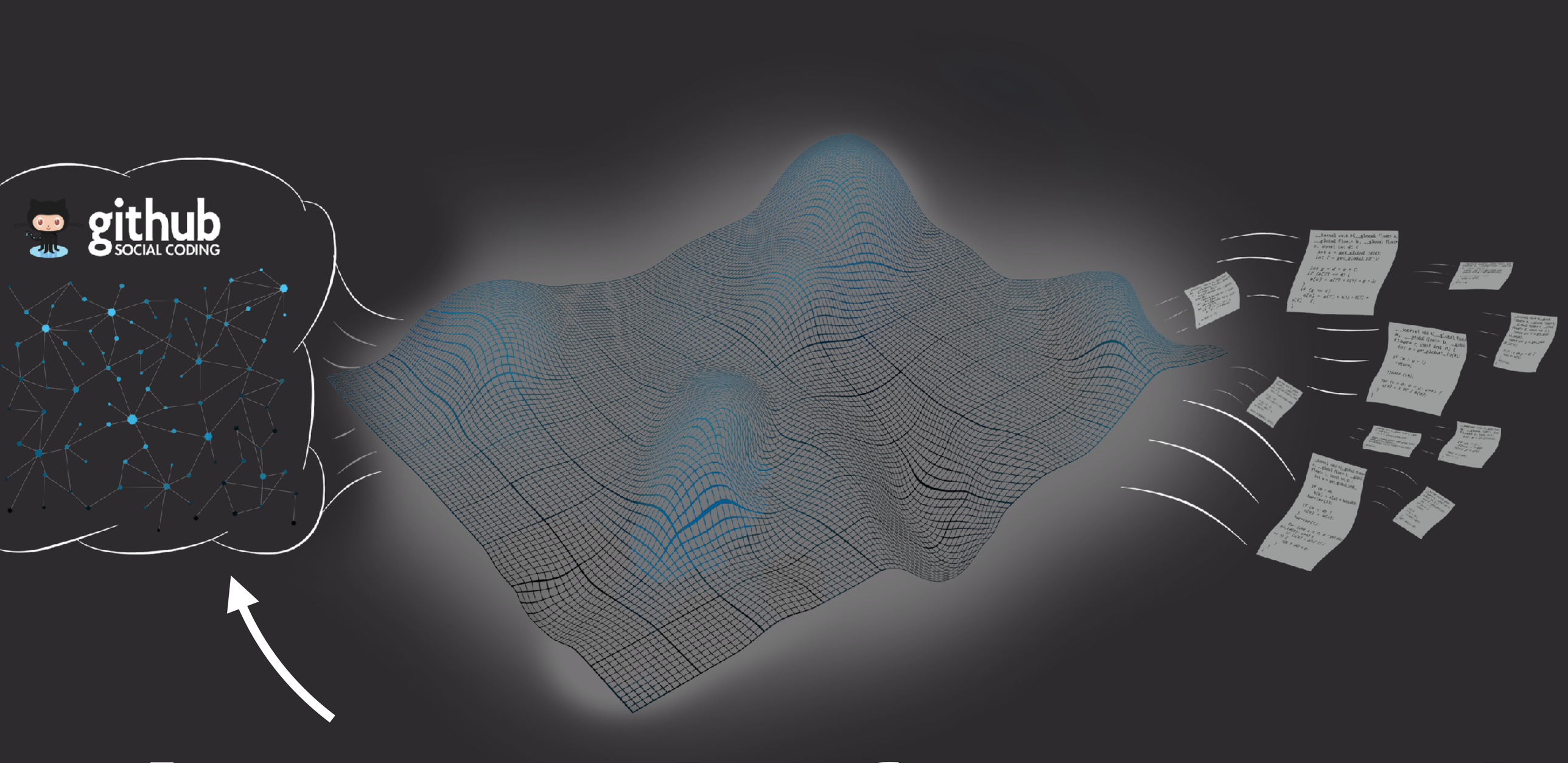
3. no adequate solution

```
1  begin genesis
2
3    geninclude gen_c.glb
4
5    global
6      distribution epochdist = {1:10}
7      distribution numvardist = {8;16;32}
8      distribution compdist = {1:20}
9      distribution coefdist = {0:7}
10     distribution offsdist = {0:15}
11   end
12
13   program
14     value numepochs sample epochdist
15     value numvars sample numvardist
16     varlist temp[${numvars}]
17   end
18
19   feature computation
20     variable dest,src1,src2,src3 from temp
21     ${dest} = ${src1} * ${src2} + ${src3};
22   end
23
24   feature read_access
25     variable dest from temp
26     value coef1,coef2 sample coefdist
27     value offs sample offsdist
28     ${dest} = arr_in[${coef1}*it0 + ${coef2}*
           it1 +${offs}];
29   end
30
31   feature write_access
32     variable src from temp
33     value offs sample offsdist
34     arr_out[inner_tc*it0 + it1 +${offs}] = ${
           src};
35   end
36
37   feature epoch (epoch_type)
38     value numcomps sample compdist
39     genloop i:1:${numcomps}
40       ${computation}
41     end
42     genif ${epoch_type} eq "read"
43       ${read_access}
44     genelse
45       ${write_access}
46     end
47   end
48
49   feature epochs
50     genloop i:1:${numepochs}
51       ${epoch("read")}
52     end
53     ${epoch("write")}
54   end
55
56   generate 1000
57 end genesis
```

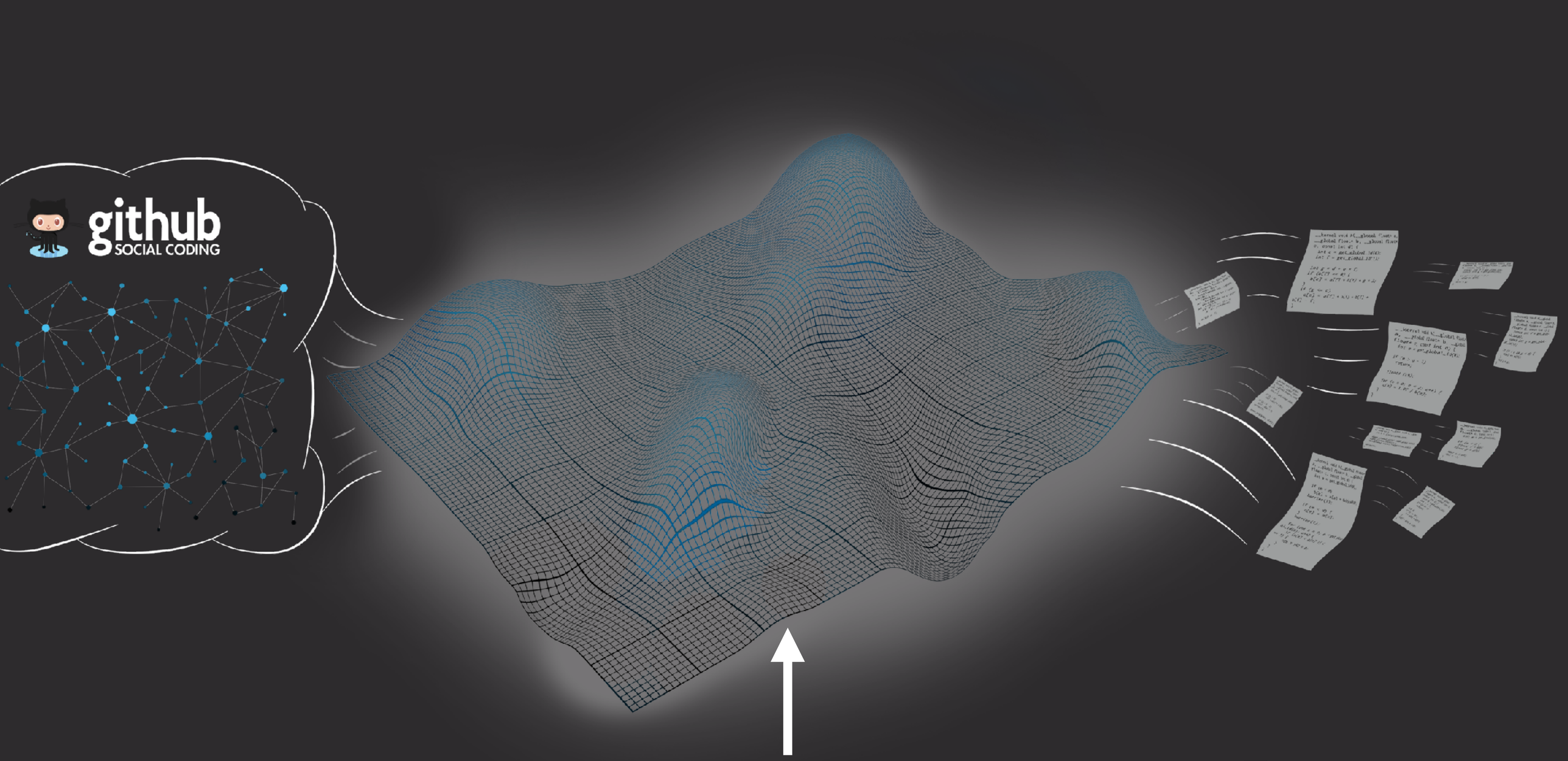
our approach:

automatic program generation

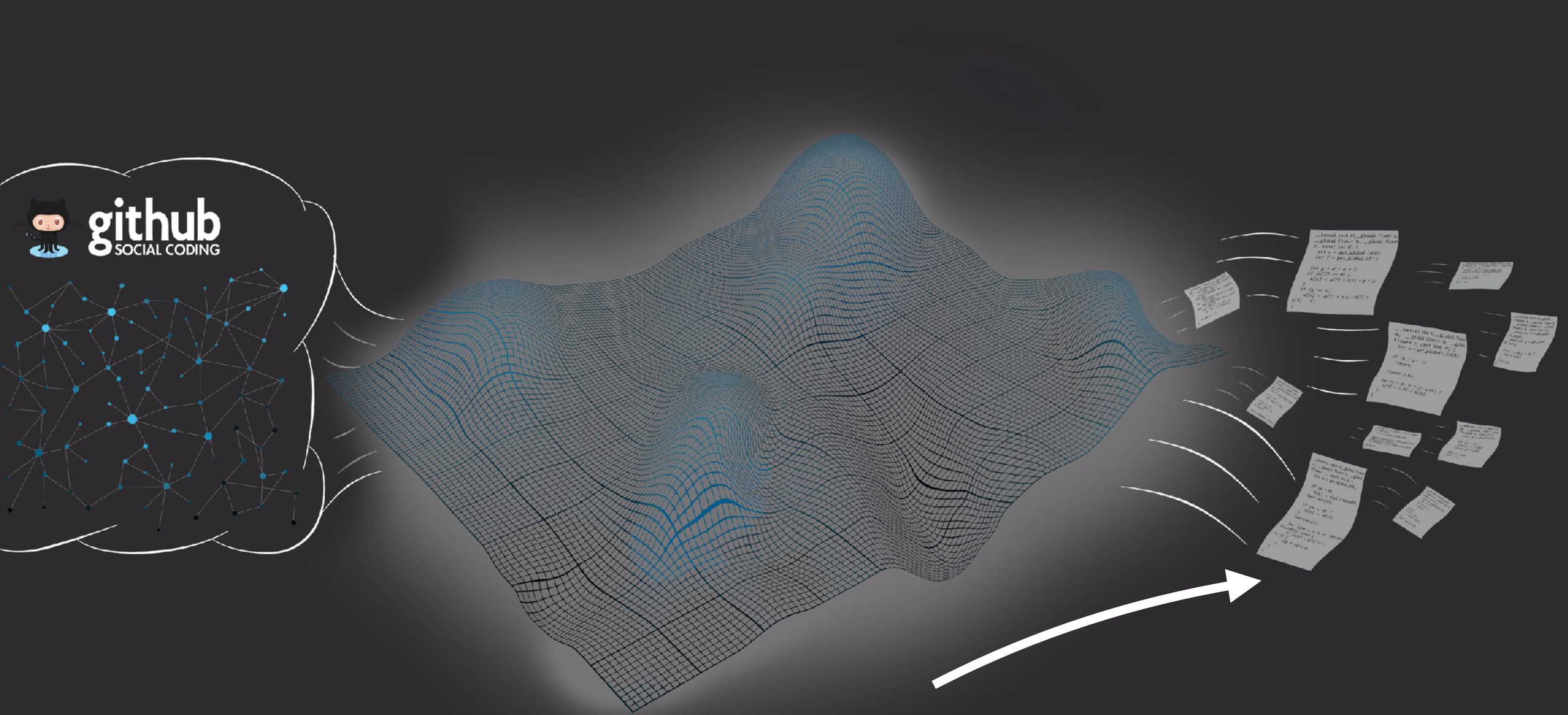




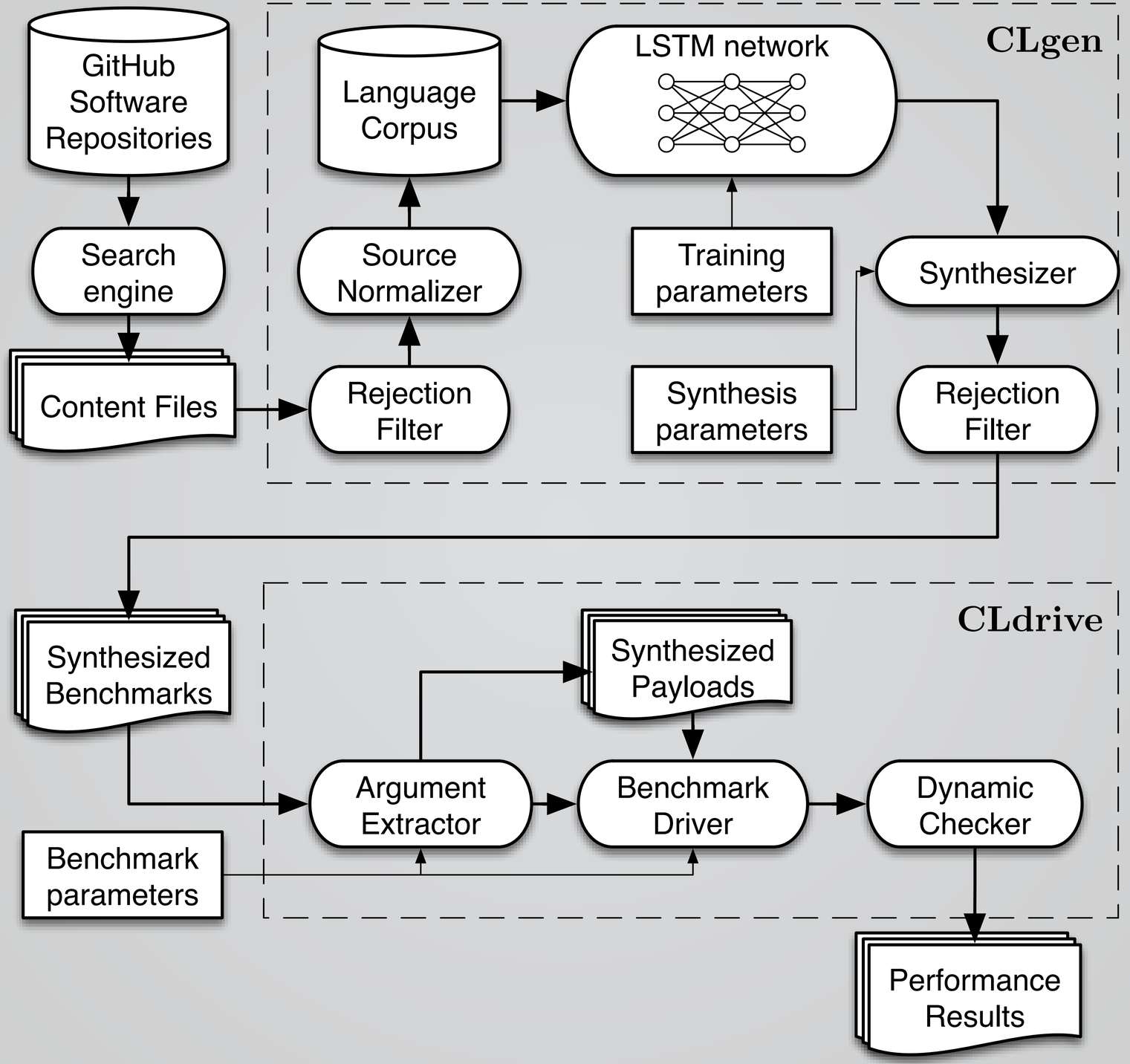
mine code from web

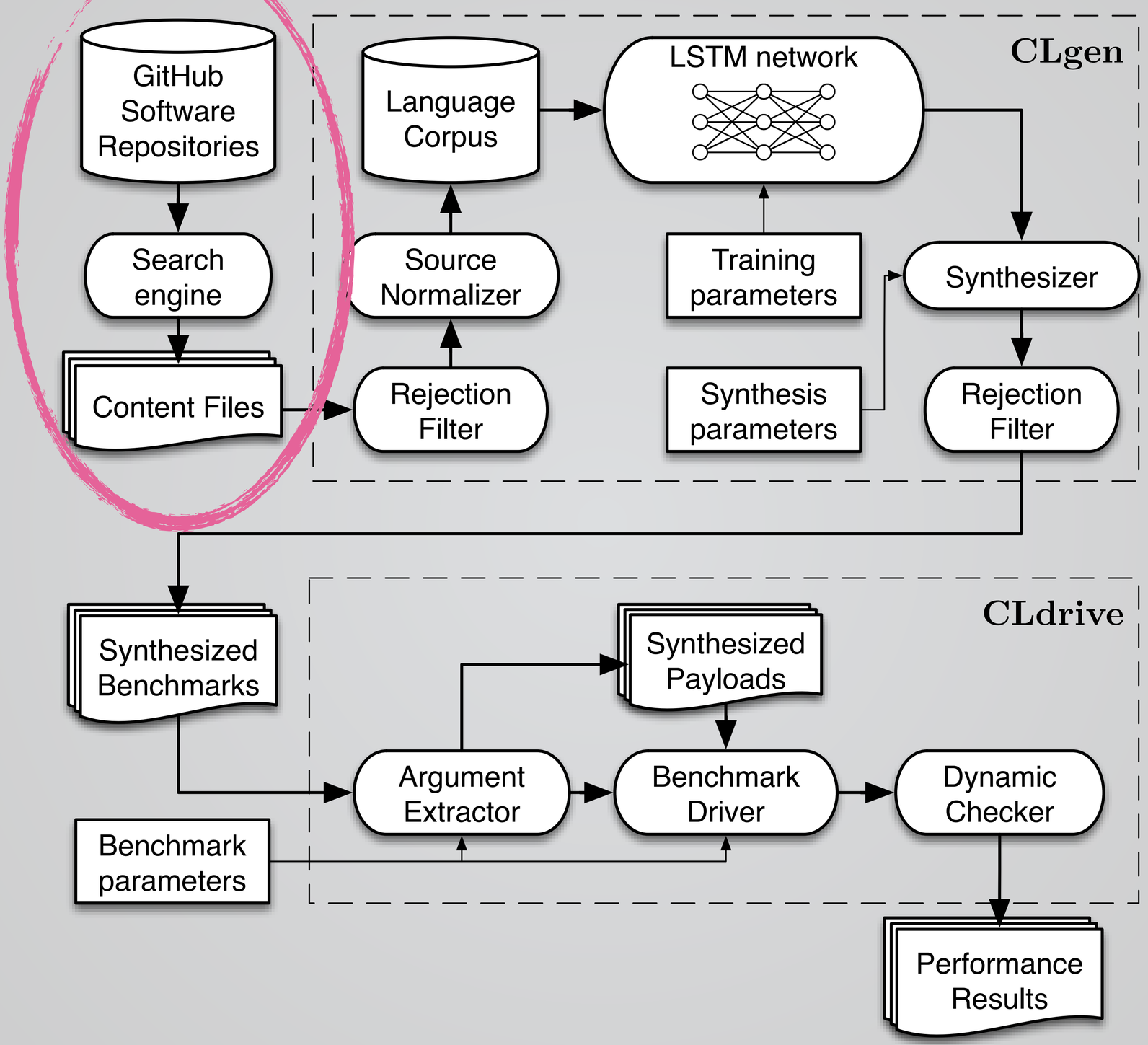


model source distr.



sample and reject


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Language Corpus
Source Normalizer
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Source Normalizer
Language Corpus
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results

Infer the common usage of a PL from samples.

Huge repository of public knowledge: 

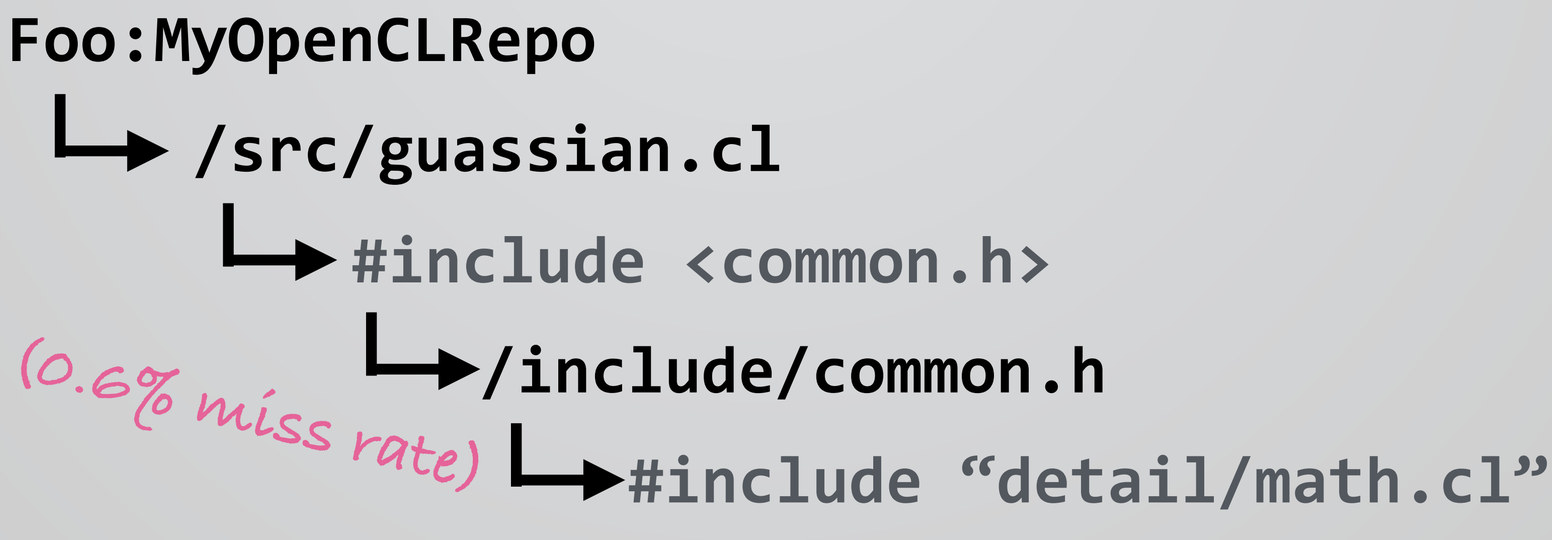
And they have an API :-)

```
$ curl https://api.github.com/search/repositories\?
q\=opencl\&sort\=stars\&order\=desc
{
  "total_count": 3155,
  "incomplete_results": false,
  "items": [
    {
      "id": 7296244,
      "name": "lwjgl3",
      "full_name": "LWJGL/lwjgl3",
```

OpenCL is not a first-class language.

Search repositories using loose keyword terms.
e.g. **`opencl`**, **`nvidia`**, **`gpu`**, **`cl`**, **`amd`**.

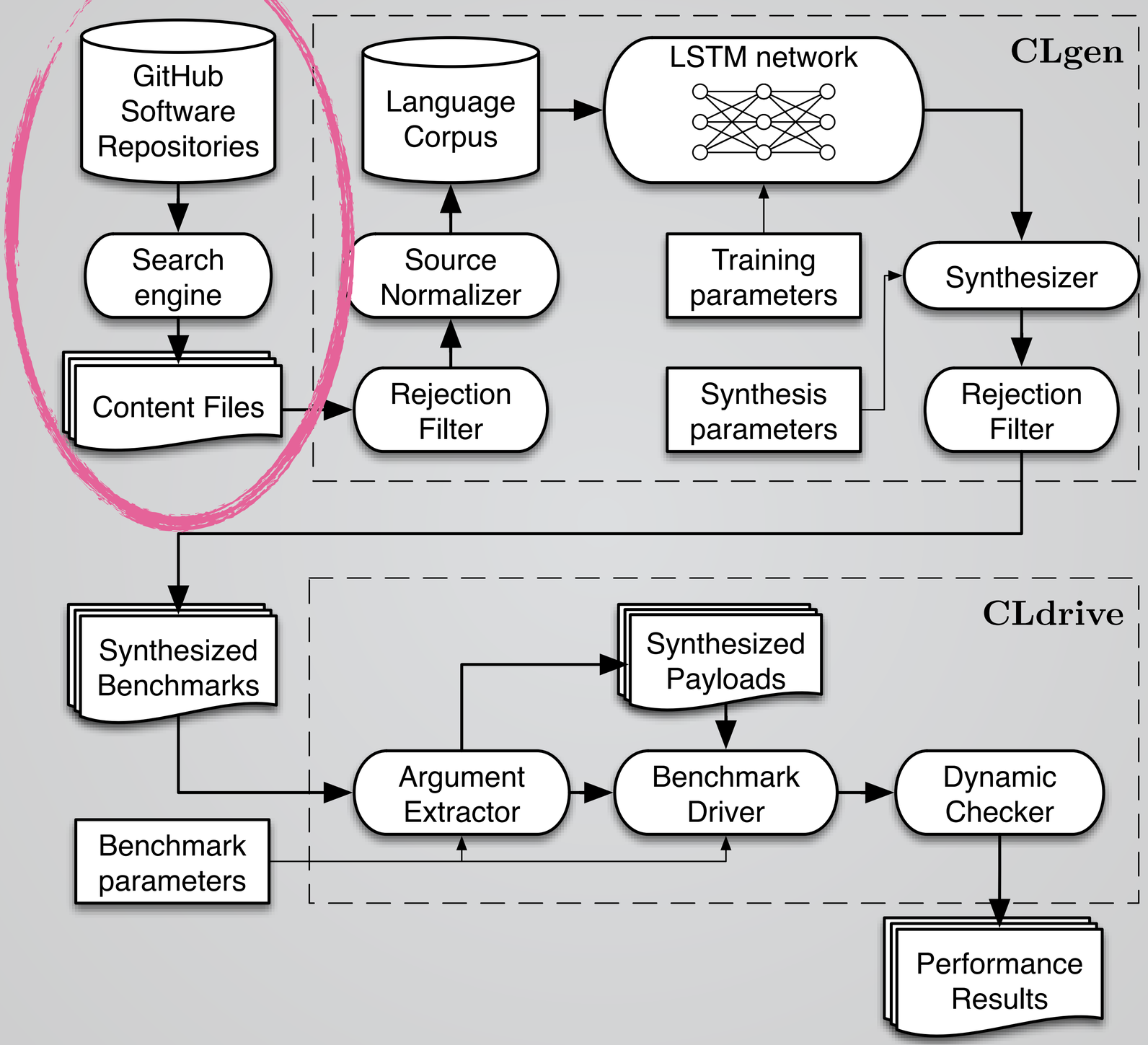
Recursively iterate over git trees to get **`.cl`** files.

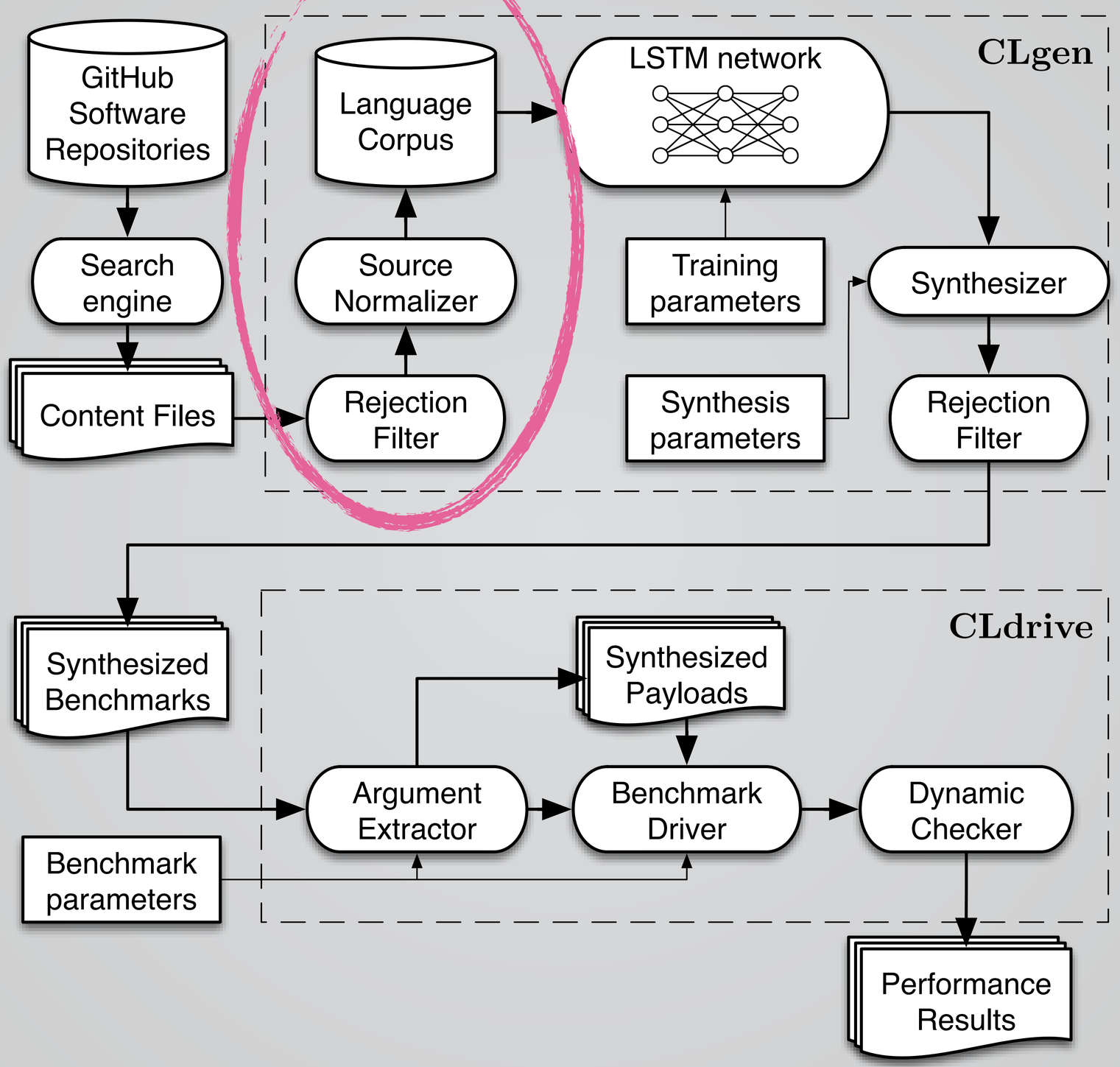


```
/* Copyright (C) 2004 Joe Bloggs <joe@bloggs.io>  */
//
//            DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
//   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
//
//  0. You just DO WHAT THE FUCK YOU WANT TO.
#define CLAMPING
#define THRESHOLD_MIN 1.0f
#define THRESHOLD_MAX 1.0f

float myclamp(float in) {
#ifdef CLAMPING
    return in > THRESHOLD_MAX ? THRESHOLD_MAX : in < THRESHOLD_MIN ? THRESHOLD_MIN : in;
#else
    return in;
#endif // CLAMPING
}

// Do something really flipping cool
__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    //
    //
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

x 8078 files

2.8 million lines


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Source Normalizer
Language Corpus
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results


CLgen
GitHub Software Repositories
Search engine
Content Files
Language Corpus
Source Normalizer
Rejection Filter
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Synthesized Payloads
Argument Extractor
Benchmark Driver
Dynamic Checker
Performance Results

```
/* Copyright (C) 2004 Joe Bloggs <joe@bloggs.io>  */
//
//            DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
//   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
//
//  0. You just DO WHAT THE FUCK YOU WANT TO.
#define CLAMPING
#define THRESHOLD_MIN 1.0f
#define THRESHOLD_MAX 1.0f

float myclamp(float in) {
#ifdef CLAMPING
    return in > THRESHOLD_MAX ? THRESHOLD_MAX : in < THRESHOLD_MIN ? THRESHOLD_MIN : in;
#else
    return in;
#endif // CLAMPING
}

// Do something really flipping cool
__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    //
    //
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

```
/* Copyright (C) 2004 Joe Bloggs <joe@bloggs.io>  */
//
//            DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
//   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
//
//  0. You just DO WHAT THE FUCK YOU WANT TO.
#define CLAMPING
#define THRESHOLD_MIN 1.0f
#define THRESHOLD_MAX 1.0f

float myclamp(float in) {
#ifdef CLAMPING
    return in > THRESHOLD_MAX ? THRESHOLD_MAX : in < THRESHOLD_MIN ? THRESHOLD_MIN : in;
#else
    return in;
#endif // CLAMPING
}

// Do something really flipping cool
__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    //
    //
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

Is this real, valid OpenCL?
Can we minimise non-functional variance?

```
/* Copyright (C) 2004 Joe Bloggs <joe@bloggs.io>  */
//
//            DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
//   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
//
//  0. You just DO WHAT THE FUCK YOU WANT TO.
#define CLAMPING
#define THRESHOLD_MIN 1.0f
#define THRESHOLD_MAX 1.0f

float myclamp(float in) {
#ifdef CLAMPING
    return in > THRESHOLD_MAX ? THRESHOLD_MAX : in < THRESHOLD_MIN ? THRESHOLD_MIN : in;
#else
    return in;
#endif // CLAMPING
}

// Do something really flipping cool
__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    //
    //
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

Strip comments

Is this real, valid OpenCL?

Can we minimise non-functional variance?

```
/* Copyright (C) 2004 Joe Bloggs <joe@bloggs.io>  */
//
//            DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
//   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
//
//  0. You just DO WHAT THE FUCK YOU WANT TO.
#define CLAMPING
#define THRESHOLD_MIN 1.0f
#define THRESHOLD_MAX 1.0f

float myclamp(float in) {
#ifdef CLAMPING
    return in > THRESHOLD_MAX ? THRESHOLD_MAX : in < THRESHOLD_MIN ? THRESHOLD_MIN : in;
#else
    return in;
#endif // CLAMPING
}

// Do something really flipping cool
__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    //
    //
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

Strip comments

Is this real, valid OpenCL?
Can we minimise non-functional variance?

Skip comments

```
#define CLAMPING
#define THRESHOLD_MIN 1.0f
#define THRESHOLD_MAX 1.0f

float myclamp(float in) {
#ifdef CLAMPING
    return in > THRESHOLD_MAX ? THRESHOLD_MAX : in < THRESHOLD_MIN ? THRESHOLD_MIN : in;
#else
    return in;
#endif
}

__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

~~Strip comments~~
Preprocess

```
#define CLAMPING
#define THRESHOLD_MIN 1.0f
#define THRESHOLD_MAX 1.0f

float myclamp(float in) {
#ifdef CLAMPING
    return in > THRESHOLD_MAX ? THRESHOLD_MAX : in < THRESHOLD_MIN ? THRESHOLD_MIN : in;
#else
    return in;
#endif
}

__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

Strip comments

Preprocess

```
#define CLAMPING
#define THRESHOLD_MIN 1.0f
#define THRESHOLD_MAX 1.0f

float myclamp(float in) {
#ifdef CLAMPING
    return in > THRESHOLD_MAX ? THRESHOLD_MAX : in < THRESHOLD_MIN ? THRESHOLD_MIN : in;
#else
    return in;
#endif
}

__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

~~Strip comments~~

~~Preprocess~~

```
float myclamp(float in) {
   return in > 1.0f ? 1.0f : in < 0.0f ? 0.0f : in;
}

__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

~~Strip comments~~

~~Preprocess~~

```
float myclamp(float in) {
   return in > 1.0f ? 1.0f : in < 0.0f ? 0.0f : in;
}

__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

Does it compile?

Does it contain instructions?

~~Strip comments~~

~~Preprocess~~

```
float myclamp(float in) {
   return in > 1.0f ? 1.0f : in < 0.0f ? 0.0f : in;
}

__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

~~Does it compile?~~

~~Does it contain instructions?~~

~~Strip comments~~

~~Preprocess~~

Rewrite function names

```
float myclamp(float in) {
   return in > 1.0f ? 1.0f : in < 0.0f ? 0.0f : in;
}

__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

~~Does it compile?~~

~~Does it contain instructions?~~

~~Strip comments~~

~~Preprocess~~

Rewrite function names

```
float myclamp(float in) {
   return in > 1.0f ? 1.0f : in < 0.0f ? 0.0f : in;
}

__kernel void findAllNodesMergedAabb(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = myclamp(in[id]);
    }
}
```

~~Does it compile?~~

~~Does it contain instructions?~~

~~Strip comments~~

~~Preprocess~~

~~Rewrite function names~~

```
float A(float in) {
   return in > 1.0f ? 1.0f : in < 0.0f ? 0.0f : in;
}

__kernel void B(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = A(in[id]);
    }
}
```

~~Does it compile?~~

~~Does it contain instructions?~~

```
float A(float in) {
   return in > 1.0f ? 1.0f : in < 0.0f ? 0.0f : in;
}

__kernel void B(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = A(in[id]);
    }
}
```

~~Strip comments~~

~~Preprocess~~

~~Rewrite function names~~

Rewrite variable names

~~Does it compile?~~

~~Does it contain instructions?~~

```
float A(float in) {
   return in > 1.0f ? 1.0f : in < 0.0f ? 0.0f : in;
}

__kernel void B(__global float* in, __global float* out, int num_elems)
{
    int id = get_global_id(0);
    if (id < num_elems)
    {


        out[id] = A(in[id]);
    }
}
```

~~Strip comments~~

~~Preprocess~~

~~Rewrite function names~~

Rewrite variable names

~~Does it compile?~~

~~Does it contain instructions?~~

```
float A(float a) {
   return a > 1.0f ? 1.0f : a < 0.0f ? 0.0f : a;
}

__kernel void B(__global float* a, __global float* b, int c)
{
    int d = get_global_id(0);
    if (d < c)
    {


        b[d] = A(a[d]);
    }
}
```

~~Strip comments~~

~~Preprocess~~

~~Rewrite function names~~

~~Rewrite variable names~~

~~Does it compile?~~

~~Does it contain instructions?~~

```
float A(float a) {
   return a > 1.0f ? 1.0f : a < 0.0f ? 0.0f : a;
}

__kernel void B(__global float* a, __global float* b, int c)
{
    int d = get_global_id(0);
    if (d < c)
    {


        b[d] = A(a[d]);
    }
}
```

~~Strip comments~~

~~Preprocess~~

~~Rewrite function names~~

~~Rewrite variable names~~

Enforce code style

~~Does it compile?~~

~~Does it contain instructions?~~

```
float A(float a) {
   return a > 1.0f ? 1.0f : a < 0.0f ? 0.0f : a;
}

__kernel void B(__global float* a, __global float* b, int c)
{
    int d = get_global_id(0);
    if (d < c)
    {

        b[d] = A(a[d]);
    }
}
```

~~Strip comments~~

~~Preprocess~~

~~Rewrite function names~~

~~Rewrite variable names~~

Enforce code style

~~Does it compile?~~

~~Does it contain instructions?~~

```
float A(float a) {
  return a > 1.0f ? 1.0f : in < 0.0f ? 0.0f : a;
}

__kernel void B(__global float* a, __global float* b, int c) {
  int d = get_global_id(0);
  if (d < c) {
    b[d] = A(a[d]);
  }
}
```

~~Strip comments~~

~~Preprocess~~

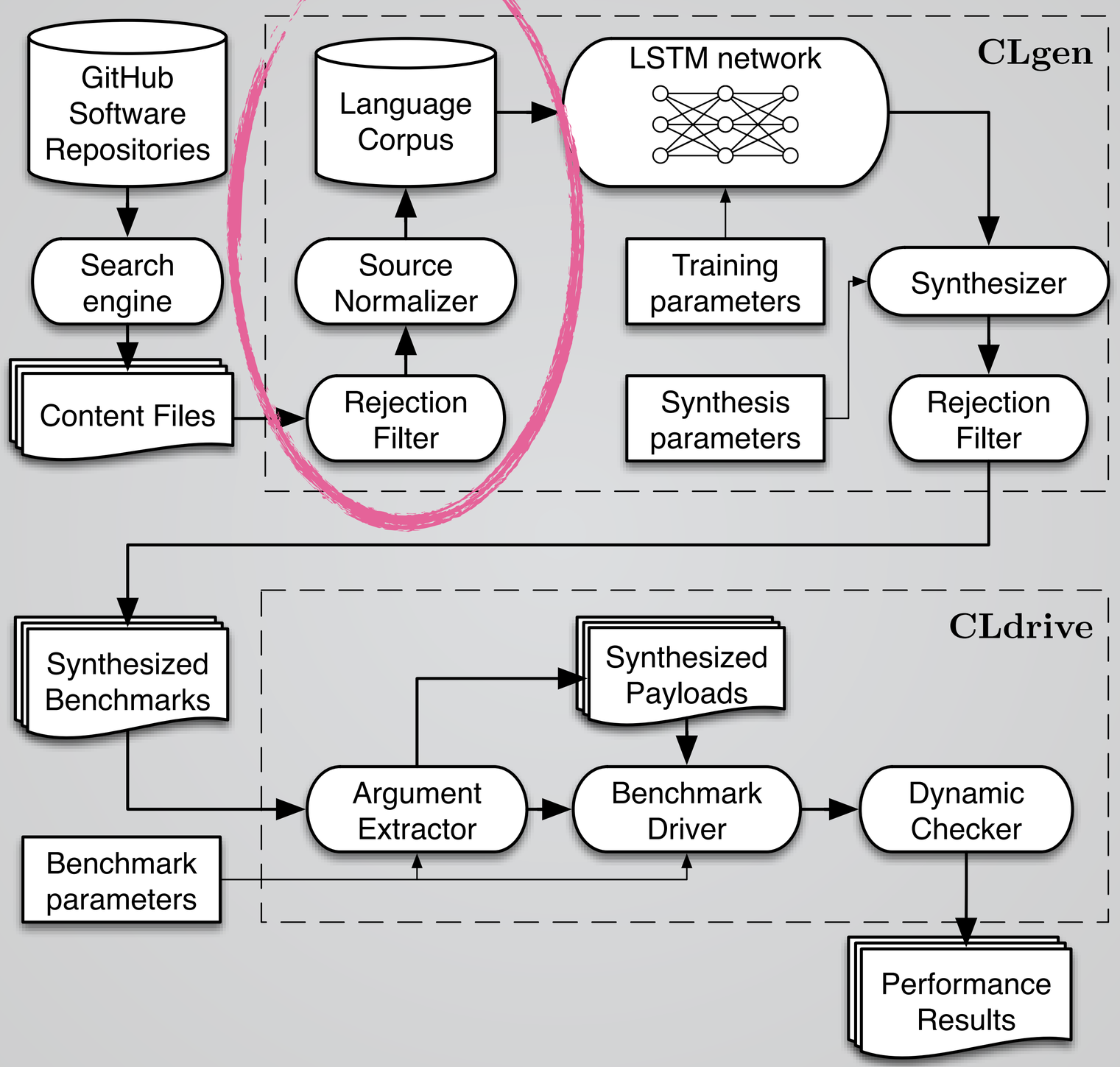
~~Rewrite function names~~

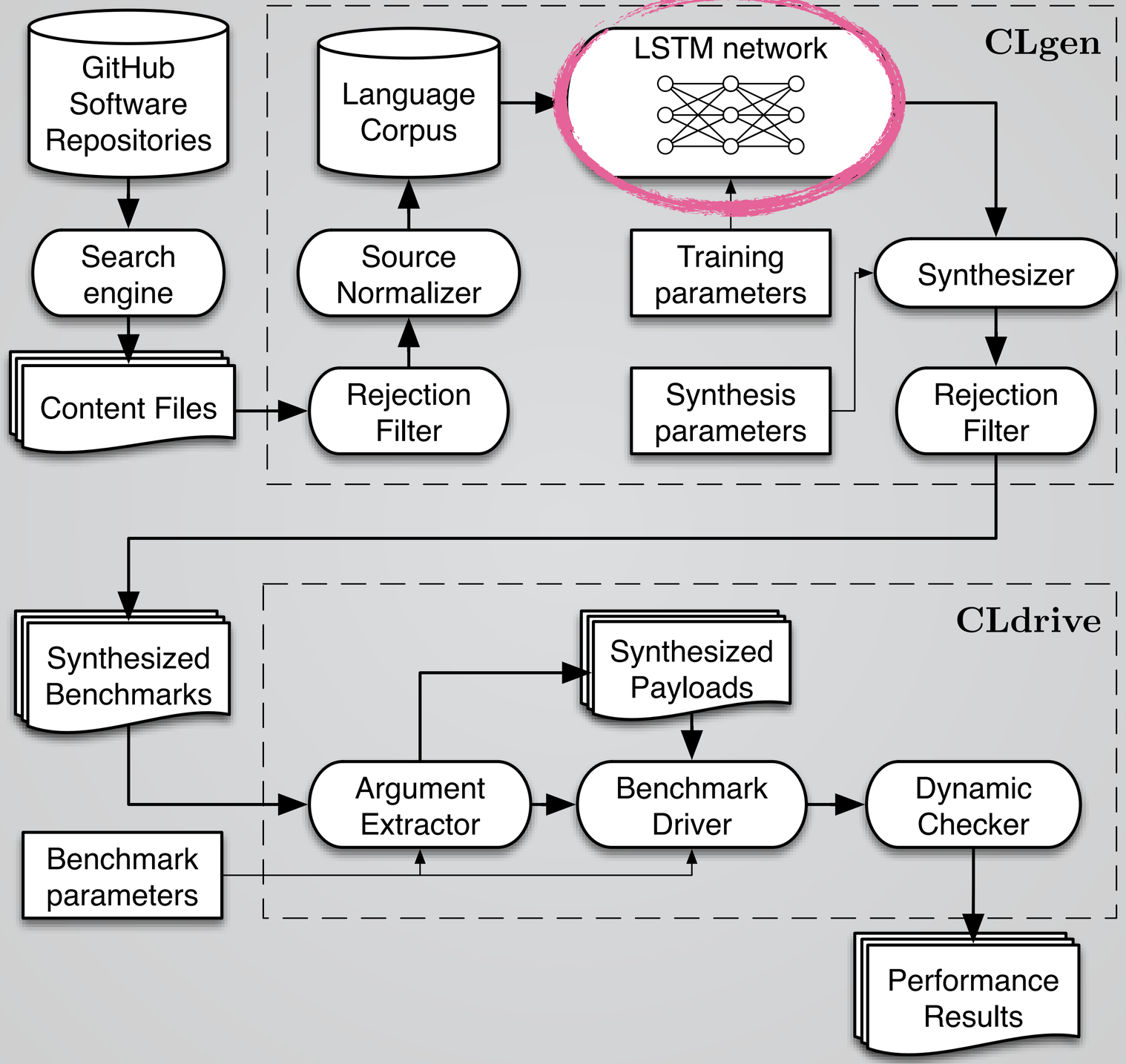
~~Rewrite variable names~~

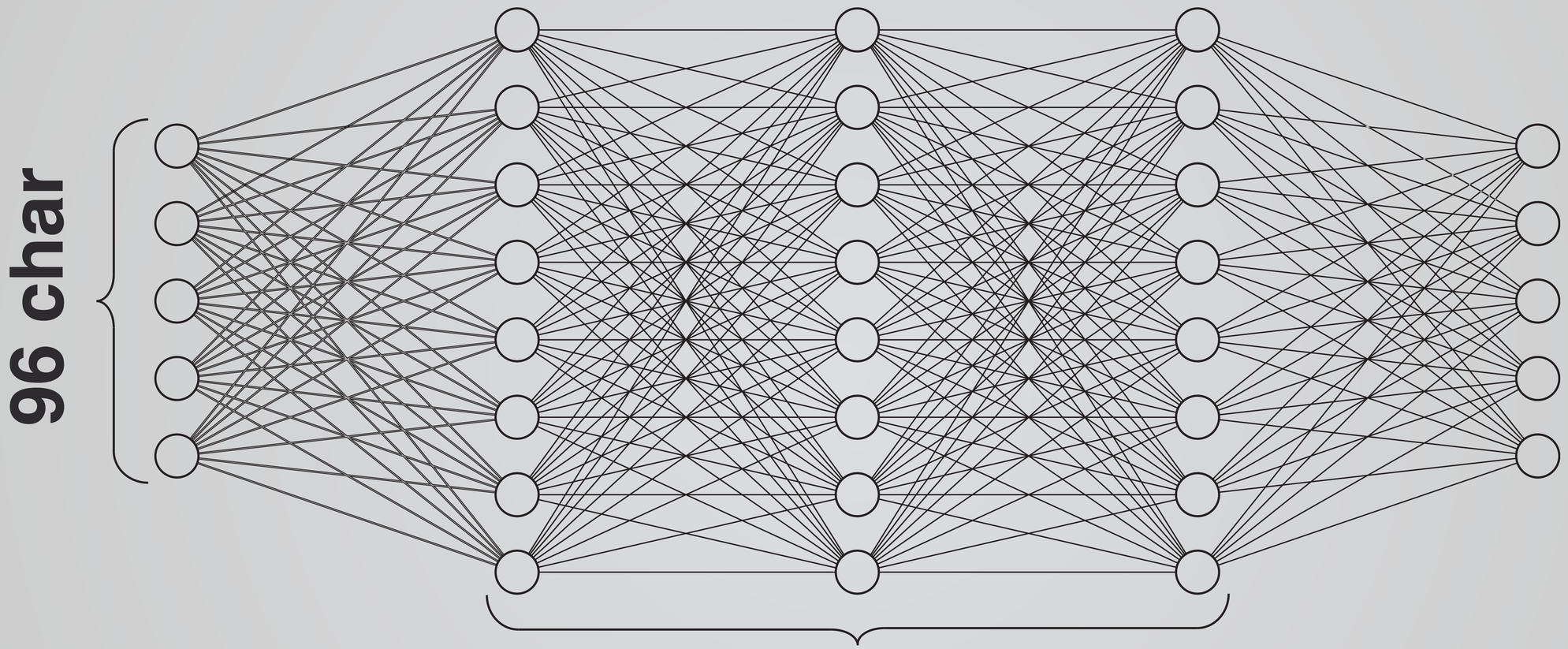
~~Enforce code style~~

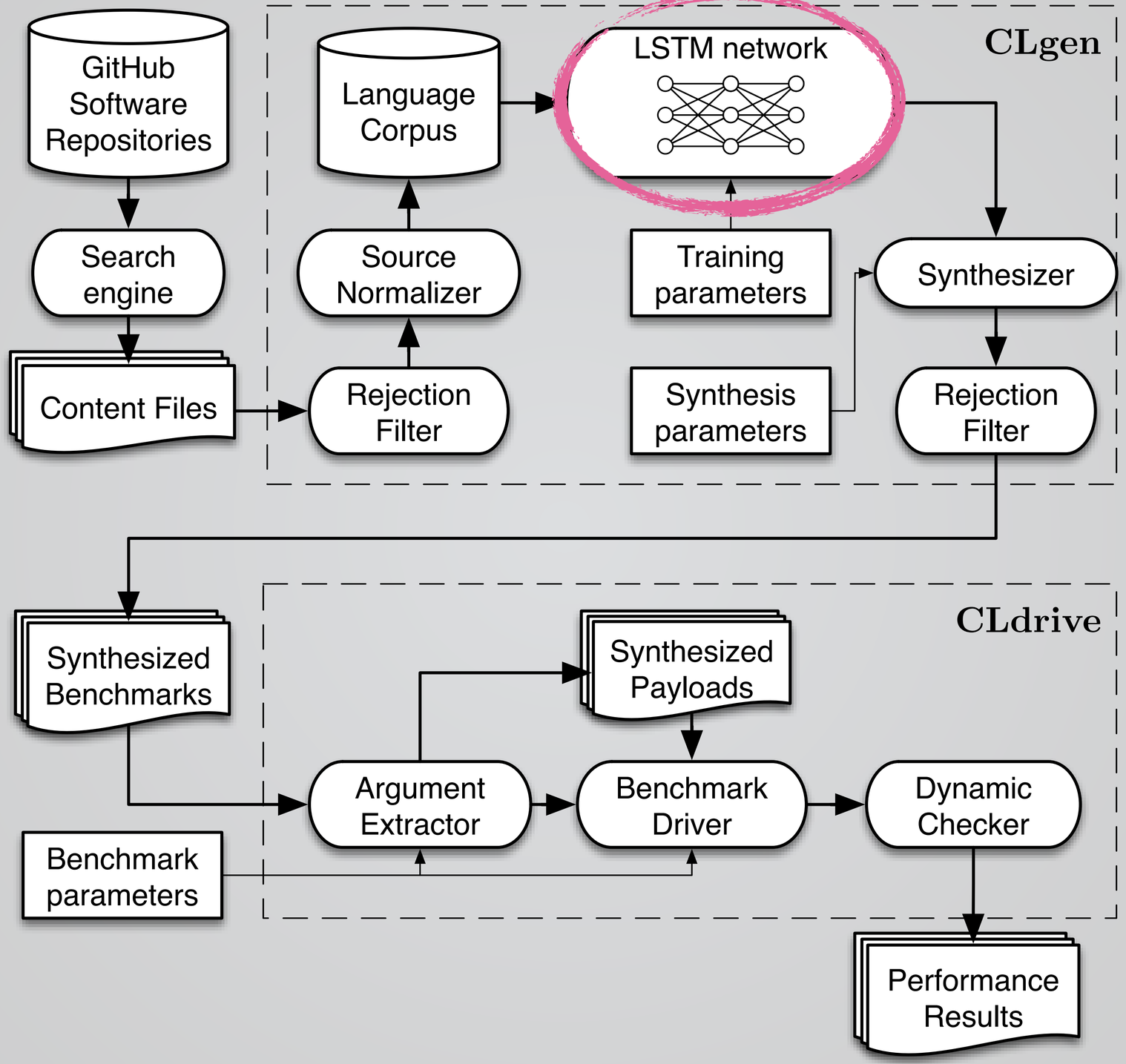
~~Does it compile?~~

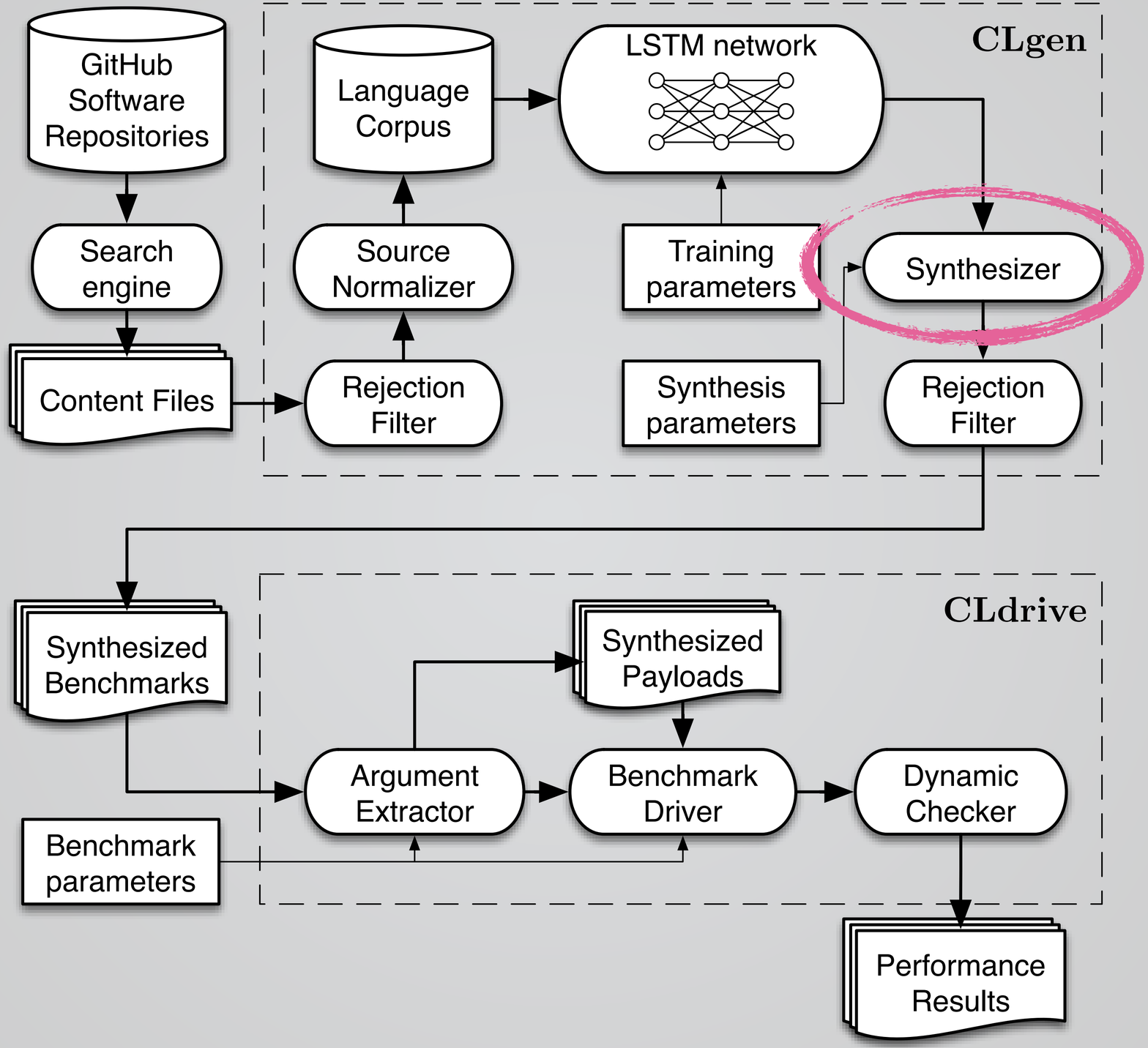
~~Does it contain instructions?~~


CLgen
GitHub Software Repositories
Search engine
Content Files
Language Corpus
Source Normalizer
Rejection Filter
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Synthesized Payloads
Argument Extractor
Benchmark Driver
Dynamic Checker
Performance Results


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Source Normalizer
Language Corpus
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results

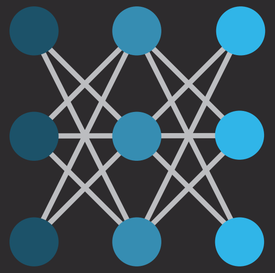

96 char
2048 node, 3 layer


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Source Normalizer
Language Corpus
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Language Corpus
Source Normalizer
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results

kernel synthesis

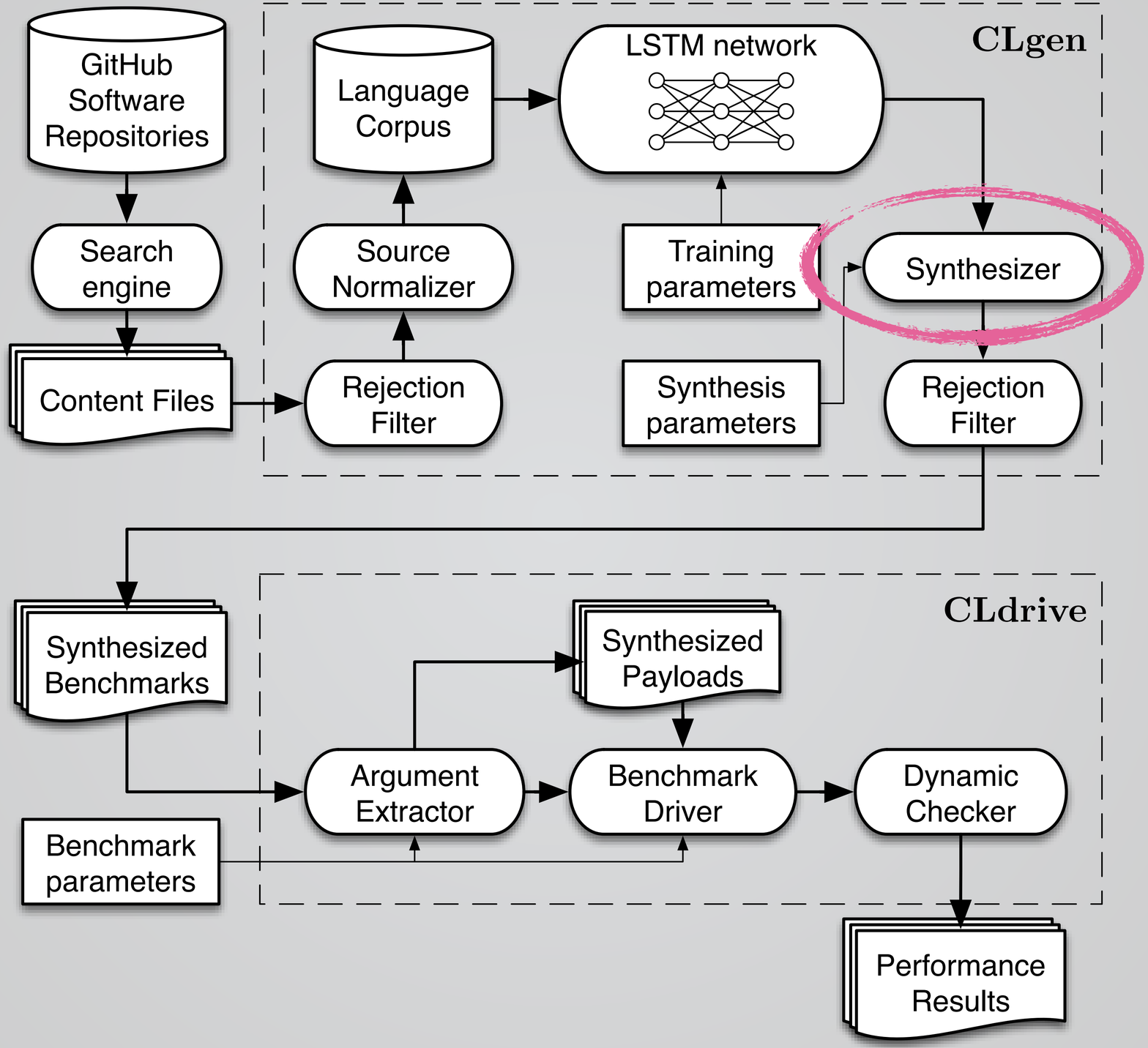
```
S = ‘__kernel void A(__global float* a) {’
depth = 1
while depth > 0:
  c = predict_next_character(S)
  if c == ‘{’:
    depth += 1
  if c == ‘}’:
    depth -= 1
  S += c

return S
```

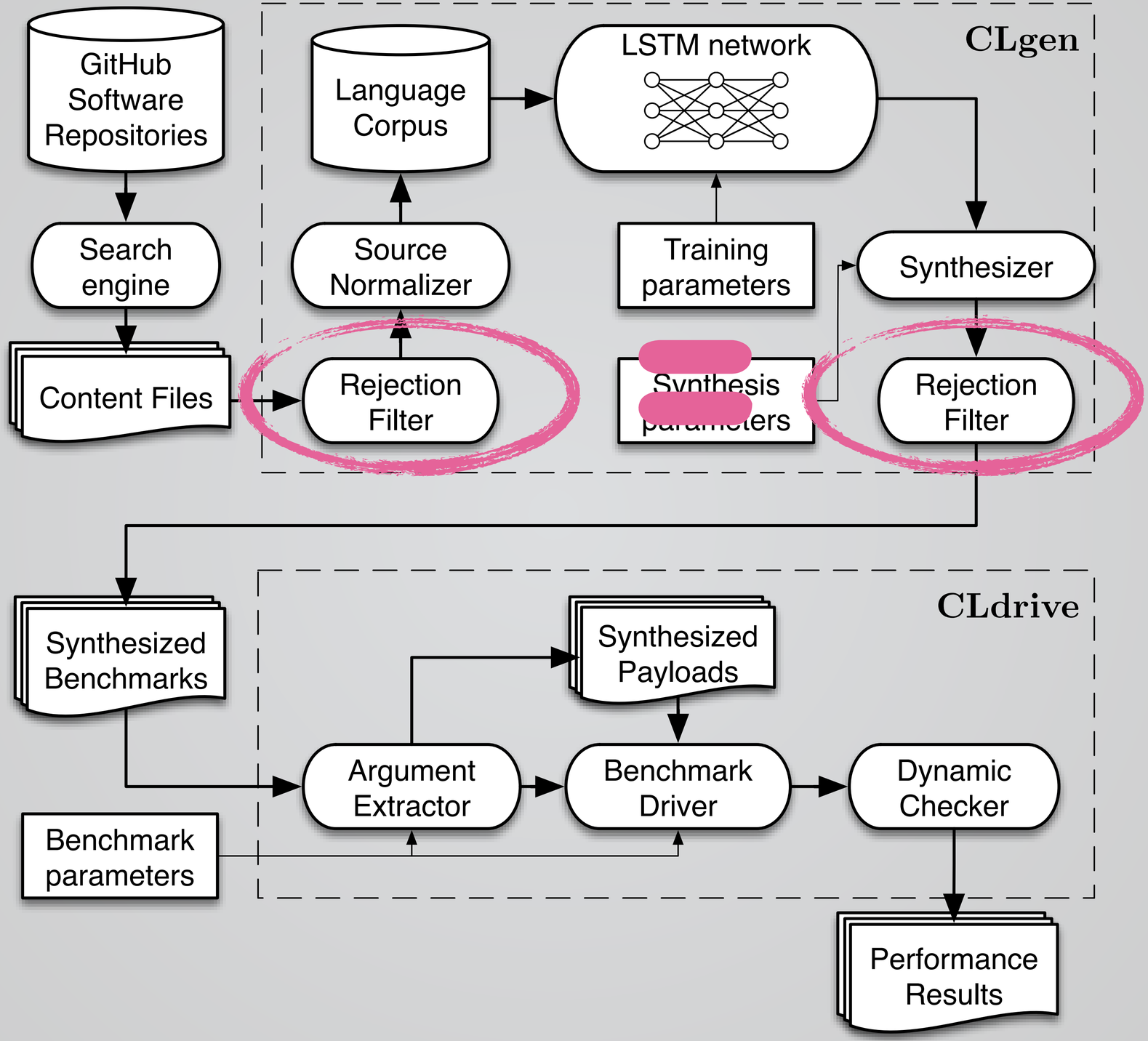


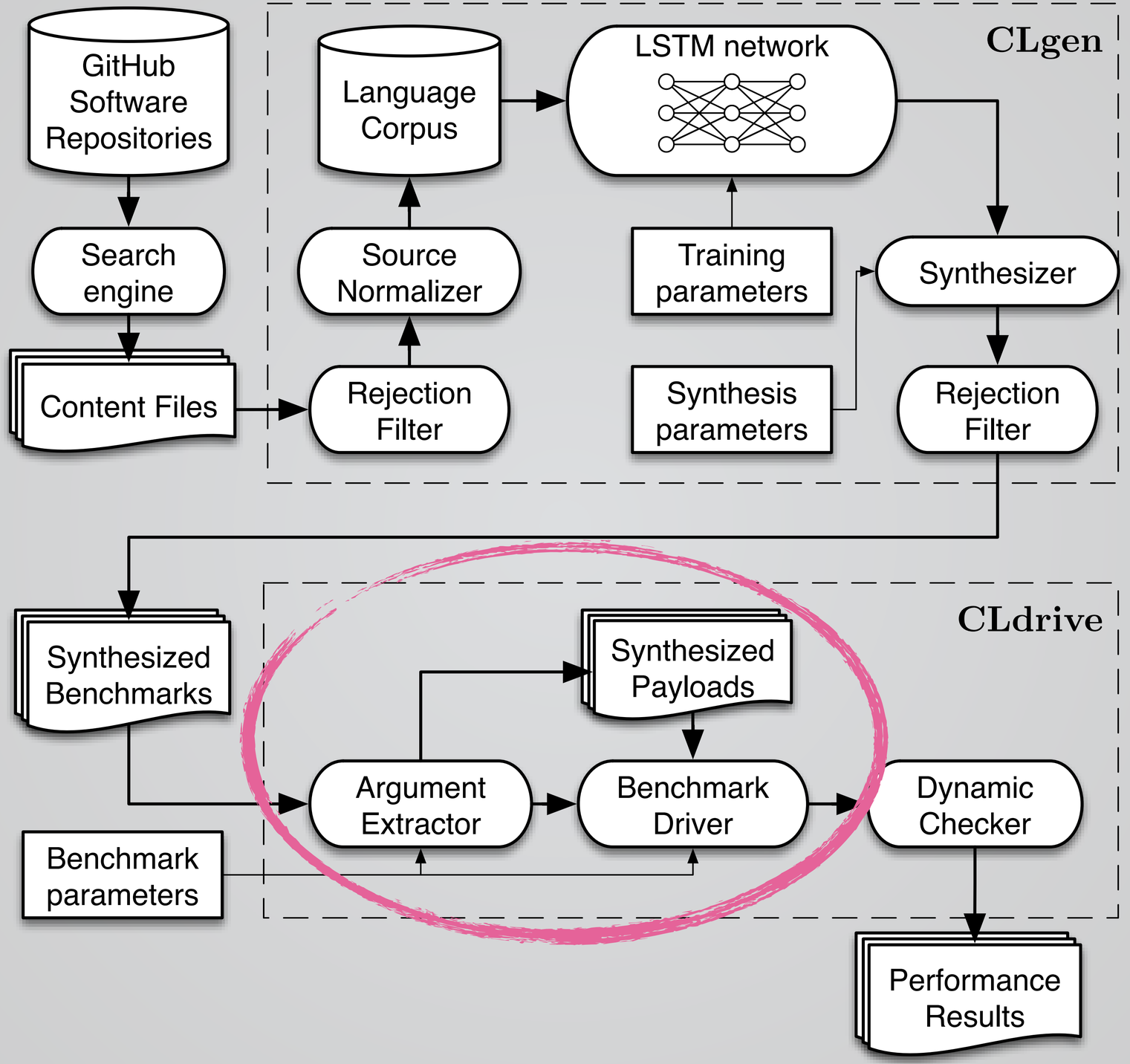
CLgen

Deep Learning Program Generator.

```
$ clgen
```

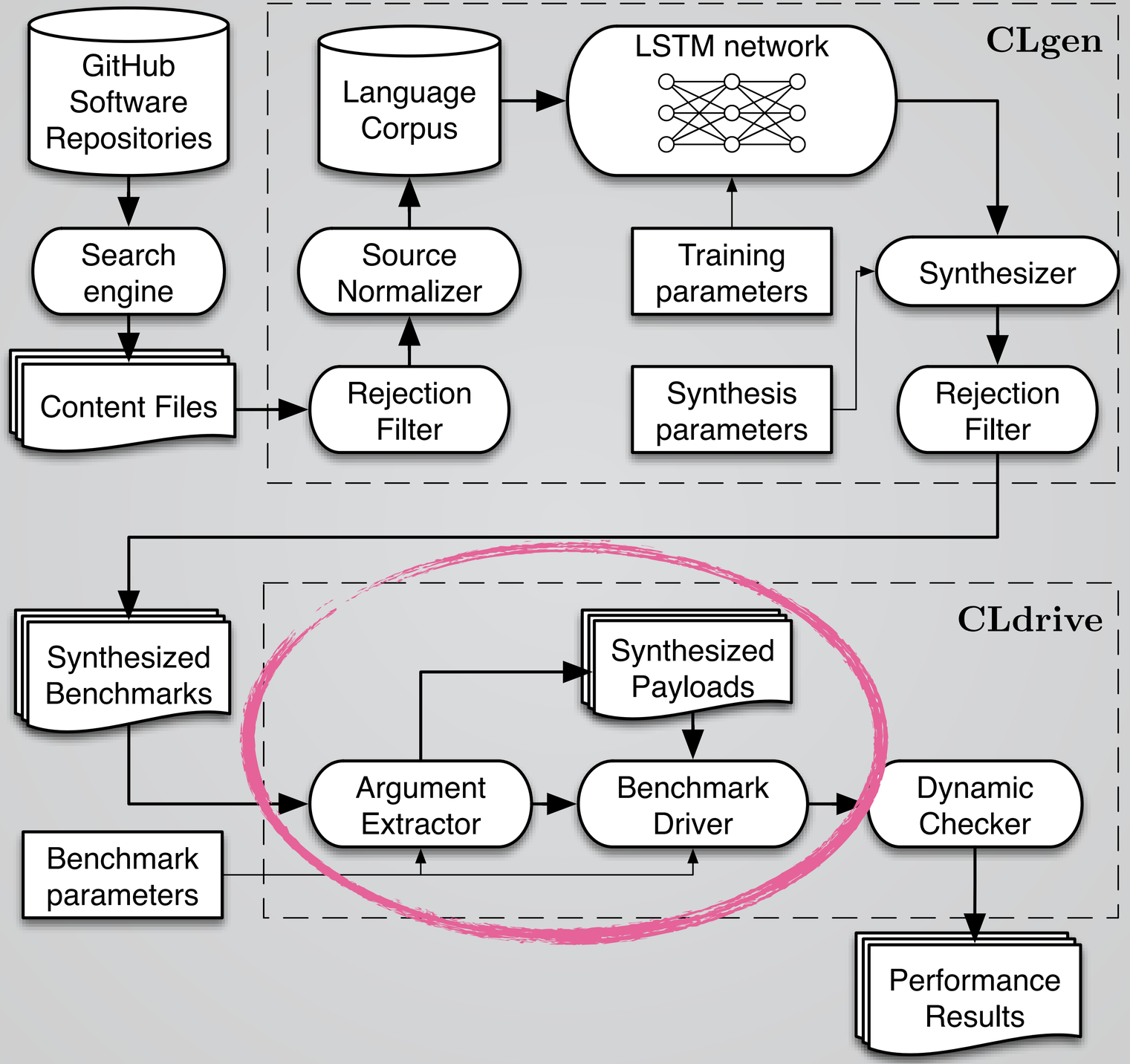

CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Language Corpus
Source Normalizer
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results

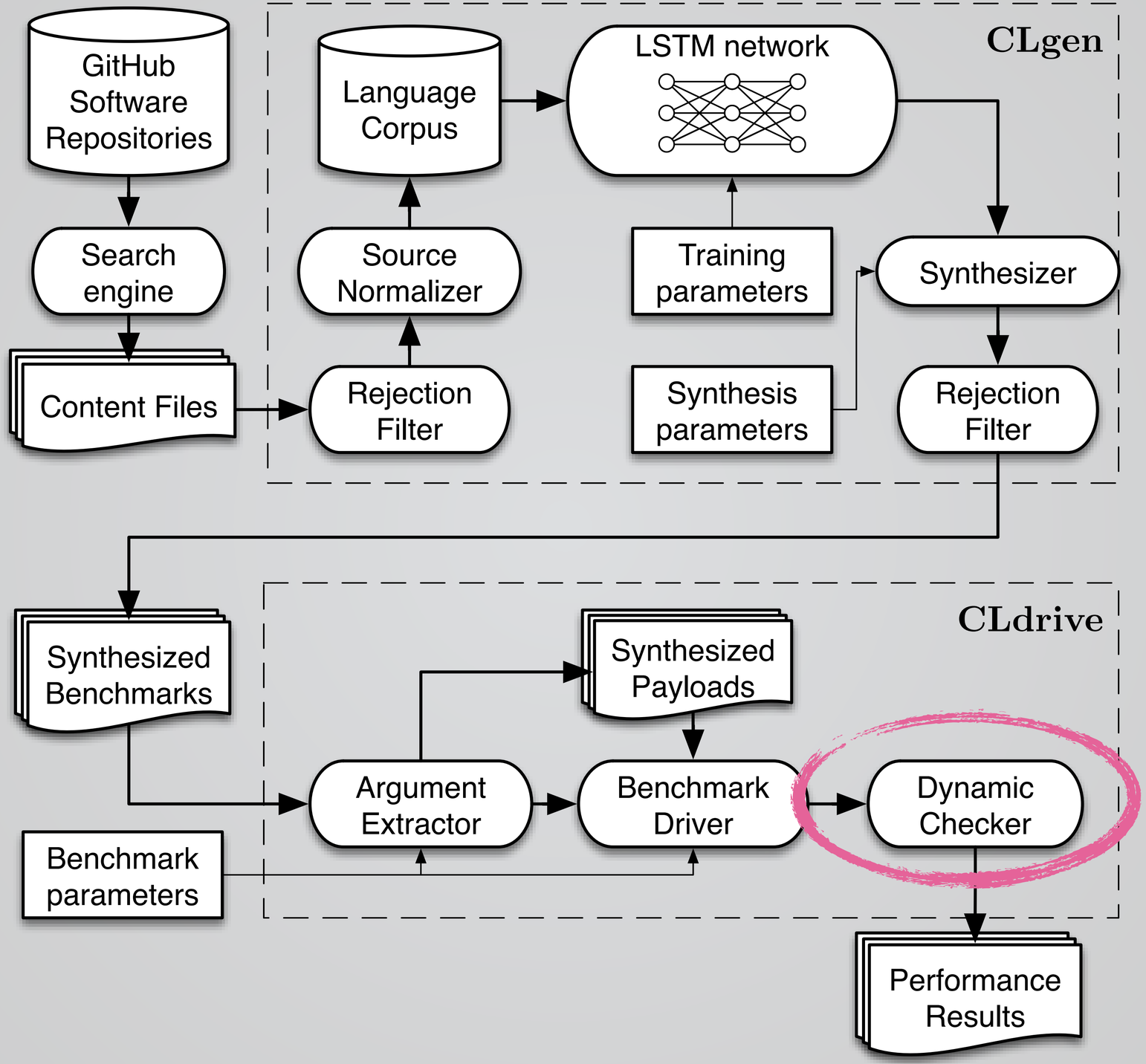

CLgen
GitHub Software Repositories
Search engine
Content Files
Language Corpus
Source Normalizer
Rejection Filter
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Synthesized Payloads
Argument Extractor
Benchmark Driver
Dynamic Checker
Performance Results


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Source Normalizer
Language Corpus
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results

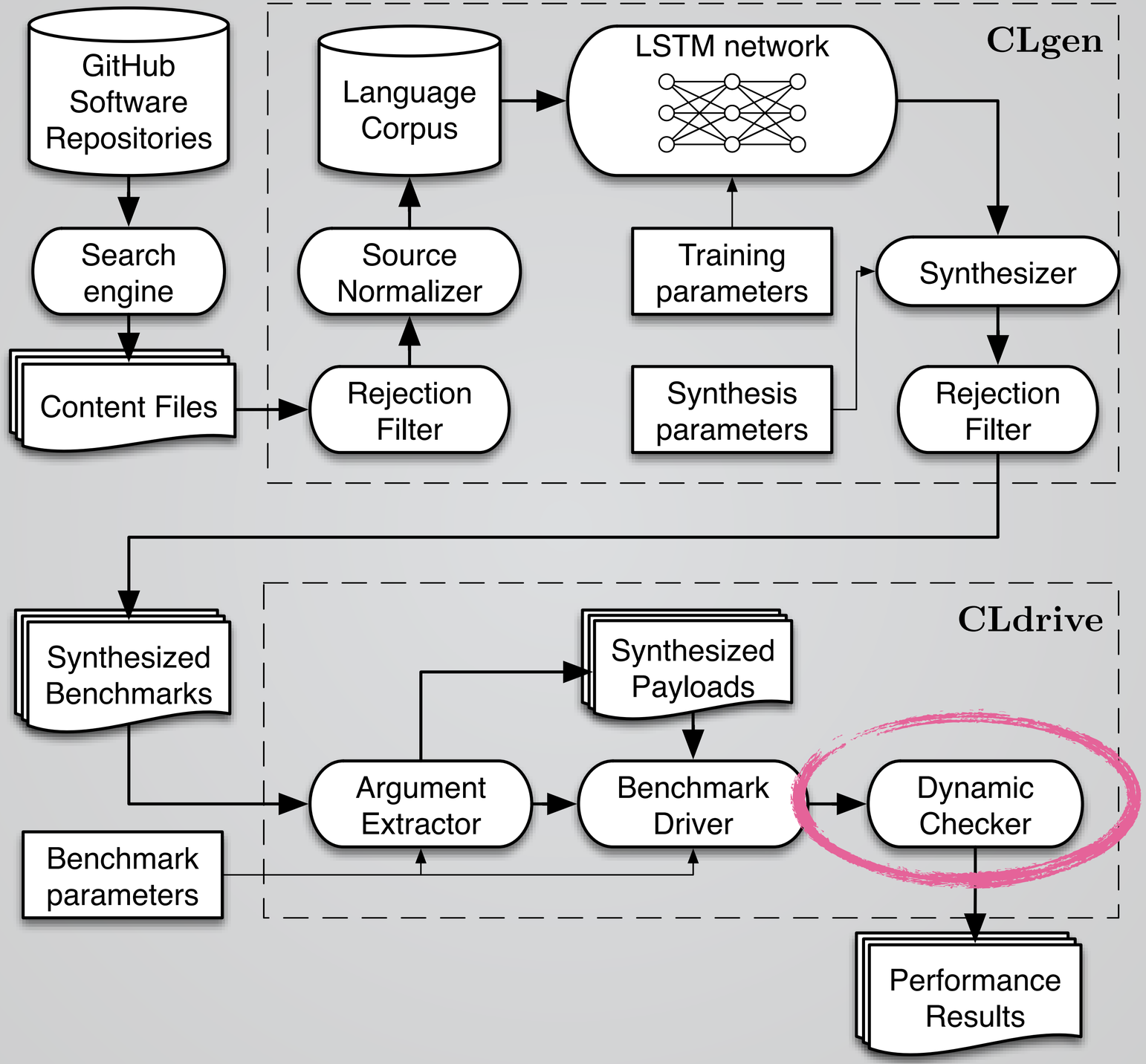
```
__kernel void A(__global float* a,
                __global float* b,
                __global float* c,
                const int d) {
  int e = get_global_id(0);
  if (e >= d) {
    return;
  }
  c[e] = a[e] + b[e] + 2 * a[e] + b[e] + 4;
}
```

```
__kernel void A(__global float* a,
                __global float* b,
                __global float* c,
                const int d) {
  int e = get_global_id(0);
  if (e >= d) {
    return;
  }
  c[e] = a[e] + b[e] + 2 * a[e] + b[e] + 4;
}
```


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Source Normalizer
Language Corpus
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Source Normalizer
Language Corpus
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results

```
_A = random_payload(_A)  # generate inputs
_B = random_payload(_B)
_C = copy(_C)
_D = copy(_B)

A_ = k(_A)  # compute outputs
B_ = k(_B)
C_ = k(_C)
D_ = k(_D)

# differential test
assert (A_ != _A || B_ != _B) else NO_OUTPUTS
assert (A_ != B_ || C_ != D_) else INPUT_INSENSITIVE
assert (A_ == C_ && B_ == D_) else NON_DETERMINISTIC
```


CLgen
GitHub Software Repositories
Search engine
Content Files
Rejection Filter
Source Normalizer
Language Corpus
LSTM network
Training parameters
Synthesis parameters
Synthesizer
Rejection Filter
CLdrive
Synthesized Benchmarks
Benchmark parameters
Argument Extractor
Synthesized Payloads
Benchmark Driver
Dynamic Checker
Performance Results

does it work?

humanorrobot.uk

humanorrobot.uk

Games About

Round 1

Player: **1002**, Robot: **998**

★☆☆☆☆ **What a piece of junk!**

By George S. Mitchell "gsmitchell"

I checked around Amazon as well as some other si tes, and decided these were what I needed. I read the reviews, looked at the pictures and bought two of these. I received them today. I unpacked one o f them, read the instructions and immediately filled out the return request for Amazon. These mounts are very, very heavy and require 6, yes SIX, screw s to mount to the wall. Are you kidding me? The T Vs are only 22 lbs! Two screws in the studs and th en more damage to the wall than is necessary. W hat a joke! And Lag bolts? Ever heard of screws? The other issue is that these are very hard to move . I bought an articulating wall mount for a reason, and I expect them to be able to move without a wr ench in my hand.

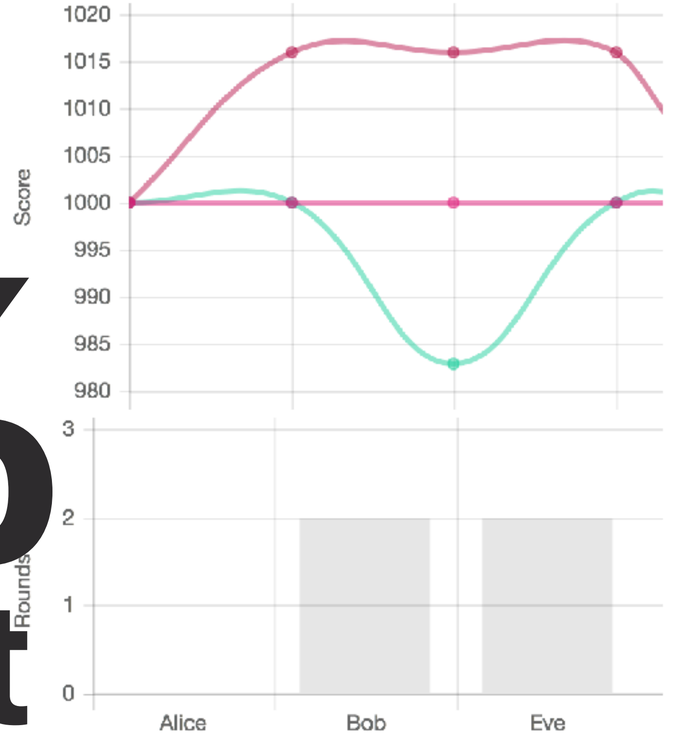
This side is the Robot

★★★★★ **Excellent for the price**

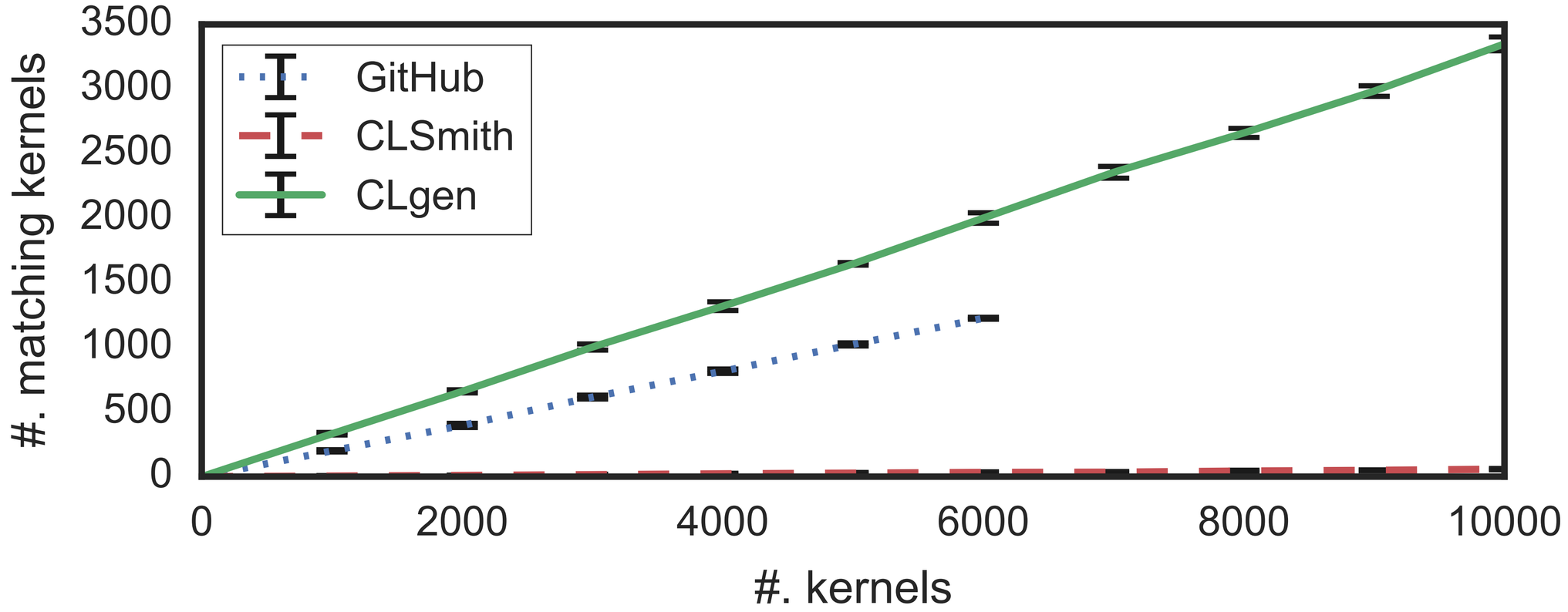
By A. Lopez "David Roberts"

This thing is fantastic. It has the same size and sp eed and sharp sound. I like the power turntable. I u se it every time you connect your computer and try ing to figure out you want to get a problem with a ti e and a duct. I ever h d problems with conve at co out. Good price for it and a coupl d a are w oftware installe o far I em happy w is product rks great for wh w anted to he wa aptop company a lo sive n't tell that any r devic this is a t product that works well for a reasonable price and works great for the pric e, and works

This side is the Robot

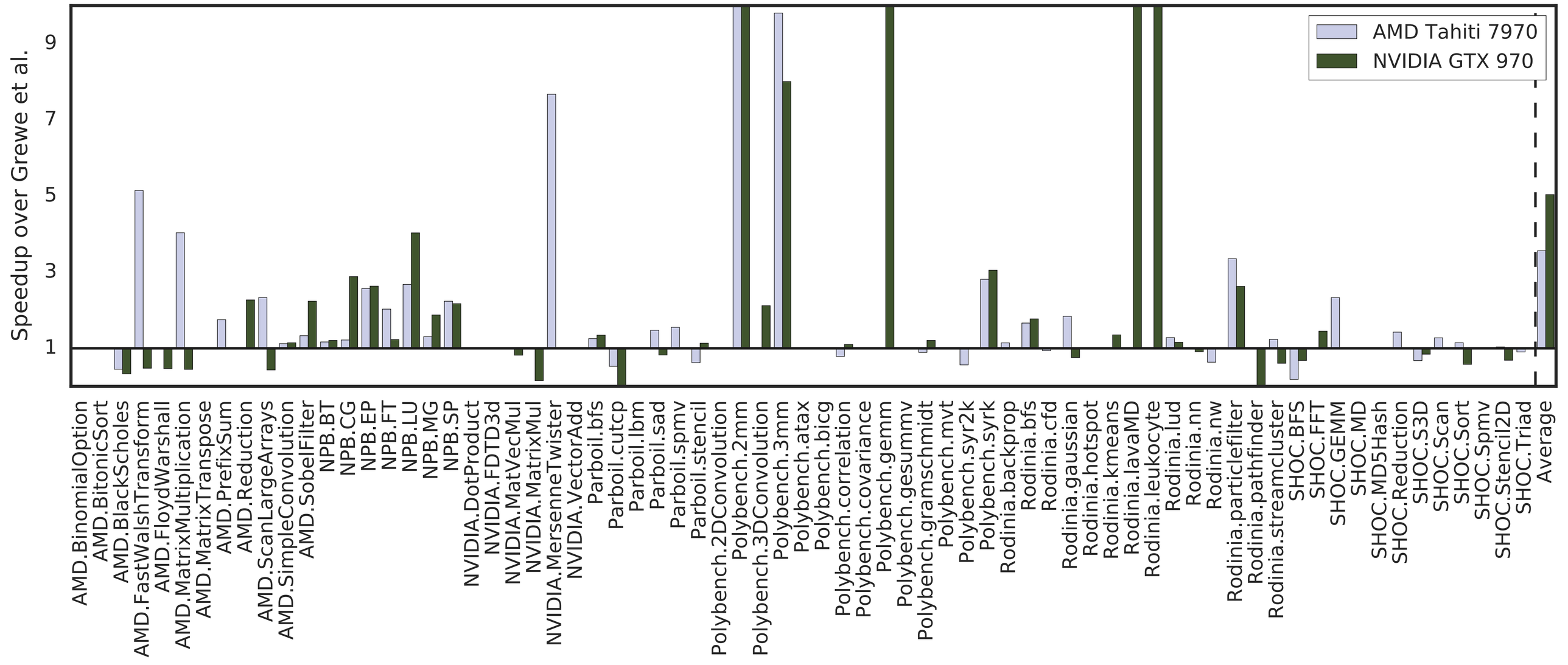


52% blind test



Raw Code Features		
comp	static	#. compute operations
mem	static	#. accesses to global memory
localmem	static	#. accesses to local memory
coalesced	static	#. coalesced memory accesses

7 programs, 1,000 synthetic benchmarks. 1.27x faster



71 programs, 1,000 synthetic benchmarks. 4.30x faster

Good Things

Basically* language agnostic.
35 million repos on GitHub. We're using 0.00004%.
Generates 2000 OpenCL benchmarks per machine per day.

Bad Things

No support for things declared outside of kernel scope.
Undirected almost to a fault.

what next?

very short term

paper + ae
google
eurolab
taco

next six months

discard rate < 90%
searching feature space
grammar

next six months

discard rate < 90%

searching feature space grammar

next six months

discard rate < 90%

1. alternate vocabularies
2. corpus encodings

next six months

discard rate < 90%

searching feature space grammar

next six months

discard rate < 90%

searching feature space

grammar

next six months

1. hill climb mutated seeds and RNG
2. train models with source + features

searching feature space

Experiments:
- convergence to 256 benchmarks
- can it beat GitHub for all kernels?

next six months

discard rate < 90%

searching feature space

grammar

next six months

discard rate < 90%

searching feature space

grammar

next six months

Oracle:

+ generate_identifier(S, regexp)
+ chose_next(S, production_rules[])

grammar

thesis

1. introduction
2. literature review
3. language modelling for program code
4. synthesising programs
5. searching the feature space
6. learning the compiler optimisation pipeline
7. conclusions

thesis

recap

papers

adapt

hlpgpu

cgo

posters

google
hipeac
pldi
acaces

talks

amazon
ocado
codeplay

idea Directed random program generation

idea Gamifying hyperparameter tuning through non-interactive turing tests

idea Optimisation pass selection and ordering using reinforcement learning