Library Learning for Neurally-Guided Bayesian Program Induction

Kevin Ellis[1], Lucas Morales[1], Mathias Sablé-Meyer[2],
Armando Solar-Lezama[1], Joshua B. Tenenbaum[1]

[1]: MIT. [2]: ENS Paris-Saclay.

List processing:

[5 2 9] → [9 2 5]

[1 1 2 2] → [2 2 1 1]

[1 2 3 2] → [2 3 2 1]

Text editing:

P Kohli → Dr. Kohli

Sumit Gulwani → Dr. Gulwani

Danny Tarlow → Dr. Tarlow

Symbolic regression:

$ax^3 + bx^2 + cx + d$

$a/(x-b)$

Explore/Compress/Compile (EC^2) learns to solve programming tasks like these by growing a library of code and training a neural net to search for programs written using the library

Library learning

$$[7\ \ 2\ \ 3] \rightarrow [7\ \ 3]$$
$$[1\ \ 2\ \ 3\ \ 4] \rightarrow [3\ \ 4]$$
$$[4\ \ 3\ \ 2\ \ 1] \rightarrow [4\ \ 3]$$

Library learning

$$[7\ \ 2\ \ 3] \rightarrow [7\ \ 3]$$
$$[1\ \ 2\ \ 3\ \ 4] \rightarrow [3\ \ 4]$$
$$[4\ \ 3\ \ 2\ \ 1] \rightarrow [4\ \ 3]$$

Library:

$f_1(\ell,\mathrm{p}) = \texttt{(foldr}\ \ell\ \texttt{nil (}\lambda\ \texttt{(x a) (if (p x) (cons x a) a)))}$

(f_1: *Higher-order filter function*)

(*Get elements x from ℓ where (p x) returns true*)

Library learning

$$[7\ 2\ 3] \rightarrow [7\ 3]$$
$$[1\ 2\ 3\ 4] \rightarrow [3\ 4]$$
$$[4\ 3\ 2\ 1] \rightarrow [4\ 3]$$

$$f(\ell) = (f_1\ \ell\ (\lambda\ (\mathtt{x})\ (>\ \mathtt{x}\ 2)))$$

Library:

$$f_1(\ell,\mathrm{p}) = (\mathtt{foldr}\ \ell\ \mathtt{nil}\ (\lambda\ (\mathtt{x}\ \mathtt{a})\ (\mathtt{if}\ (\mathtt{p}\ \mathtt{x})\ (\mathtt{cons}\ \mathtt{x}\ \mathtt{a})\ \mathtt{a})))$$

(f_1: *Higher-order filter function*)

(*Get elements x from* ℓ *where (p x) returns true*)

Subset of 38 learned library routines for list processing

$$f_0(\ell,\mathtt{r}) = \mathtt{(foldr\ r}\ \ell\ \mathtt{cons)}$$

f_0: *Append lists* $\mathtt{r}$ *and* ℓ

$$f_1(\ell,\mathtt{p}) = \mathtt{(foldr}\ \ell\ \mathtt{nil\ (}\lambda\ \mathtt{(x\ a)}$$
$$\mathtt{(if\ (p\ x)\ (cons\ x\ a)\ a)))}$$

f_1: *Higher-order filter function*

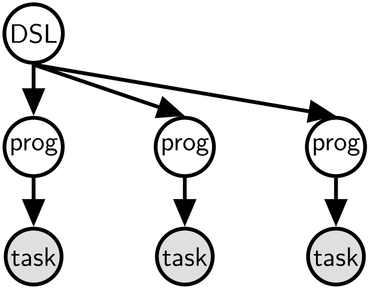
$$f_2(\ell) = \mathtt{(foldr}\ \ell\ \mathtt{0\ (}\lambda\ \mathtt{(x\ a)}$$
$$\mathtt{(if\ (>\ a\ x)\ a\ x)))}$$

f_2: *Maximum element in list* ℓ

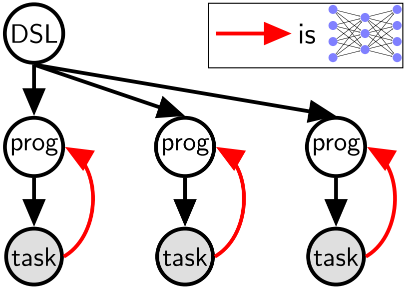
$$f_3(\ell,\mathtt{k}) = \mathtt{(foldr}\ \ell\ \mathtt{(is\text{-}nil}\ \ell\mathtt{)}$$
$$\mathtt{(}\lambda\ \mathtt{(x\ a)\ (if\ a\ a\ (=\ k\ x))))}$$

f_2: *Whether* ℓ *contains* $\mathtt{k}$

Explore/Compress/Compile as Bayesian Inference

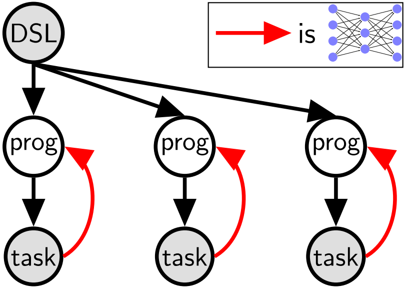


Explore/Compress/Compile as Amortized Bayesian Inference



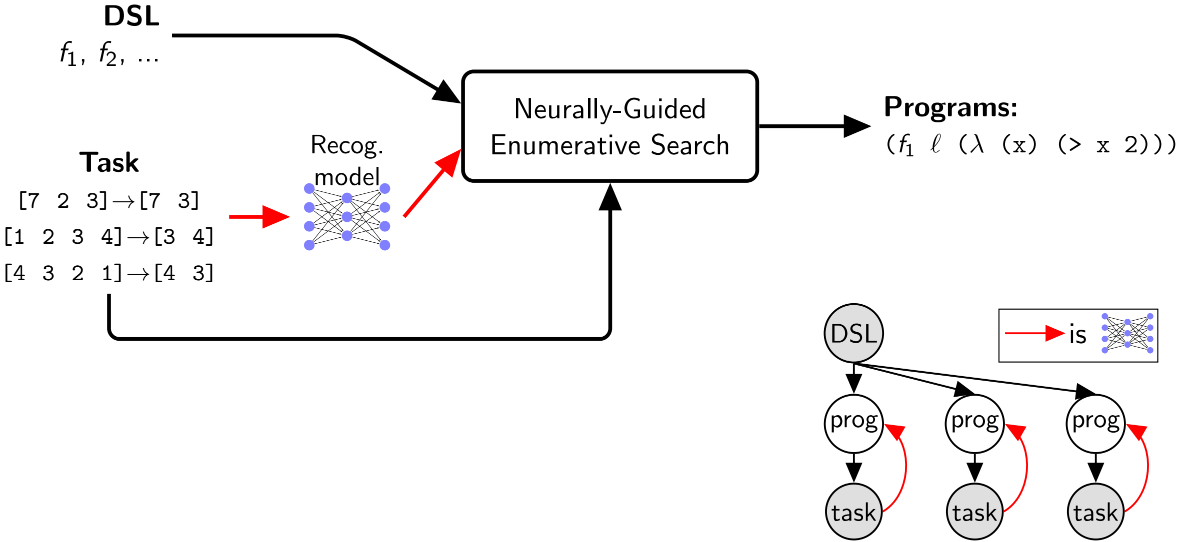
Explore/Compress/Compile as Amortized Bayesian Inference

Explore: Infer programs, fixing DSL and neural recognition model



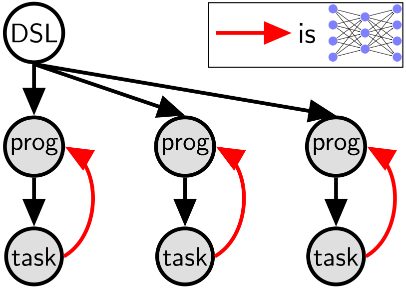
Explore/Compress/Compile as Amortized Bayesian Inference

Explore: Infer programs, fixing DSL and neural recognition model



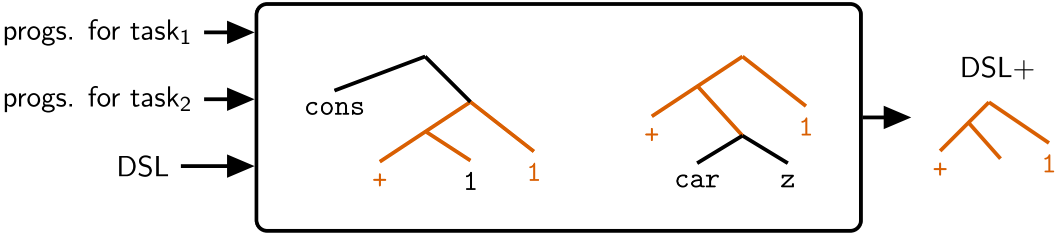
Explore/**Compress**/Compile as Amortized Bayesian Inference

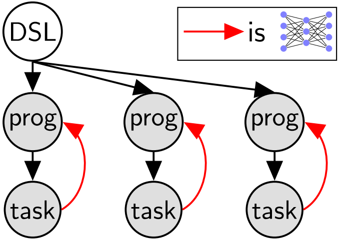
Compress: Update DSL, fixing programs



Explore/**Compress**/Compile as Amortized Bayesian Inference

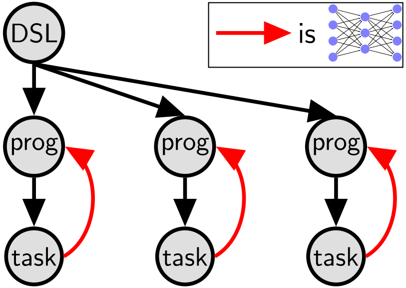
Compress: Update DSL, fixing programs





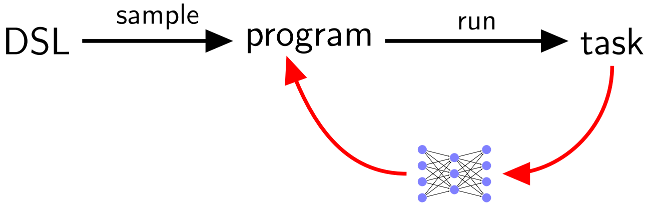
Explore/Compress/**Compile** as Amortized Bayesian Inference

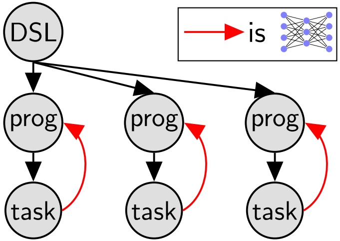
Compile: Train recognition model



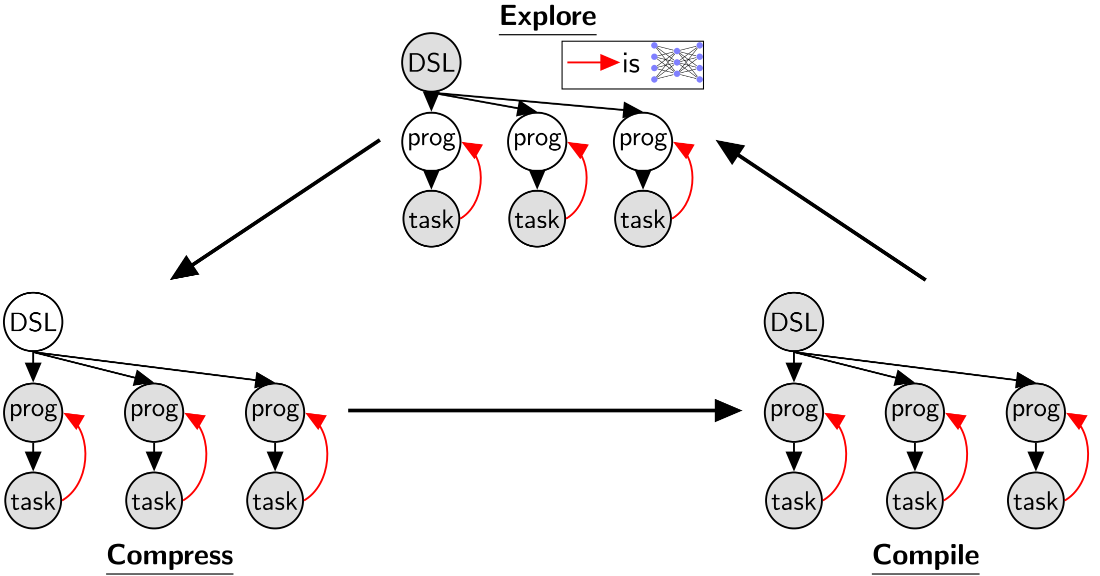
Explore/Compress/**Compile** as Amortized Bayesian Inference

Compile: Train recognition model





Explore/Compress/Compile as Amortized Bayesian Inference

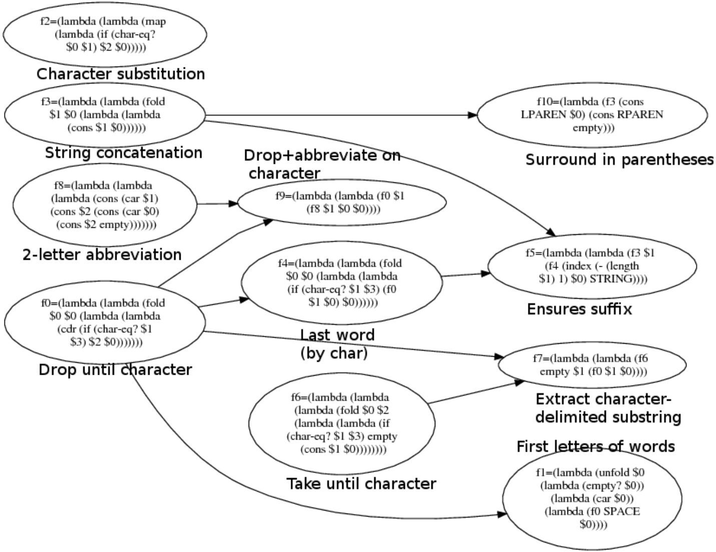


Domain: Text Editing

In the style of FlashFill (Gulwani 2012). Starts with `map`, `fold`, etc.

Input	Output
+106 769-438	106.769.438
+83 973-831	83.973.831
Temple Anna H	TAH
Lara Gregori	LG

Text editing: Library learning builds on itself

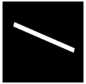


Learned DSL primitives over 3 iterations (3 columns). Learned primitives call each other (arrows).

Programs with numerical parameters: Symbolic regression from visual input

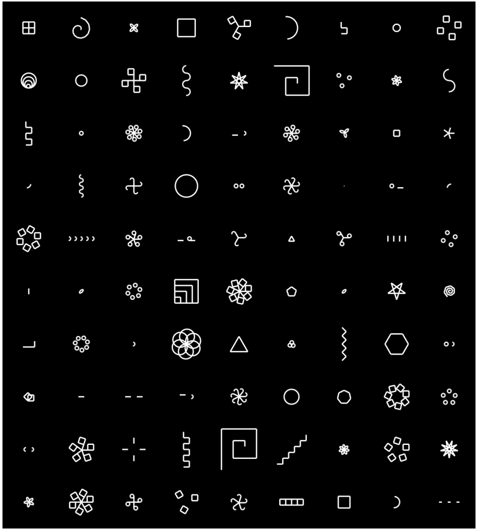
Fits parameters by autograd-ing thru program

Recognition model looks at picture of function's graph

Programs & Tasks		DSL
 $f(\mathtt{x}) = (f_1\ \mathtt{x})$	 $f(\mathtt{x}) = (f_5\ \mathtt{x})$	$f_0(\mathtt{x}) = (\mathtt{+\ x\ real})$ $f_1(\mathtt{x}) = (f_0\ (\mathtt{*\ real\ x}))$ $f_2(\mathtt{x}) = (f_1\ (\mathtt{*\ x}\ (f_0\ \mathtt{x})))$ $f_3(\mathtt{x}) = (f_0\ (\mathtt{*\ x}\ (f_2\ \mathtt{x})))$
 $f(\mathtt{x}) = (f_4\ \mathtt{x})$	 $f(\mathtt{x}) = (f_3\ \mathtt{x})$	$f_4(\mathtt{x}) = (f_0\ (\mathtt{*\ x}\ (f_3\ \mathtt{x})))$ *(f_4: 4th order polynomial)* $f_5(\mathtt{x}) = (\mathtt{/\ real}\ (f_0\ \mathtt{x}))$ *(f_5: rational function)*

New domain: Generative graphics programs (Turtle/LOGO)

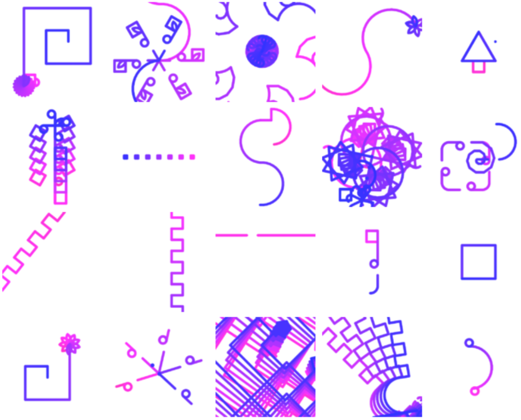
Training tasks:



Generative graphics programs: Samples from DSL

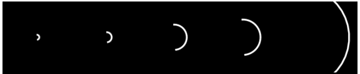
DSL samples before learning

DSL samples after learning



Generative graphics programs: Learned library contains parametric drawing routines

Semicircle:



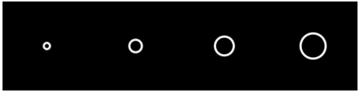
Greek Spiral:

Polygons & Stars:

Spiral:

S-Curves:

Circles:



Learning to program: Poster AB #24

f_2(p,f,n,x) = (if (p x) nil
                    (cons (f x) (f_2 (n x))))

(f_2: *unfold*)

f_3(i,l) = (if (= i 0) (car l)
                    (f_3 (f_1 i) (cdr l))))

(f_3: *index*)

f_4(f,l,x) = (if (empty? l) x
                    (f (car l) (f_4 (cdr l))))

(f_4: *fold*)

f_5(f,l) = (if (empty? l) nil
                    (cons (f (car l)) (f_5 (cdr l)))

(f_5: *map*)