# Neural Symbolic Machines

## Neural Program Induction for Semantic Parsing

Chen Liang, Ni Lao
Collaborate: Jonathan Berant, Quoc Le, Kenneth Forbus

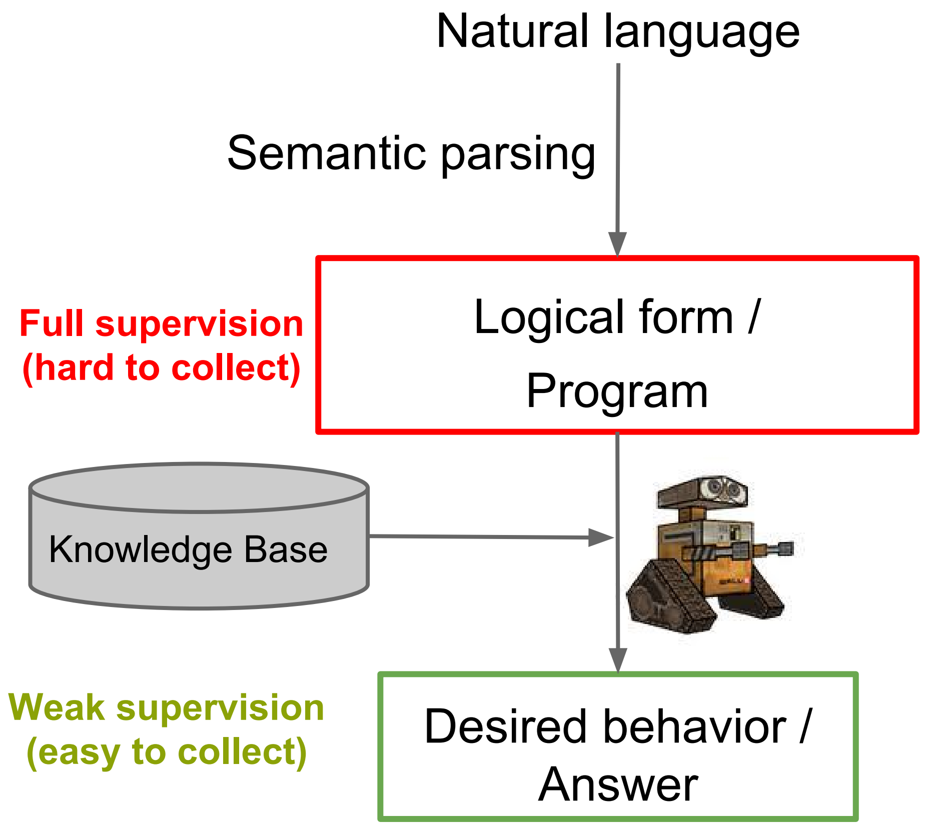
## Overview

- Semantic parsing: (updated) WebQuestions dataset
- Neural program induction
- Manager-Programmer-Computer (MPC) framework
- Neural Symbolic Machine
- Experiments and analysis

[Berant, et al 2013]
[Liang 2013]

## Semantic Parsing

- Question answering with structured data (KG / tables / personal data)
- Voice to action
- Personal assistant


Natural language
Semantic parsing
Full supervision (hard to collect)
Logical form / Program
Knowledge Base
Weak supervision (easy to collect)
Desired behavior / Answer

# Question Answering with Knowledge Base

## Large-scale Knowledge Base

Properties of Hundreds of millions of entities

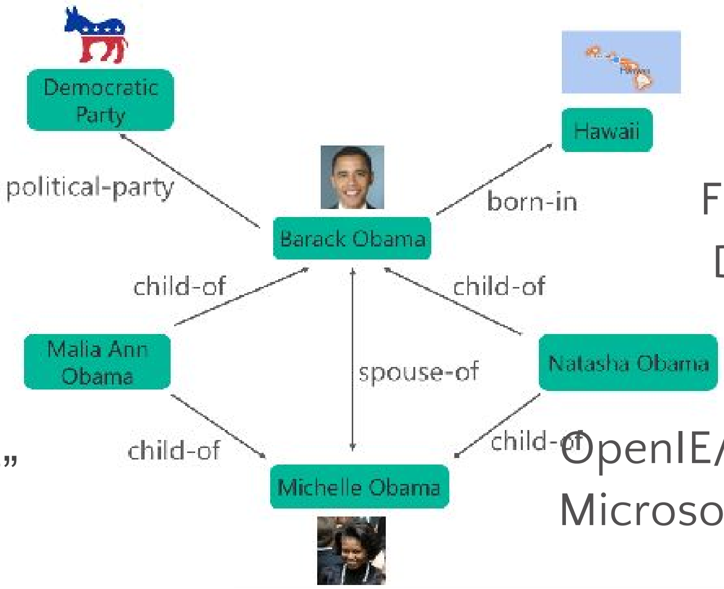
Plus relations among them

E.g. Freebase, 26k predicates, 200M entities, 3B triples

## Question Answering

"What are the names of Obama's daughters?"

$\lambda x.\text{parent}(\text{Obama},x) \cap \text{gender}(x,\text{Female})$



Freebase

DBpedia

YAGO

NELL

OpenIE/ReVerb

Microsoft Satori

Slides adapted from [Yih+ 2016]

[Berant+ 13]

# WebQuestions Dataset

- What character did Natalie Portman play in Star Wars? ⇒ Padme Amidala
- What currency do you use in Costa Rica? ⇒ Costa Rican colon
- What did Obama study in school? ⇒ political science
- What do Michelle Obama do for a living? ⇒ writer, lawyer
- What killed Sammy Davis Jr? ⇒ throat cancer

[Examples from Berant]

- 5,810 questions crawled from Google Suggest API and answered using Amazon MTurk
  - 3,778 training, 2,032 testing
  - A question may have multiple answers using Avg. F1 as main evaluation metric

Slides adapted from [Yih+ 2016]

[Yih+ 16]

## WebQuestionsSP Dataset

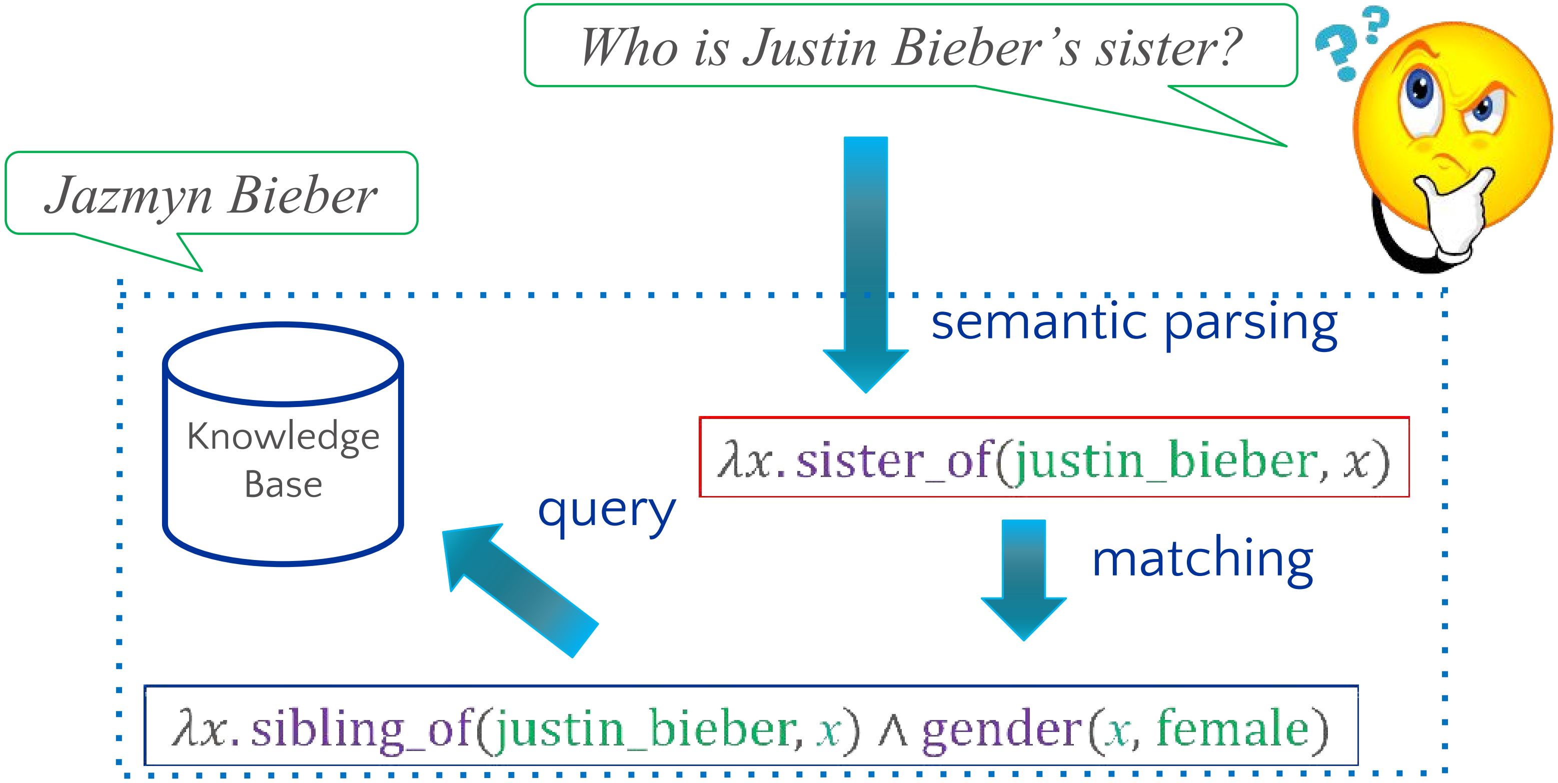
- Remove invalid QA pairs
- Add logical form annotations
- Result in 3,098 training and 1,639 testing questions

```
"Version": "1.0",
"FreebaseVersion": "2015-08-09",
"Questions": [ {
   "RawQuestion": "what does jamaican people speak?",
   "Parses": [
    {
      "ParseId": "WebQTest-0.P0",
      "Sparql": " ... ",
      "TopicEntityName": "Jamaica",
      "TopicEntityMid": "m.03_r3",
      "InferentialChain": ["location.country.languages_spoken" ],
      "Answers": [
        { "AnswerArgument": "m.01428y", },
        { "AnswerArgument": "m.04ygk0", }
      ]
    },
    {
      "ParseId": "WebQTest-0.P1",
      "Sparql": "...",
      ...
      "InferentialChain": ["location.country.official_language" ],
   ....
```

Available at http://aka.ms/WebQSP

# Generic Semantic Parsing

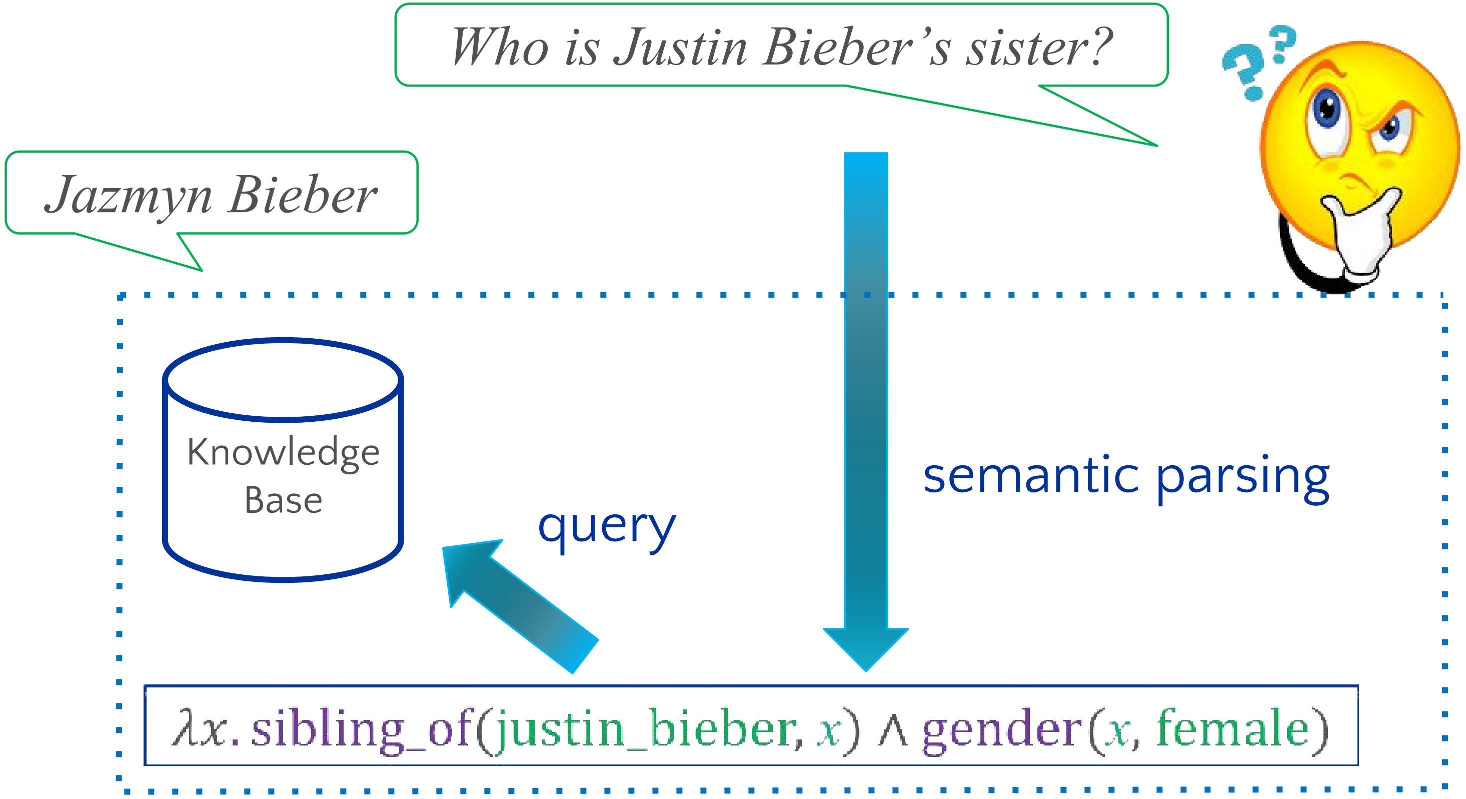
[Kwiatkowski+ 13]


*Who is Justin Bieber's sister?*
*Jazmyn Bieber*
semantic parsing
Knowledge Base
$\lambda x.\,\text{sister\_of}(\text{justin\_bieber}, x)$
query
matching
$\lambda x.\,\text{sibling\_of}(\text{justin\_bieber}, x) \wedge \text{gender}(x, \text{female})$

Slides adapted from [Yih+ 2016]

[Berant+ 13]

## KB-Specific Semantic Parsing


*Who is Justin Bieber's sister?*
*Jazmyn Bieber*
Knowledge Base
query
semantic parsing
$\lambda x.\,\mathrm{sibling\_of}(\mathrm{justin\_bieber}, x) \wedge \mathrm{gender}(x, \mathrm{female})$

Slides adapted from [Yih+ 2016]

# Key Challenges

- Language mismatch
  - Lots of ways to ask the same question

    *"What was the date that Minnesota became a state?"*

    *"When was the state Minnesota created?"*
  - Need to map them to the predicate defined in KB

    location.dated_location.date_founded
- Compositionality
  - The semantics of a question may involve multiple predicates and entities
- Large search space
  - Some Freebase entities have >160,000 immediate neighbors
  - 26k predicates in Freebase

Slides from [Yih+ 2016]

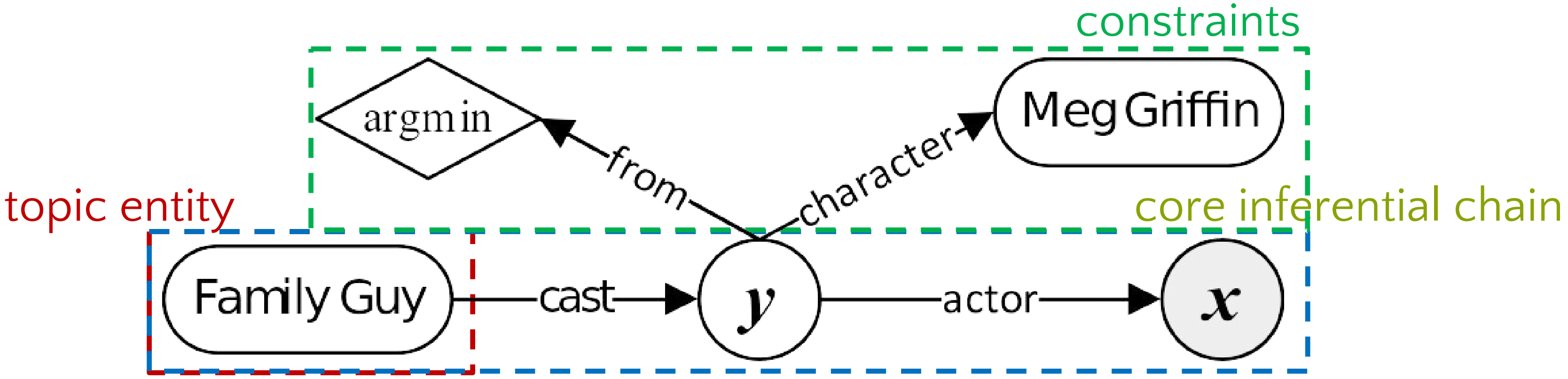
[Yih, et al 2016]

# Previous State-of-the-Art

## Staged Query Graph Generation

Who first voiced Meg on Family Guy?

$\lambda x. \exists y. \text{cast}(\text{FamilyGuy}, y) \wedge \text{actor}(y, x) \wedge \text{character}(y, \text{MegGriffin})$



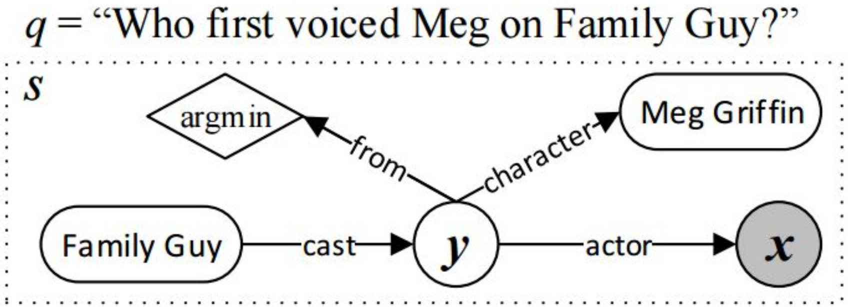
[Yih+ 2016]

[Yih, et al 2016]

# Previous State-of-the-art

## Hand crafted rules and features

- 6 rules to determine if a constraint or aggregation should be added to cvt nodes
- 3 extra rules for answer nodes
- 7 features for cvt nodes
- 3 features for answer nodes
- 3 CNN matching scores

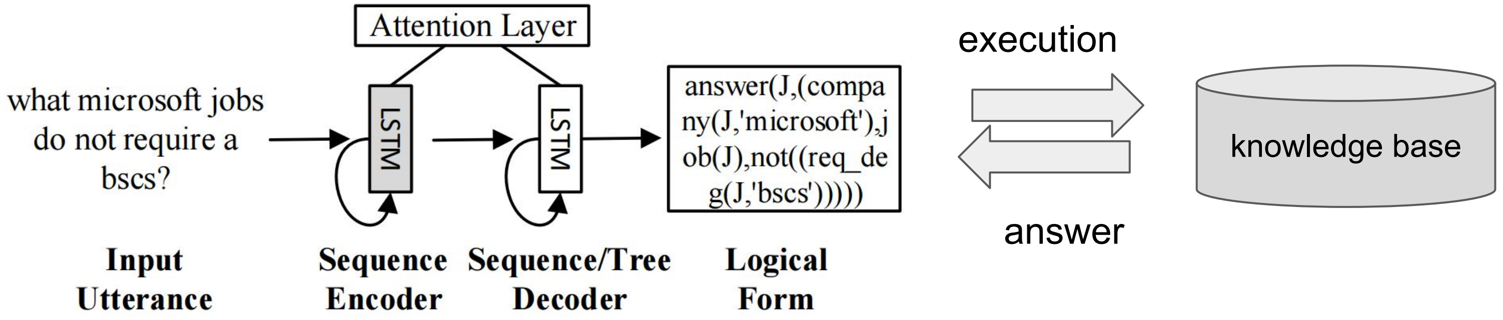


(1) EntityLinkingScore(FamilyGuy, "Family Guy") = 0.9
(2) PatChain("who first voiced meg on <e>", cast-actor) = 0.7
(3) QuesEP($q$, "family guy cast-actor") = 0.6
(4) ClueWeb("who first voiced meg on <e>", cast-actor) = 0.2
(5) ConstraintEntityWord("Meg Griffin", $q$) = 0.5
(6) ConstraintEntityInQ("Meg Griffin", $q$) = 1
(7) AggregationKeyword(argmin, $q$) = 1
(8) NumNodes(s) = 5
(9) NumAns(s) = 1

[Jia & Liang, 2016; Dong & Lapata, 2016]

## Pure seq2seq approaches to semantic parsing

- Require strong / full supervision
  - Only applied on domains with tiny schemas/vocabs and known semantic annotations: JOBS, GEO, ATIS, IFTTT
- Lack of integration with the executer
  - Only after the whole program is written can any feedback be generated from the machine
  - No way to search over the exponentially large hypothesis space given a large schema (e.g. Freebase has 26k predicates, 100M entities)
  - **The entire burden is on the neural network**



## Overview

- Semantic parsing: (updated) WebQuestions dataset
- Neural program induction
- Manager-Programmer-Computer (MPC) framework
- Neural Symbolic Machine
- Experiments and analysis

# Program induction & Memory

- **Differentiable function => computable function**
  - Use differentiable memories for backprop training
- Tasks
  - Mostly simple tasks like addition or sorting
  - Hard to scale to large knowledge bases
- A lot of recent progress
  - Neural Turing machine [Graves+ 2014]
  - Memory network [Sukhbaatar+ 2015]
  - Dynamic memory network [Kumar+ 2015]
  - Stack-augmented RNN [Joulin&Mikolov 2015]
  - Queue & stack LSTM [Grefenstette+ 2015]
  - Neural Programmer Interpreter [Reed&Freitas 2015]
  - Neural programmer [Neelakantan+ 2015]
  - Differentiable neural computer [Graves+ 2016]
  - Dynamic neural module network [Andreas+ 2016]
  - Dynamic Neural Turing Machine [Gulcehre+ 2016]

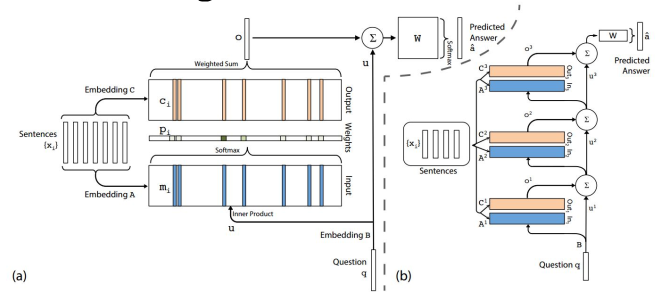
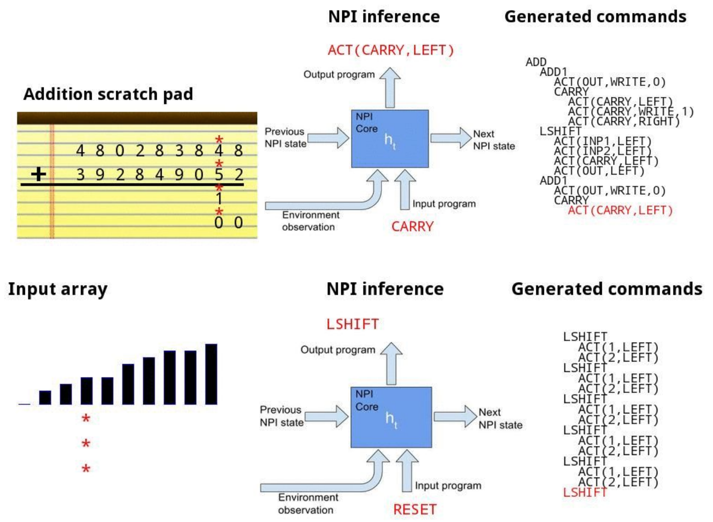


Figure 1: (a): A single layer version of our model. (b): A three layer version of our model. In practice, we can constrain several of the embedding matrices to be the same (see Section 2.2).

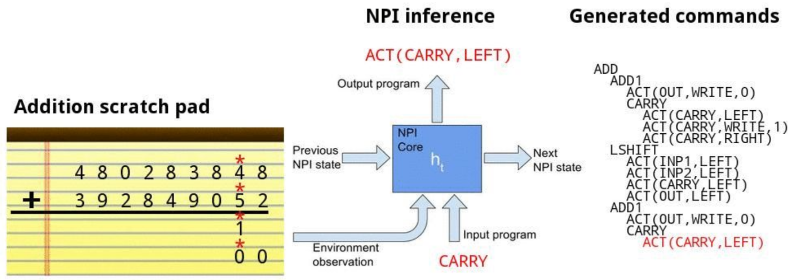


```
ADD
  ADD1
    ACT(OUT,WRITE,0)
    CARRY
      ACT(CARRY,LEFT)
      ACT(CARRY,WRITE,1)
      ACT(CARRY,RIGHT)
  LSHIFT
    ACT(INP1,LEFT)
    ACT(INP2,LEFT)
    ACT(CARRY,LEFT)
    ACT(OUT,LEFT)
  ADD1
    ACT(OUT,WRITE,0)
    CARRY
      ACT(CARRY,LEFT)
```

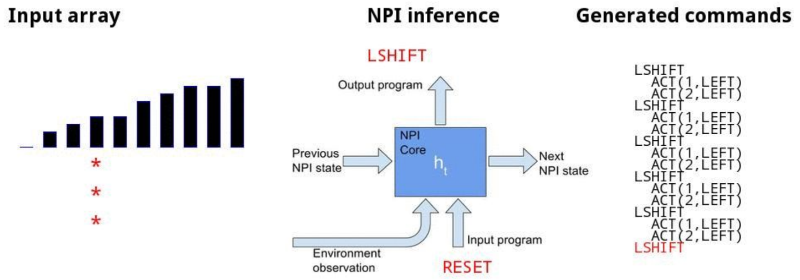
```
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
```

# How about giving NN regular operations and memory?

- Impressive example to demonstrate the power but...

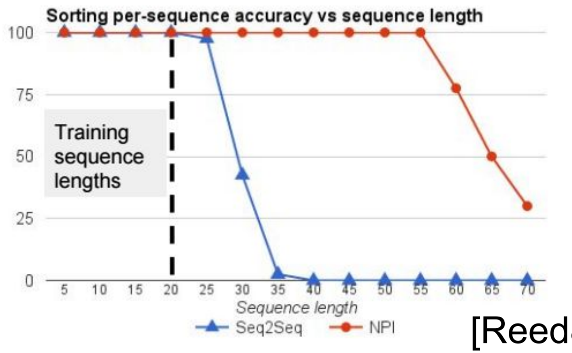


```
ADD
  ADD1
    ACT(OUT,WRITE,0)
    CARRY
      ACT(CARRY,LEFT)
      ACT(CARRY,WRITE,1)
      ACT(CARRY,RIGHT)
  LSHIFT
    ACT(INP1,LEFT)
    ACT(INP2,LEFT)
    ACT(CARRY,LEFT)
    ACT(OUT,LEFT)
  ADD1
    ACT(OUT,WRITE,0)
    CARRY
      ACT(CARRY,LEFT)
```



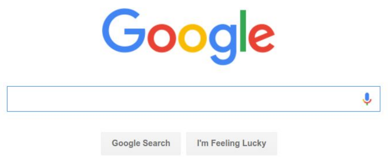
```
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
  ACT(1,LEFT)
  ACT(2,LEFT)
LSHIFT
```

- The operations learned are not as scalable and reliable.



[Reed&Freitas 2015]

- Why not leverage existing modules which are scalable and reliable?



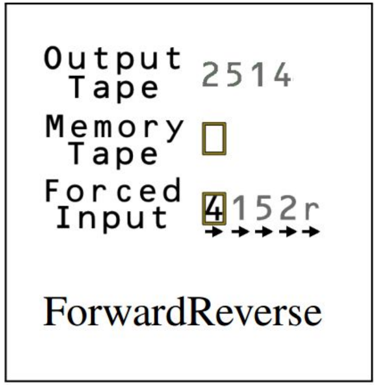
[Zaremba&Sutskever 2016]

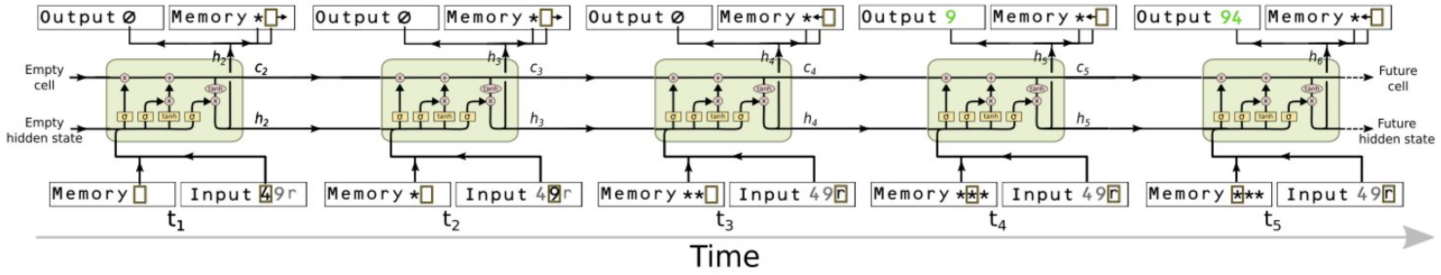
[Zaremba&Sutskever 2016]

# Reinforcement Learning Neural Turing Machines

- Interact with a discrete Interfaces
  - a memory Tape, an input Tape, and an output Tape
- Use Reinforcement Learning algorithm to train
- Solve simple algorithmic tasks
  - E.g., reversing a string

Need higher level programming language for semantic parsing





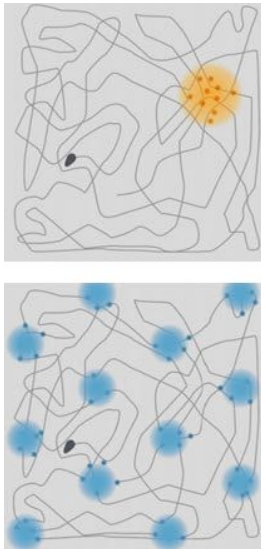
## Overview

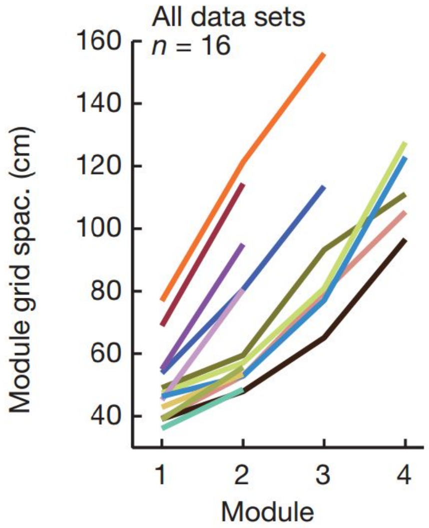
- Semantic parsing: (updated) WebQuestions dataset
- Neural program induction
- Manager-Programmer-Computer (MPC) framework
- Neural Symbolic Machine
- Experiments and analysis

[Stensola+ 2012]

## Symbolic Machines in Brains

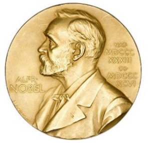
$$\begin{tikzcd}
\boxed{\text{Environment}} \arrow[r, shift left] & \boxed{\text{Brain}} \arrow[l, shift left] \arrow[r, shift left] & \boxed{\text{Symbolic Modules}} \arrow[l, shift left]
\end{tikzcd}$$





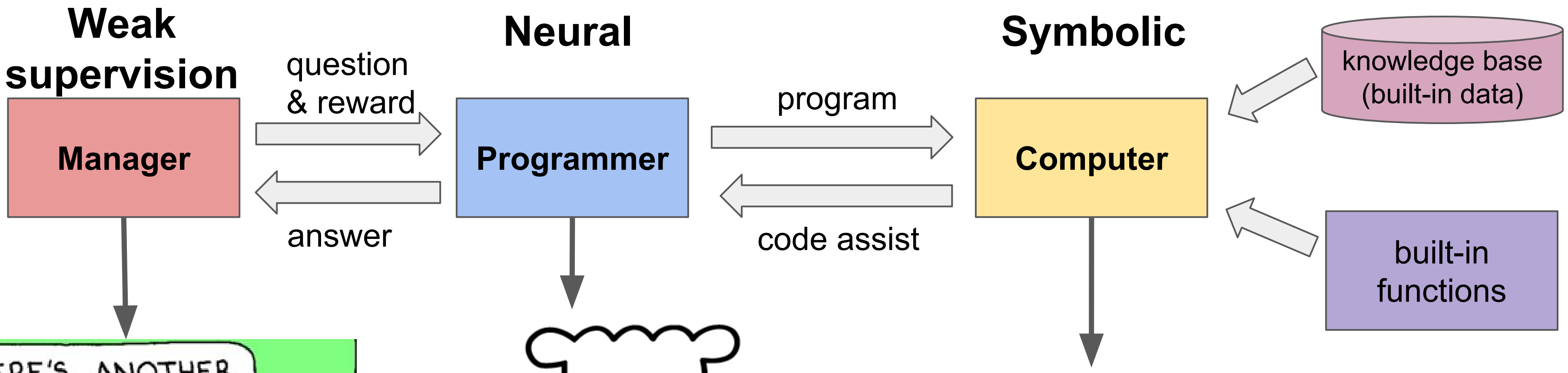
Mean grid spacing for all modules (M1–M4) in all animals (colour-coded)

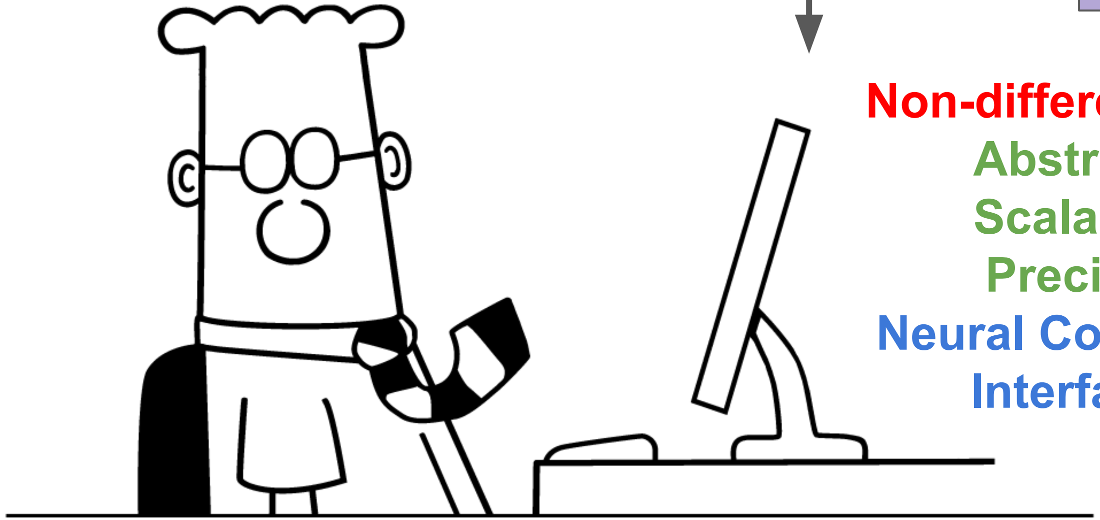
- 2014 Nobel Prize in Physiology or Medicine awarded for ‘inner GPS’ research



- Positions are represented as discrete numbers in animals' brains, which enable accurate and autonomous calculations

# The MPC Framework





Non-differentiable
Abstract
Scalable
Precise
Neural Computer Interface

## Overview

- Semantic parsing: (updated) WebQuestions dataset
- Neural program induction
- Manager-Programmer-Computer (MPC) framework
- Neural Symbolic Machine
- Experiments and analysis

## Knowledge Base & Semantic Parsing

- Knowledge graph
  - Let $E$ denote a set of entities (e.g., ABELINCOLN), and
  - let $P$ denote a set of relations (or properties, e.g., PLACEOFBIRTH)
  - A knowledge base $K$ is a set of assertions or triples $(e_1, p, e_2) \in E \times P \times E$
    e.g., (ABELINCOLN, PLACEOFBIRTH, HODGENVILLE)

- Semantic parsing
  - given a knowledge base $K$, and a question $q = (w_1, w_2, ..., w_k)$,
  - produce a program or logical form $z$ that
    when executed against $K$ generates the right answer $y$

## Lisp: High-level Language with Uniform Syntax

- Predefined functions, equivalent to a subset of λ-calculus
  - A program C is a list of expressions (c1...cl)
  - An expression is either a special token "Return" or a list "( F A0 ... Ak )"
  - F is one of the functions

$$
\begin{array}{c}
\hline
(\,\mathit{Hop}\ v\ p\,) \Rightarrow \{e_2 | e_1 \in v, (e_1, p, e_2) \in \mathbb{K}\} \\
(\,\mathit{ArgMax}\ v\ p\,) \Rightarrow \{e_1 | e_1 \in v, \exists e_2 \in \mathcal{E} : (e_1, p, e_2) \in \mathbb{K}, \forall e : (e_1, p, e) \in \mathbb{K}, e_2 \geq e\} \\
(\,\mathit{ArgMin}\ v\ p\,) \Rightarrow \{e_1 | e_1 \in v, \exists e_2 \in \mathcal{E} : (e_1, p, e_2) \in \mathbb{K}, \forall e : (e_1, p, e) \in \mathbb{K}, e_2 \leq e\} \\
(\,\mathit{Equal}\ v_1\ v_2\ p\,) \Rightarrow \{e_1 | e_1 \in v_1, \exists e_2 \in v_2 : (e_1, p, e_2) \in \mathbb{K}\} \\
\hline
\end{array}
$$

  - An argument Ai can be either a relation p ∈ P or a variable v
  - A variable v is a special token (e.g. "R1") representing a list of entities

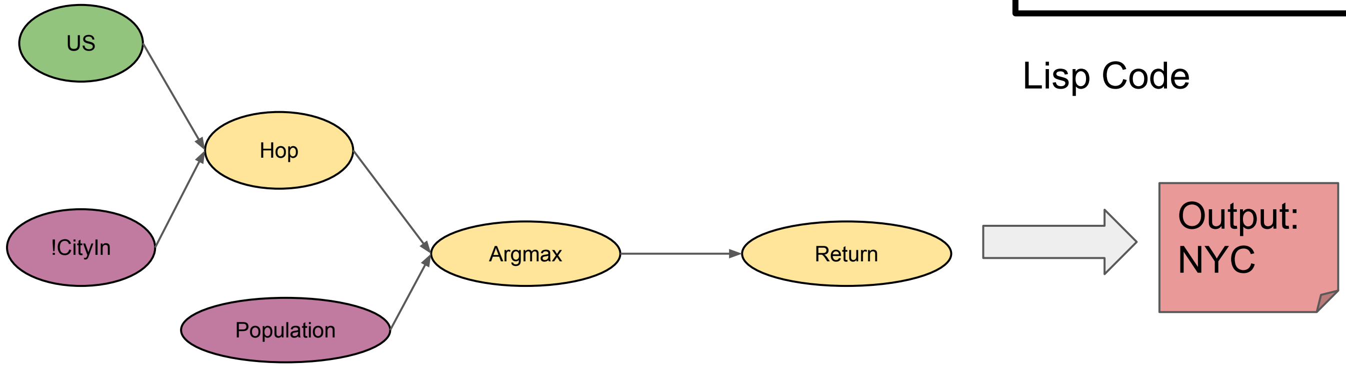
# Program as a sequence of tokens

Question: Largest city in US

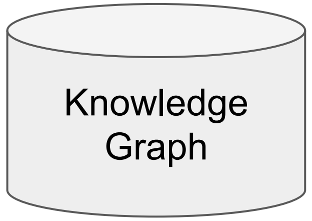
Seq2Seq

```
(define v0 US)
(define v1 (Hop v0 ?CityIn))
(define v2 (Argmax v1 Population))
(return v2)
```

Lisp Code



| Entities | Relations | Functions |
|---|---|---|
| US | CityInCountry | Hop |
| Obama | BeersFrom | ArgMax |
| …… | …… | …… |



# Semantic Parsing with NSM

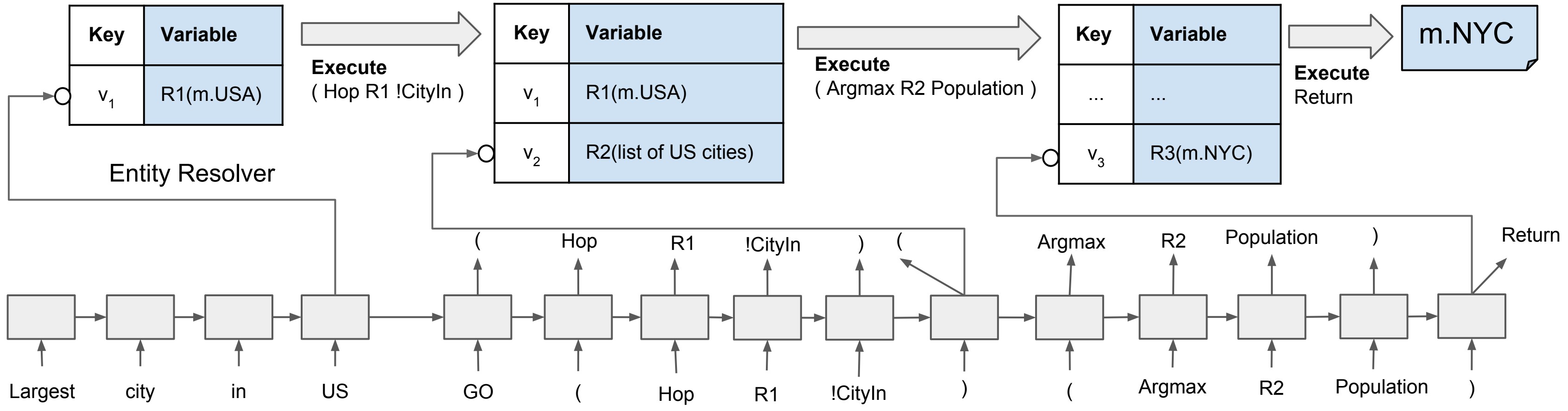
- Add a **key-variable memory** to Seq2Seq model for compositionality
- The **'keys'** are the output of GRUs
- The **'variables'** are just symbols referencing results in computer: 'R1', 'R2'

**Encoder Vocab**

| | |
|---|---|
| **Words** | who<br>largest<br>…… |

**Merged Decoder Vocab**

| | |
|---|---|
| **Functions** | Hop<br>…… |
| **Predicates** | CityInCountry<br>…… |
| **Variables** | R1<br>…… |
| **Specials** | GO<br>…… |



# Key-Variable Memory

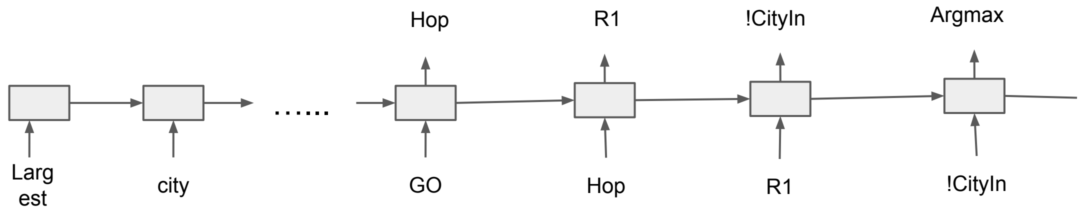
- The memory is 'symbolic'
  - Variables are symbols referencing intermediate results in computer
  - No need to have embeddings for hundreds of millions of entities in KG
  - Keys are differentiable, but variables are not

- Human use names/comments to index intermediate results

| Comments | Variable |
|---|---|
| Entity extracted from the word after “city in” | R1(m.USA) |
| Generated by querying v1 with !CityIn | R2(a list of US cities) |

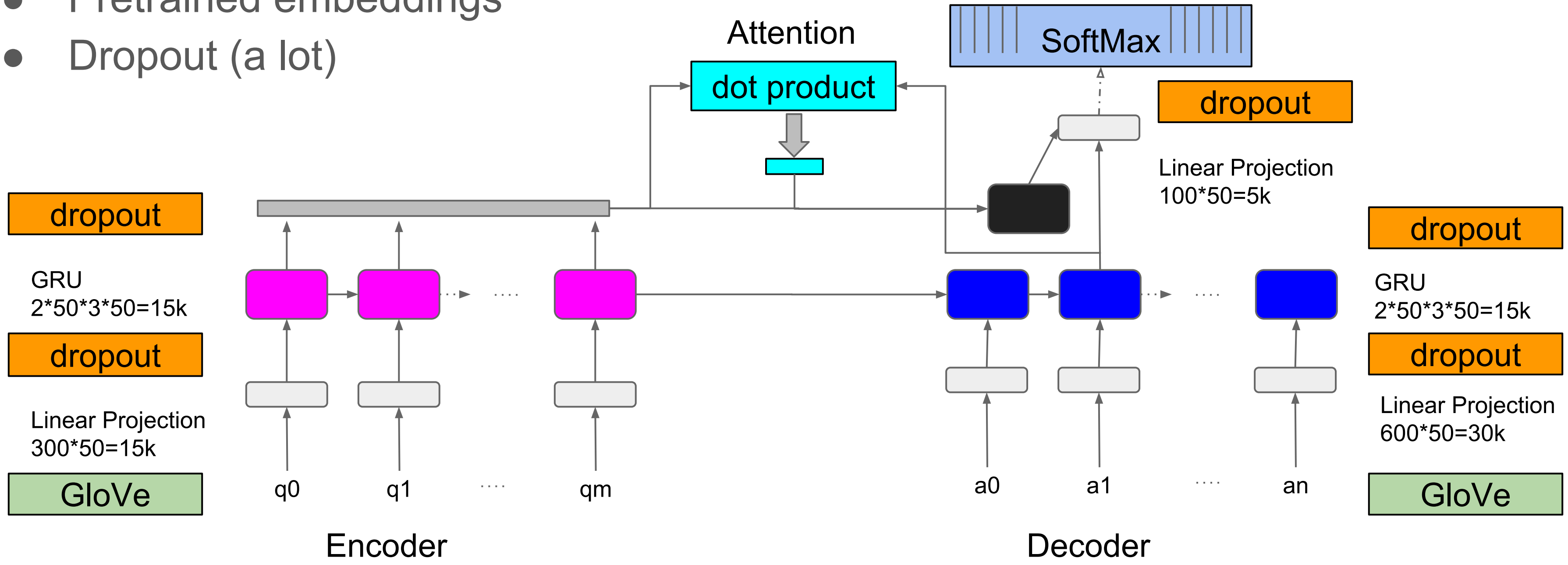
- NN use embeddings (outputs of GRUs) to index results

| Embeddings | Variable |
|---|---|
| [0.1, -0.2, 0.3, …] | R1(m.USA) |
| [0.8, 0.5, -0.3, …] | R2(a list of US cities) |

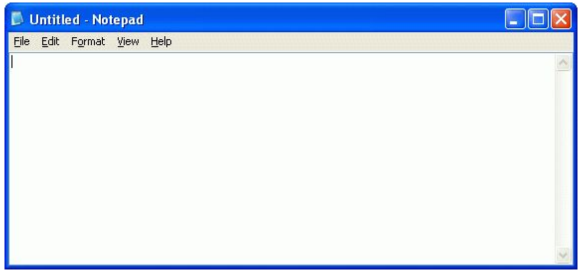


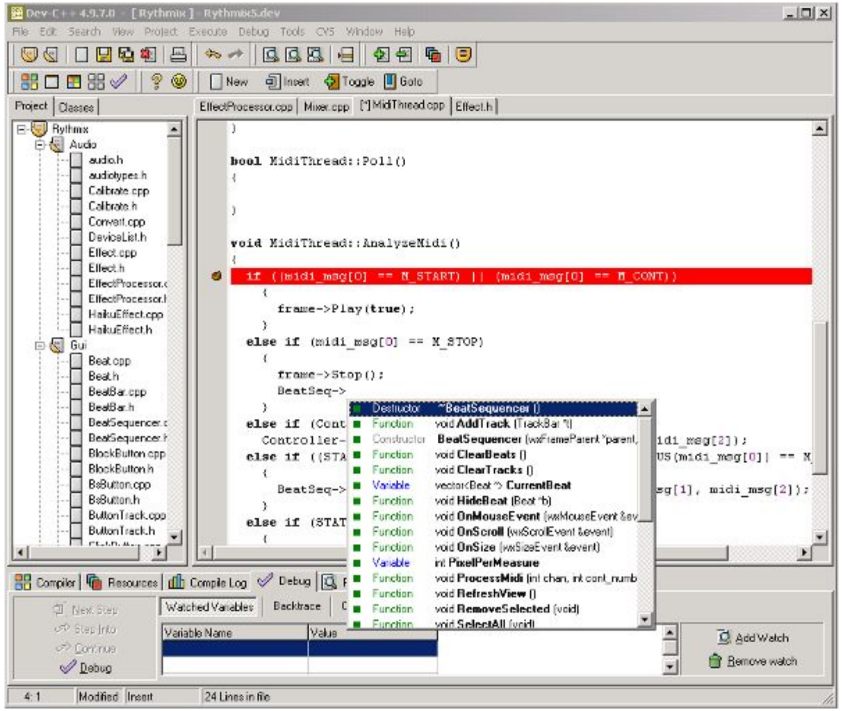
# Model Architecture

- Small model: 15k+30k+15k*2+5k = 80k params
- Dot product attention
- Pretrained embeddings
- Dropout (a lot)



# Give the NN a better coding interface





```
}

bool MidiThread::Poll()
{

}

void MidiThread::AnalyzeMidi()
{
  if ((midi_msg[0] == M_START) || (midi_msg[0] == M_CONT))
    {
      frame->Play(true);
    }
  else if (midi_msg[0] == M_STOP)
    {
      frame->Stop();
      BeatSeq->
    }
  else if (Cont
    Controller-                                  idi_msg[2]);
  else if ((STA                                  US(midi_msg[0]) == M
    {
      BeatSeq->                                  sg[1], midi_msg[2]);
    }
  else if (STAT
    {
```

| | | |
|---|---|---|
| ■ | Destructor | ~BeatSequencer () |
| ■ | Function | void AddTrack (TrackBar *t) |
| ■ | Constructor | BeatSequencer (wxFrameParent *parent, |
| ■ | Function | void ClearBeats () |
| ■ | Function | void ClearTracks () |
| ■ | Variable | vector<Beat *> CurrentBeat |
| ■ | Function | void HideBeat (Beat *b) |
| ■ | Function | void OnMouseEvent (wxMouseEvent &ev |
| ■ | Function | void OnScroll (wxScrollEvent &event) |
| ■ | Function | void OnSize (wxSizeEvent &event) |
| ■ | Variable | int PixelPerMeasure |
| ■ | Function | void ProcessMidi (int chan, int cont_numb |
| ■ | Function | void RefreshView () |
| ■ | Function | void RemoveSelected (void) |
| ■ | Function | void SelectAll (void) |

**Code Assist**

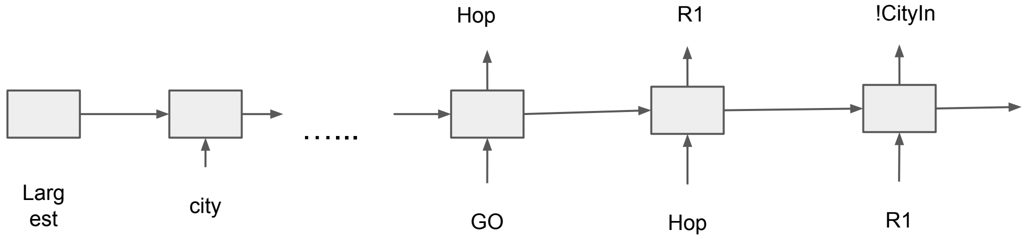
## Neural Computer Interface

- A Strong IDE / Interpreter helps reduce the search space
    - Exclude the invalid choices that will cause syntax and semantic error

Syntax check: Only a variable can follow 'Hop'

Semantic check: only relations that are connected to R1 can be used **~ 20k => ~ 100**

Implemented as a changing mask on decoding vocabulary



## Non-differentiable => REINFORCE Training

- Optimizing expected F1

$$J^{RL}(\theta) = \sum_q \mathbb{E}_{P(a_{0:T}|q,\theta)}[R(q, a_{0:T})]$$

- Use baseline B(q) to reduces variance without changing the optima

$$\nabla_\theta J^{RL}(\theta) = \sum_q \sum_{a_{0:T}} P(a_{0:T}|q,\theta)[R(q, a_{0:T}) - B(q)]\nabla_\theta \log P(a_{0:T}|q,\theta)$$

$$B(q) = \sum_{a_{0:T}} P(a_{0:T}|q,\theta) R(q, a_{0:T})$$

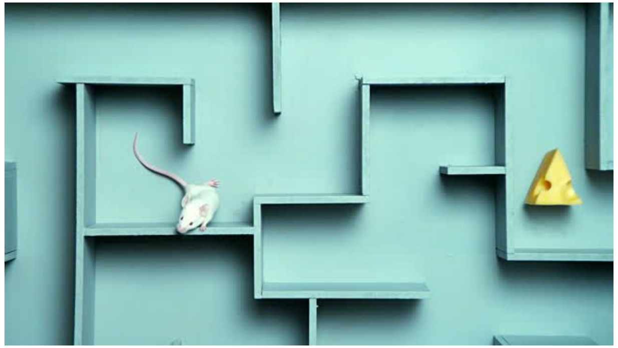
- Gradient computation is approximated by beam search instead of sampling

# Sampling v.s. Beam search

- Decoding uses beam search
    - Use top k in beam ( normalized probabilities) to compute gradients
    - Reduce variance and estimate the baseline better

- The coding environment is deterministic. Closer to a maze than Atari game.

Stochastic


Deterministic

## Problem with REINFORCE

Training is slow and get stuck on local optimum

- Large search space
  - model probability of good programs with non-zero F1 is very small
- Large beam size
  - Normalized probability small
  - Decoding and training is slow because larger number of sequences
- Small beam size
  - Good programs might fall off the beam

**Solution:**
Add some gold programs into the beam with reasonably large probability... but we **don't** have gold programs, only **weak supervision**

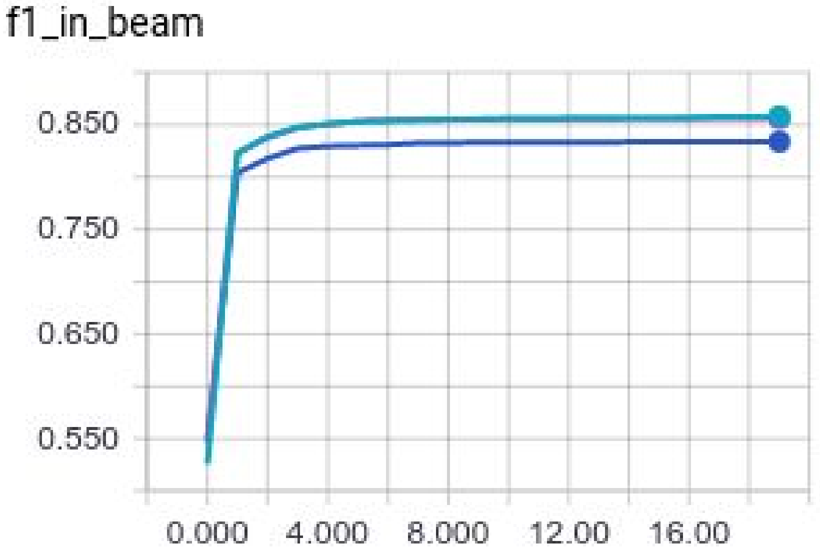
# Finding Approximate Gold Programs

- Ideally we want to do supervised pretraining for REINFORCE, but we only have weak supervision
- Use an iterative process interleaving decoding with large beam and maximum likelihood training

- Training objective:

$$J^{ML}(\theta) = \sum_{q} \log P(a_{0:T}^{best}(q) | q, \theta)$$

- Training is fast and has a bootstrap effect

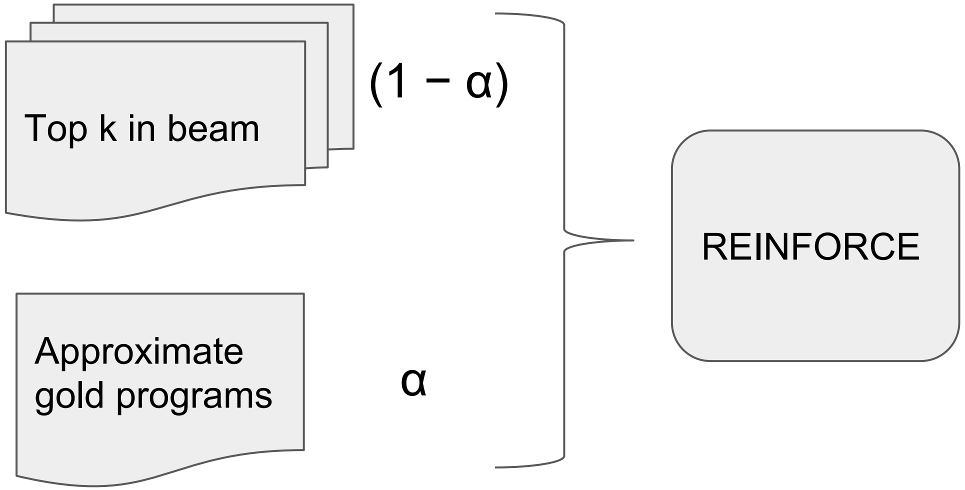


## Drawbacks of the ML objective

- Not directly optimizing expected F1
- The best program for a question could be a spurious program that accidentally produced the correct answer, and thus does not generalize to other questions
  - e.g., answering PLACEOFBIRTH with PLACEOFDEATH
- Because training lacks explicit negative examples, the model fails to distinguish between tokens that are related to one another
  - e.g., PARENTSOF vs. SIBLINGSOF vs. CHILDRENOF

# Augmented REINFORCE

- Add the approximate gold program into the final beam with probability α, and the probabilities of the original programs in the beam are normalized to be (1 − α).
- The rest of the process is the same as in standard REINFORCE



## Overview

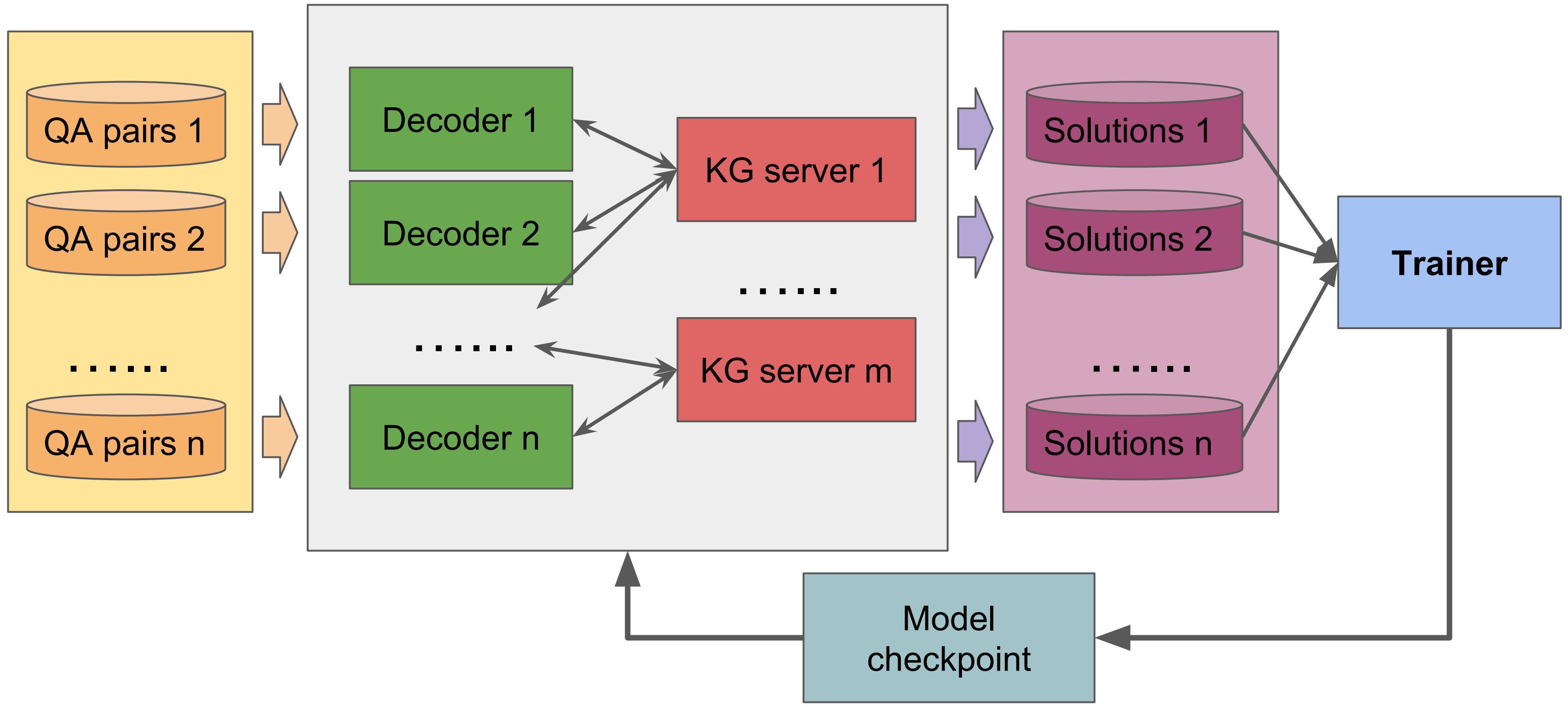
- Semantic parsing: (updated) WebQuestions dataset
- Neural program induction
- Manager-Programmer-Computer (MPC) framework
- Neural Symbolic Machine
- Experiments and analysis

## Freebase Preprocessing

- Remove predicates which are not related to world knowledge
  - Those starting with "/common/", "/type/", "/freebase/"
- Remove all text valued predicates
  - They are almost never the answer of questions
- Result in a graph which is small enough to fit in memory
  - #Relations=23K
  - #Nodes=82M
  - #Edges=417M

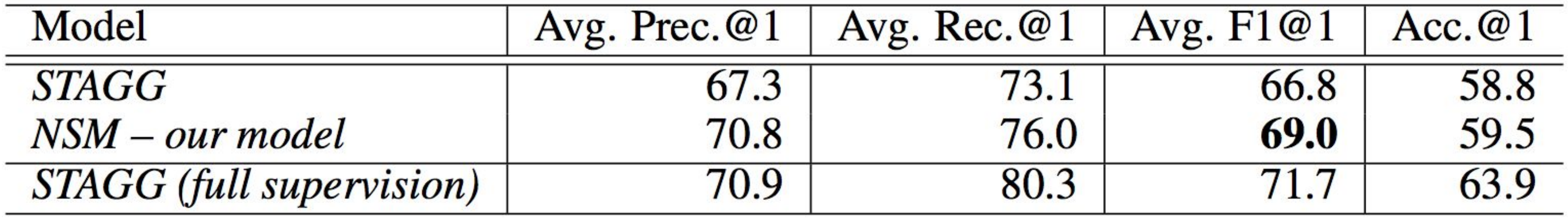
## System Architecture

- 200 decoders, 50 KG servers, 1 trainer, 251 machines in total
- Since the network is small, we didn't see much speedup from GPU



## Compare to State-of-the-Art

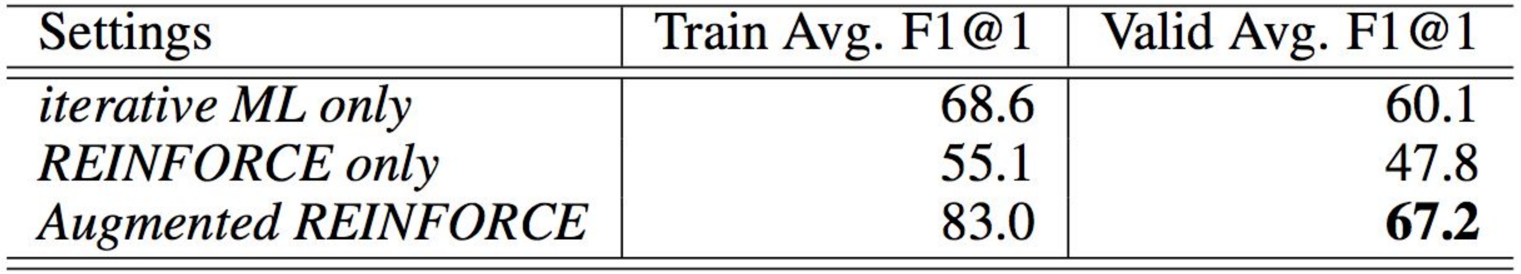
- First end-to-end neural network to achieve state-of-the-art performance on semantic parsing with weak supervision over large knowledge base
- The performance is approaching state-of-the-art result with full supervision



| Model | Avg. Prec.@1 | Avg. Rec.@1 | Avg. F1@1 | Acc.@1 |
|---|---|---|---|---|
| *STAGG* | 67.3 | 73.1 | 66.8 | 58.8 |
| *NSM – our model* | 70.8 | 76.0 | **69.0** | 59.5 |
| *STAGG (full supervision)* | 70.9 | 80.3 | 71.7 | 63.9 |

## Augmented REINFORCE

- REINFORCE get stuck at local maxima
- Iterative ML training is not directly optimizing the F1 measure
- Augmented REINFORCE obtains the best performances

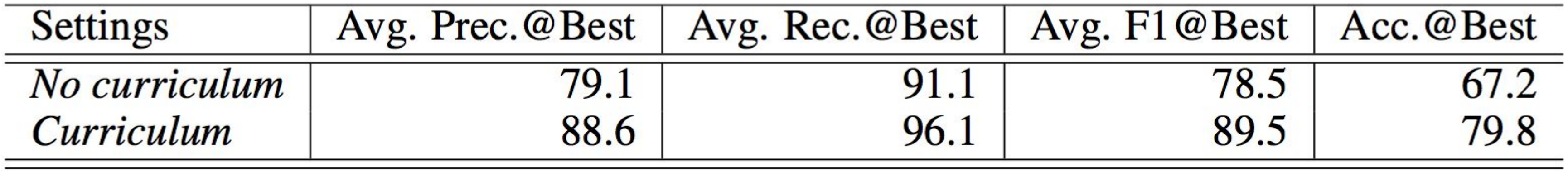


| Settings | Train Avg. F1@1 | Valid Avg. F1@1 |
|---|---|---|
| *iterative ML only* | 68.6 | 60.1 |
| *REINFORCE only* | 55.1 | 47.8 |
| *Augmented REINFORCE* | 83.0 | **67.2** |

Inspired by STAGG [Yih, et al 2016]

## Curriculum Learning

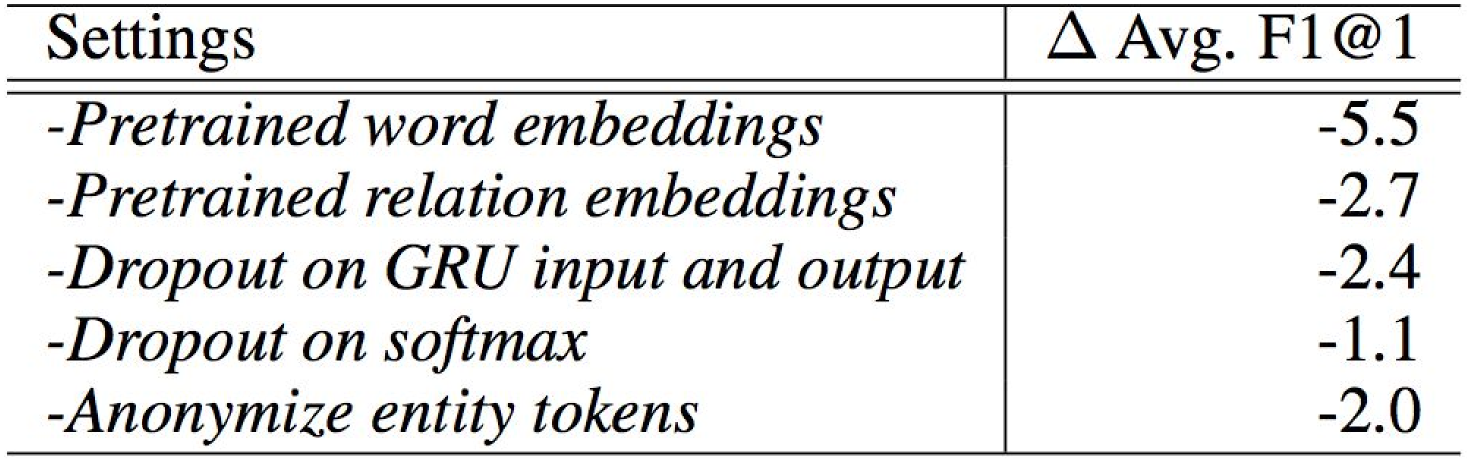
- Gradually increasing the program complexity during ML training
  - First run iterative ML training with only the "Hop" function and the maximum number of expressions is 2
  - Then run iterative ML training again with all functions, and the maximum number of expressions is 3. The relations used by the "Hop" function are restricted to those that appeared in the best programs from in first one
- A lot of search failures without curriculum learning



| Settings | Avg. Prec.@Best | Avg. Rec.@Best | Avg. F1@Best | Acc.@Best |
|---|---|---|---|---|
| *No curriculum* | 79.1 | 91.1 | 78.5 | 67.2 |
| *Curriculum* | 88.6 | 96.1 | 89.5 | 79.8 |

## Reduce Overfitting

- With all these techniques the model is still overfitting
  - Training F1@1 = 83.0%
  - Validation F1@1 = 67.2%

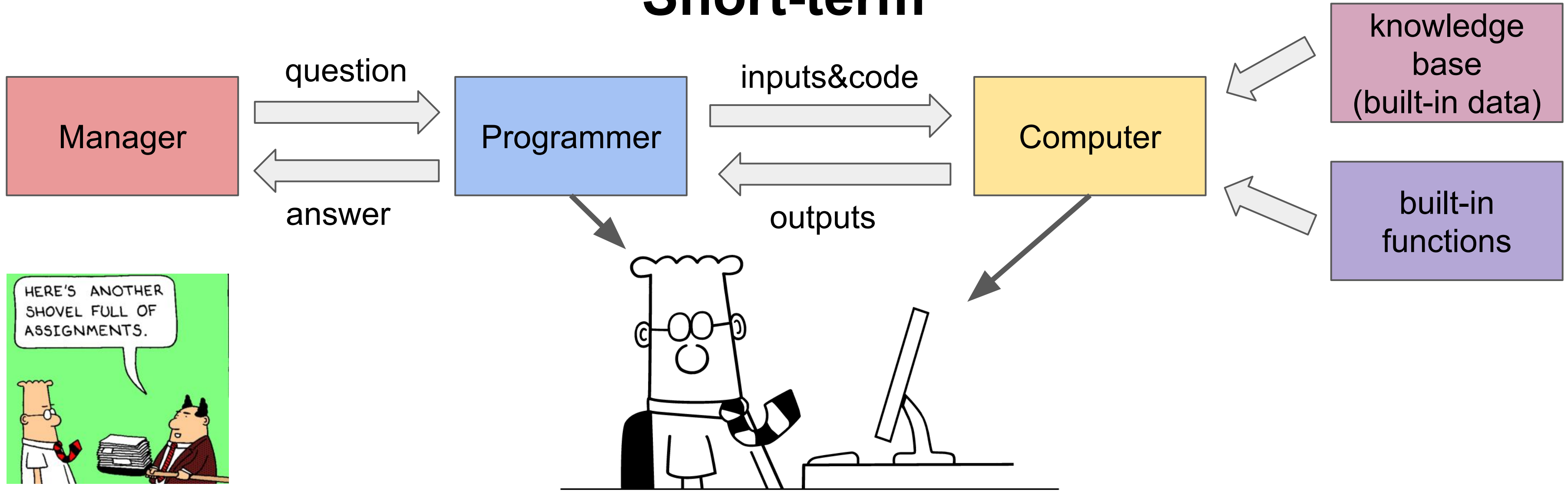


| Settings | Δ Avg. F1@1 |
|---|---|
| *-Pretrained word embeddings* | -5.5 |
| *-Pretrained relation embeddings* | -2.7 |
| *-Dropout on GRU input and output* | -2.4 |
| *-Dropout on softmax* | -1.1 |
| *-Anonymize entity tokens* | -2.0 |

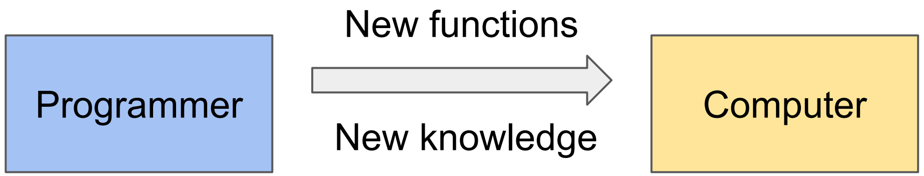
## Future work

- Better performance with more training data
- Actions to add knowledge into KG and create new schema
- Language to action

## Short-term



## Long-term



## Acknowledgement

- Thanks for discussions and helps from Arvind, Mohammad, Tom, Eugene, Lukasz, Thomas, Yonghui, Zhifeng, Alexandre, John

- Thanks for MSR researchers, who made WebQuestionSP data set available