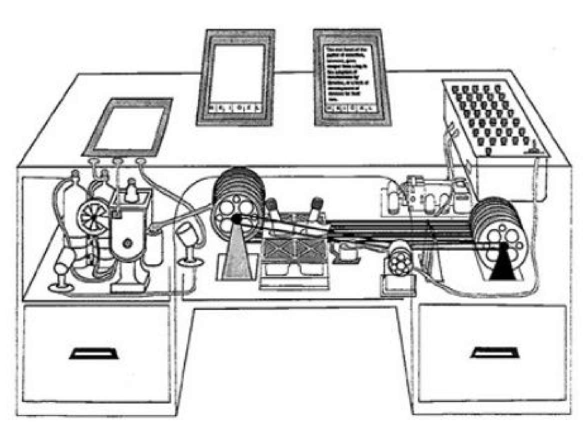
Learning to Organize Knowledge with N-Gram Machines

Fan Yang, Jiazhong Nie, William W. Cohen, Ni Lao
Carnegie Mellon University & Google
{fanyang1,wcohen}@cs.cmu.edu, {niejiazhong,nlao}@google.com

How should information be organized?

- Task-oriented, model free, and life-long learning.
- Support fast retrieval and reasoning.

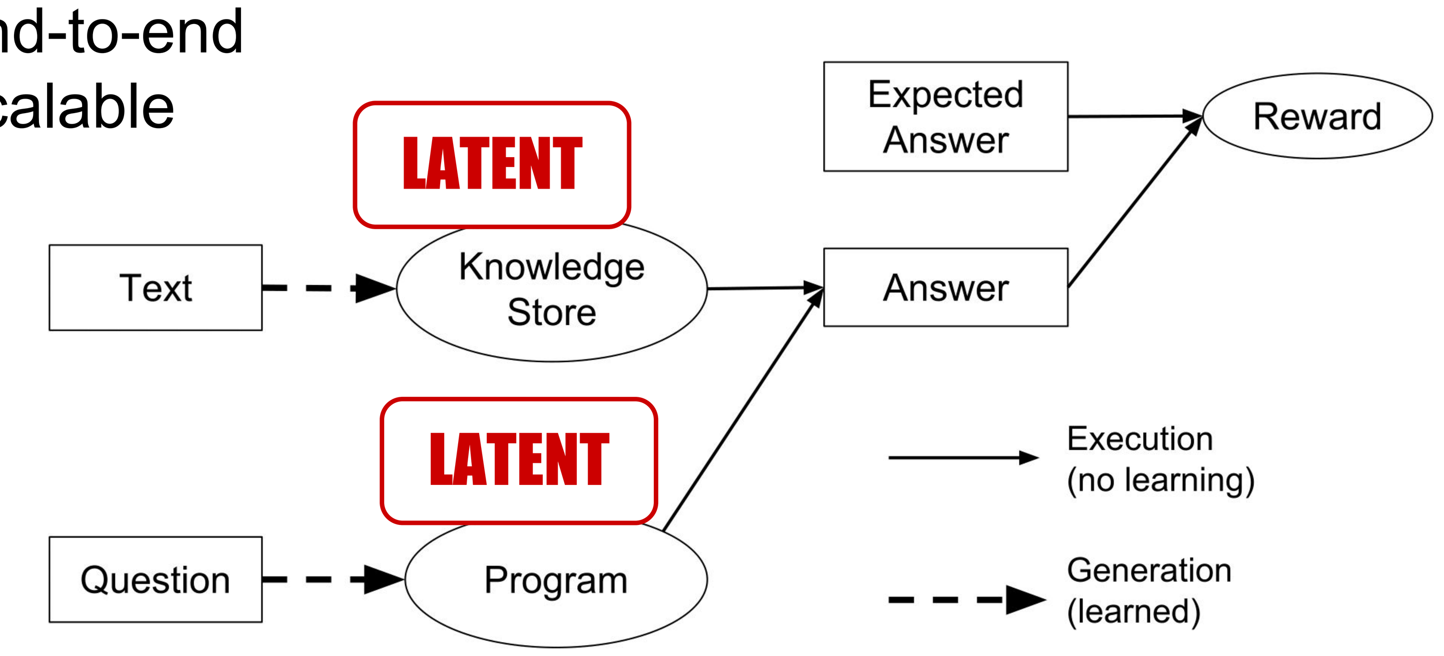
"AS WE MAY THINK"
(1945)



"Consider a future device for individual use, which is a sort of mechanized private file and library. It needs a name, and to coin one at random, memex will do. A memex is a device in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory.

Question answering as a test bed

- End-to-end
- Scalable



NGM Framework

- Probabilistic Knowledge storage

$$P(\mathbf{\Gamma}|\mathbf{s};\theta_{\text{enc}}) = \Pi_{\Gamma_i \in \mathbf{\Gamma}} P(\Gamma_i|s_i, s_{i-1};\theta_{\text{enc}})$$

Table 1: Example of probabilistic knowledge storage. Each sentence may be converted to a distribution over multiple tuples, but only the one with the highest probability is shown here.

Sentences	Knowledge tuples		
	Time stamp	Symbols	Probability
Mary went to the kitchen.	1	`mary to kitchen`	0.9
Mary picked up the milk.	2	`mary the milk`	0.4
John went to the bedroom.	3	`john to bedroom`	0.7
Mary journeyed to the garden.	4	`mary to garden`	0.8

- Programs

Table 2: Functions in N-Gram Machines. The knowledge storage on which the programs can execute is $\mathbf{\Gamma}$, and a knowledge tuple Γ_i is represented as $(i, (\gamma_1, \ldots, \gamma_N))$. "FR" means *from right*.

Name	Inputs	Return
`Hop`	$v_1 \ldots v_L$	$\{\gamma_{L+1} \mid \text{ if } (\gamma_1 \ldots \gamma_L) == (v_1, \ldots, v_L), \forall \Gamma \in \mathbf{\Gamma}\}$
`HopFR`	$v_1 \ldots v_L$	$\{\gamma_{N-L} \mid \text{ if } (\gamma_{N-L+1} \ldots \gamma_N) == (v_L, \ldots, v_1), \forall \Gamma \in \mathbf{\Gamma}\}$
`Argmax`	$v_1 \ldots v_L$	$\text{argmax}_i\{(\gamma_{L+1}, i) \mid \text{ if } (\gamma_1 \ldots \gamma_L) == (v_1, \ldots, v_L), \forall \Gamma_i \in \mathbf{\Gamma}\}$
`ArgmaxFR`	$v_1 \ldots v_L$	$\text{argmax}_i\{(\gamma_{N-L}, i) \mid \text{ if } (\gamma_{N-L+1} \ldots \gamma_N) == (v_L, \ldots, v_1), \forall \Gamma_i \in \mathbf{\Gamma}\}$

- Seq2Seq models
  - Knowledge encoder $P(\Gamma_i|s_i, s_{i-1};\theta_{\text{enc}})$ — s_{i-1} for co-references
  - Knowledge decoder $P(s_i|\Gamma_i, s_{i-1};\theta_{\text{dec}})$
  - Programmer $P(C|q, \mathbf{\Gamma};\theta_{\text{prog}})$ — Γ for code assist

Experiment Results

- Extractive bAbI tasks

Table 3: Test accuracy on bAbI tasks with auto-encoding (AE) and structure tweak (ST)

	Task 1	Task 2	Task 11	Task 15	Task 16
MemN2N	1.000	0.830	0.840	1.000	0.440
QA	0.007	0.027	0.000	0.000	0.098
QA + AE	0.709	0.551	**1.000**	0.246	**1.000**
QA + AE + ST	**1.000**	**0.853**	**1.000**	**1.000**	**1.000**

- Auto-encoding and structural tweaking help to learn good representations.

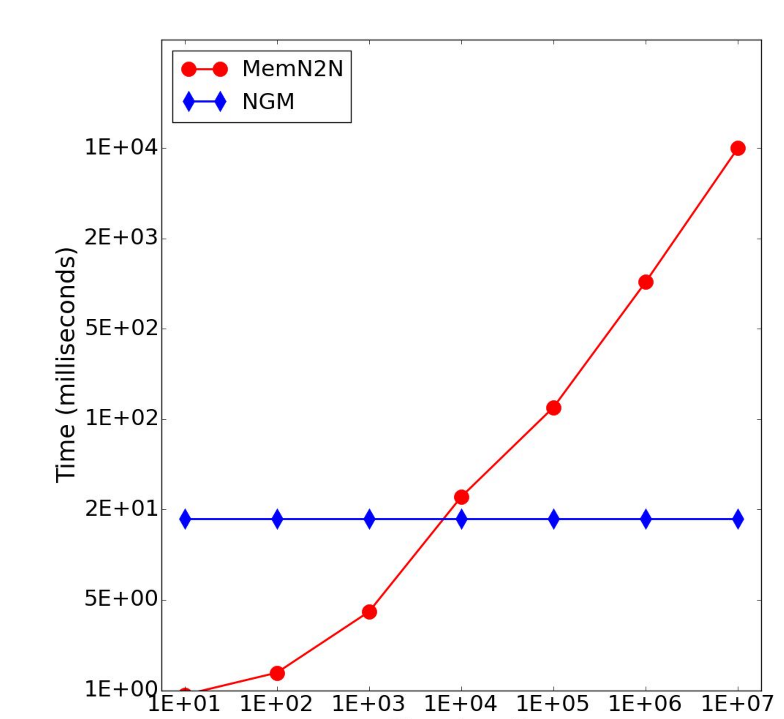
QA	QA + AE	QA + AE + ST
`went went went`	`daniel went office`	`daniel went office`
`mary mary mary`	`mary back garden`	`mary went garden`
`john john john`	`john back kitchen`	`john went kitchen`
`mary mary mary`	`mary grabbed football`	`mary got football`
`there there there`	`sandra got apple`	`sandra got apple`
`cats cats cats`	`cats afraid wolves`	`cat afraid wolves`
`mice mice mice`	`mice afraid wolves`	`mouse afraid wolves`
`is is cat`	`gertrude is cat`	`gertrude is cat`

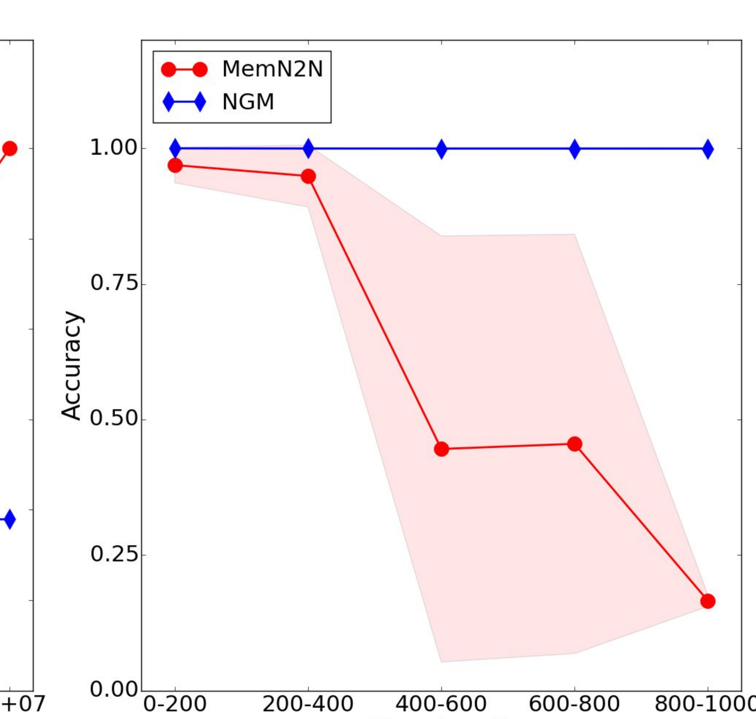
- Example solution

Table 6: Task 2 Two Supporting Facts

Story	Knowledge Storage
`Sandra journeyed to the hallway.`	`Sandra journeyed hallway`
`John journeyed to the bathroom.`	`John journeyed bathroom`
`Sandra grabbed the football.`	`Sandra got football`
`Daniel travelled to the bedroom.`	`Daniel journeyed bedroom`
`John got the milk.`	`John got milk`
`John dropped the milk.`	`John got milk`
Question	**Program**
`Where is the milk?`	`ArgmaxFR milk got` `Argmax V1 journeyed`

- Constant inference time





Inference

- Beam search instead of MCMC to avoid huge variances .
- To solve a hard search problem:
  - Stabilized auto-encoding (AE)
  - Structure tweak (ST)

Optimization

- Coordinate descent by REINFORCEs

$$O^{QA}(\theta_{\text{enc}}, \theta_{\text{prog}}) = \sum_{\mathbf{\Gamma}} \sum_{C} P(\mathbf{\Gamma}|\mathbf{s};\theta_{\text{enc}}) P(C|q, \mathbf{\Gamma};\theta_{\text{prog}}) R(\mathbf{\Gamma}, C, a),$$

$$O^{\text{AE}}(\theta_{\text{enc}}, \theta_{\text{dec}}) = \mathbb{E}_{p(z|x;\theta_{\text{enc}})}[\log p(x|z;\theta_{\text{dec}})] + \sum_{z \in \mathbf{Z}^N(x)} \log p(x|z;\theta_{\text{dec}}),$$

$Z^N(x)$: all tuples of length N which only consist of words from x