Learning to Disambiguate Natural Language Using World Knowledge

Antoine Bordes
antoine.bordes@lip6.fr
Nicolas Usunier
nicolas.usunier@lip6.fr

LIP6 - Université Paris 6
Paris, France

&

Jason Weston
jaseweston@gmail.com
Ronan Collobert
ronan@collobert.com

Google Labs, New York, USA
NEC Labs, Princeton, USA

Connect Natural Language to the World

- *Strong prior knowledge:* *We* understand language because it has a deep connection to the world it is used in/for.

- **Our goal:** learning from scratch to use both syntax and the surrounding environment to "understand" natural language.

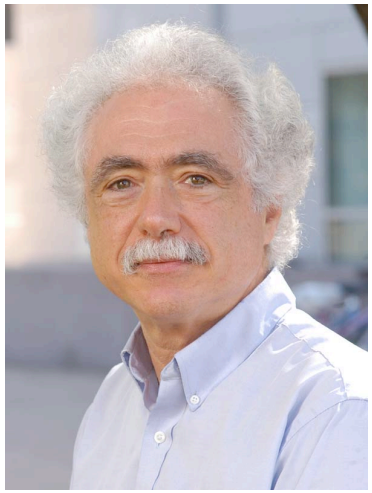
"John saw Bill in the park with his telescope."
"He passed the exam."
"John went to the bank."

World knowledge we might already have:
Bill owns a telescope.
Fred took an exam last week.
John is in the countryside (not the city).

An Old Idea ...

"When a human reader sees a sentence, he uses knowledge to understand it. This includes not only grammar, but also his knowledge about words, the context of the sentence, and most important, his knowledge about the subject matter.

A computer program supplied with only grammar for manipulating the syntax of language could not produce a translation of reasonable quality."



Terry Winograd – 1971

in *Procedures as a Representation for Data in a Computer program for Understand Natural Language*

...with Modern Applications

Multiplayer online games = world knowledge + natural language.

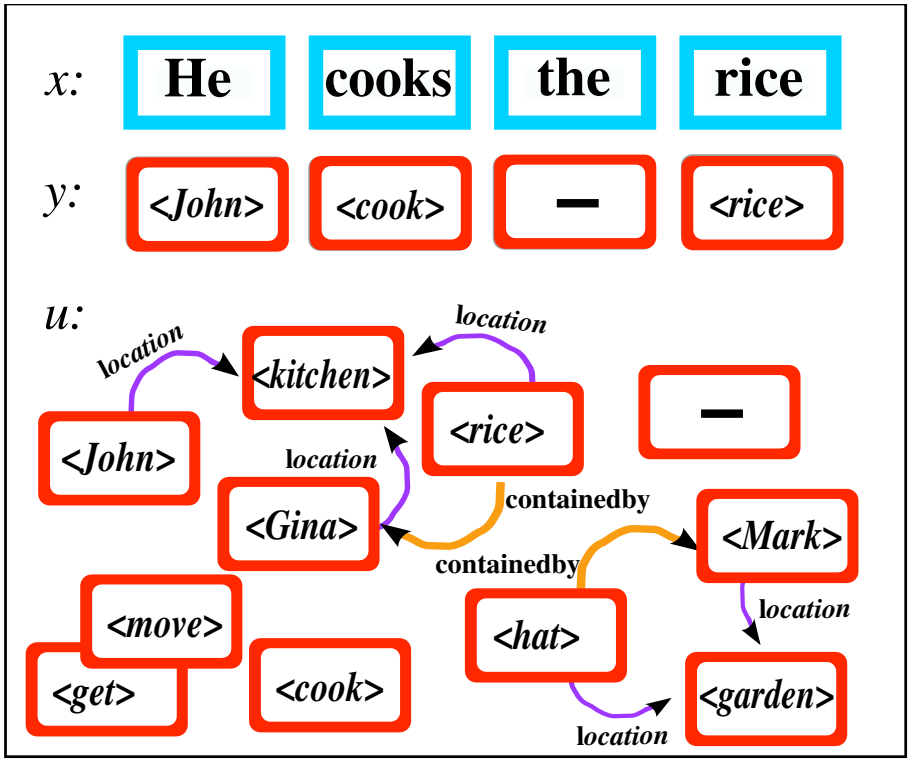
Part I

Concept Labeling

The Concept Labeling Task

Definition: Map any natural language sentence x to its labeling in terms of *concepts* y, using the current state the world u.

$u =$ "universe", set of concepts and their relations to other concepts.

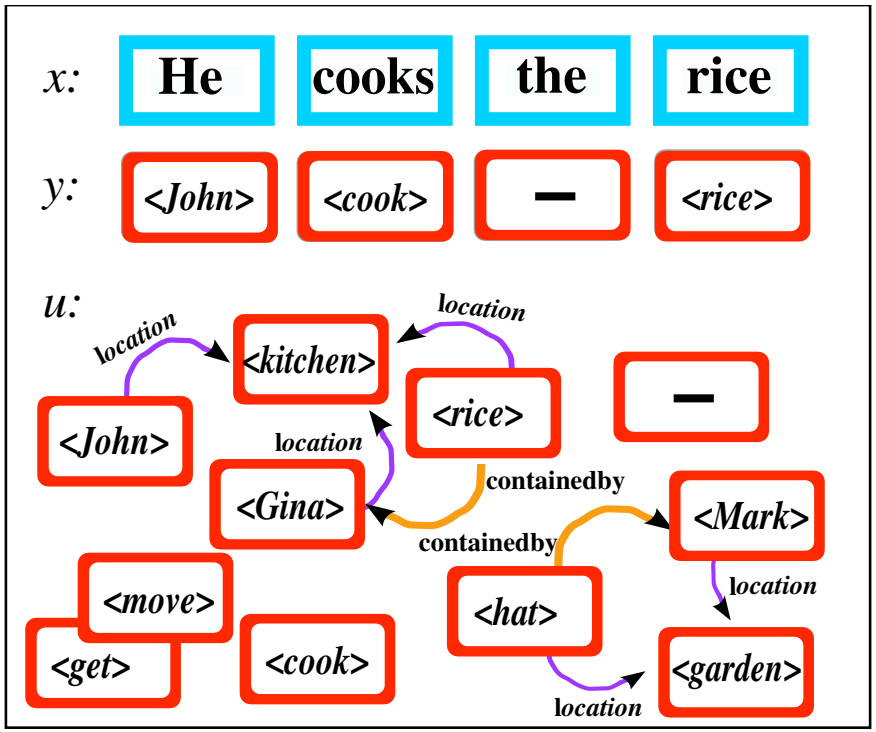


Supervised Learning

Training triples $(x, y, u) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{U}$ with two kinds of supervision:

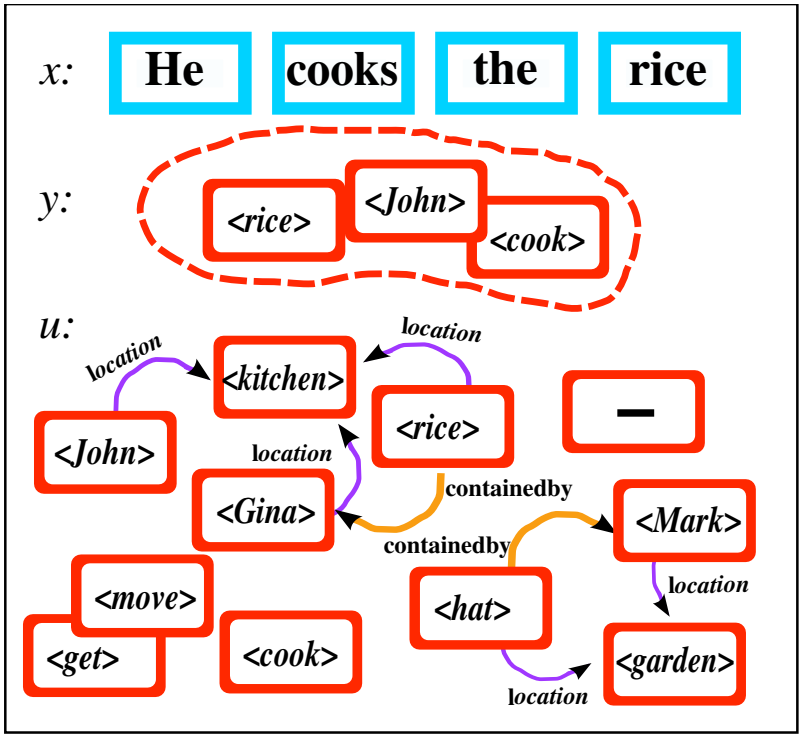
STRONG

hard & costly to gather data



WEAK

more realistic setting



Ambiguities

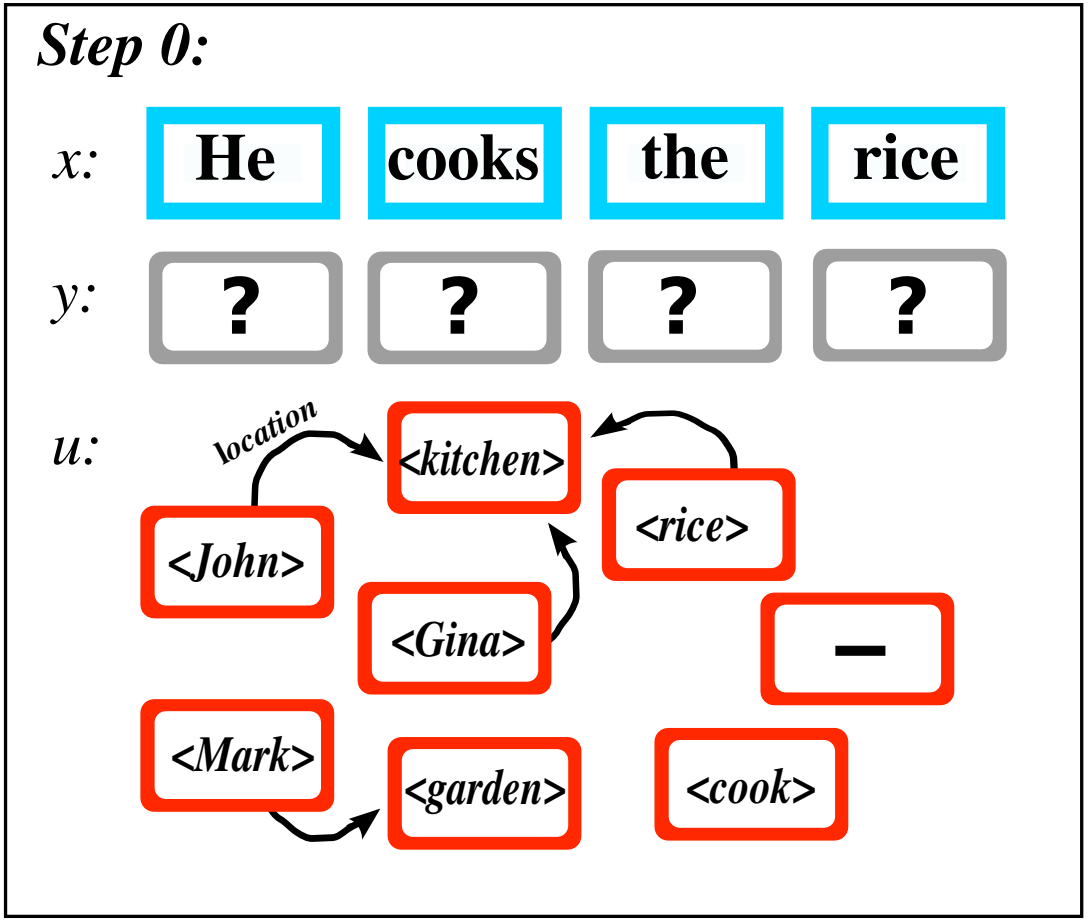
The main difficulty of concept labeling $\rightarrow$ **ambiguous words**.

He picked up the hat **there**.
The **milk** on the table.
The **one** on the table.
She left the kitchen.
The adult left the kitchen.
Mark drinks the **orange**.
...

(e.g. for sentence (2) there may be several milk cartons that exist...)

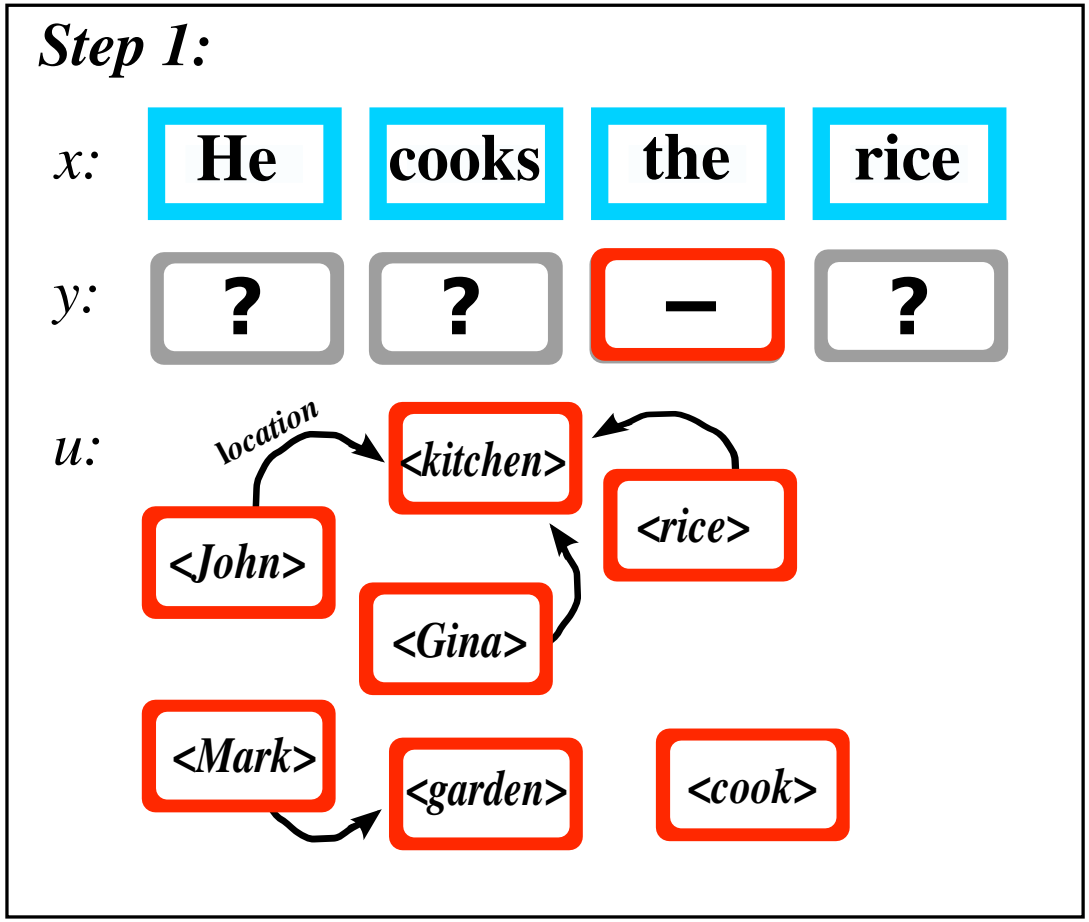
Mix of word sense disambiguation, reference resolution and entity recognition.

Disambiguation Example



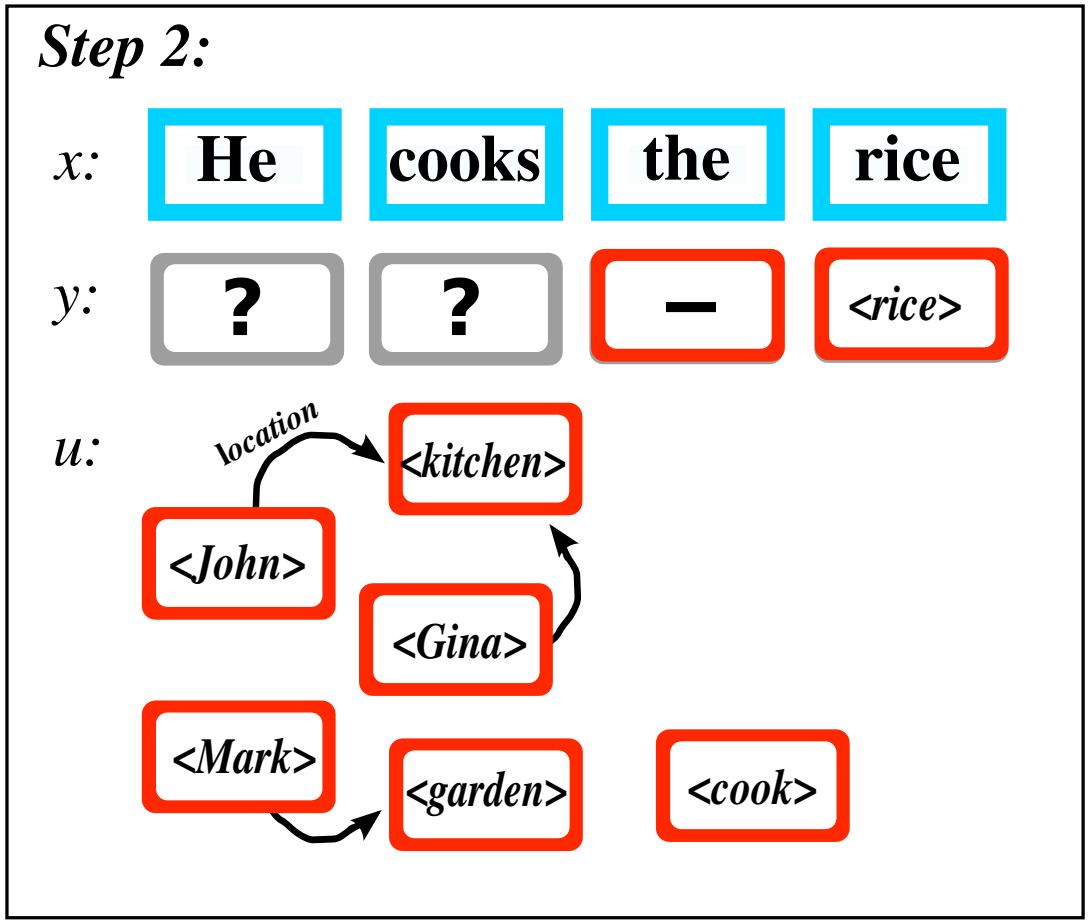
Label the above sentence.

Disambiguation Example



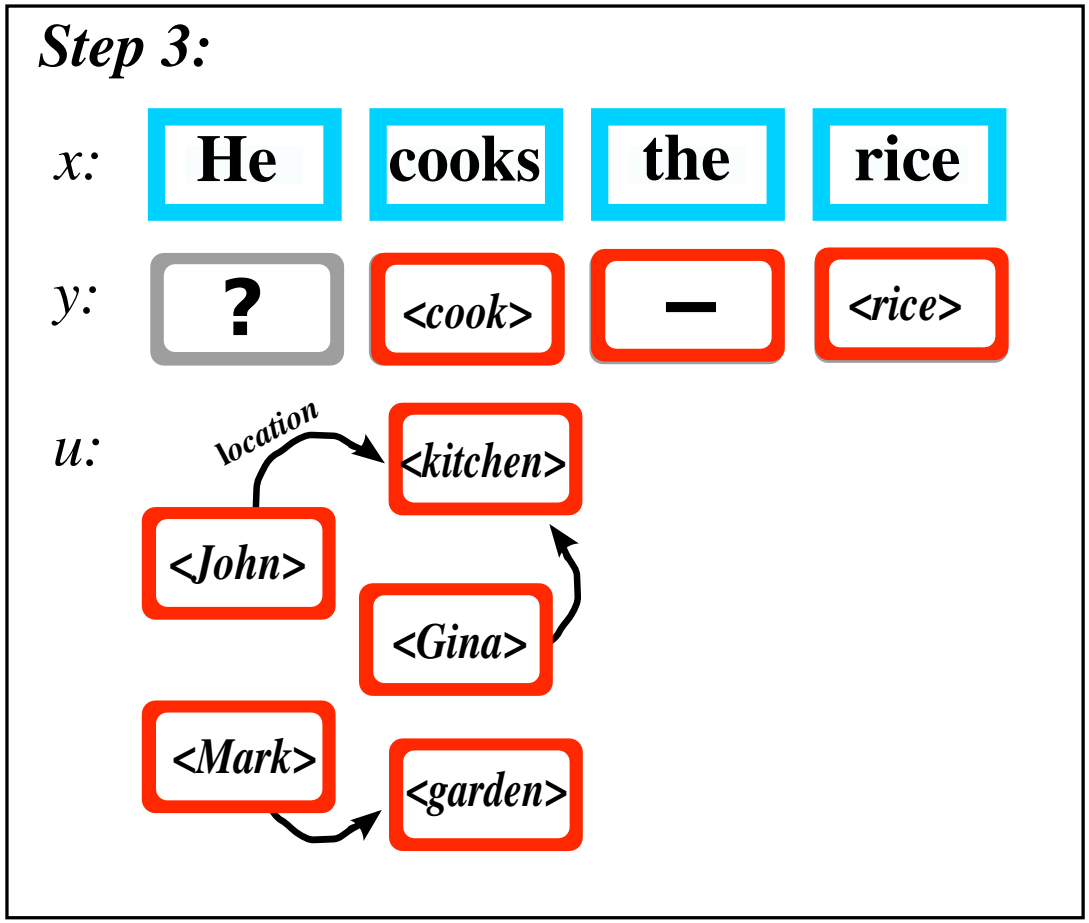
You have to start by labeling non-ambiguous words.

Disambiguation Example



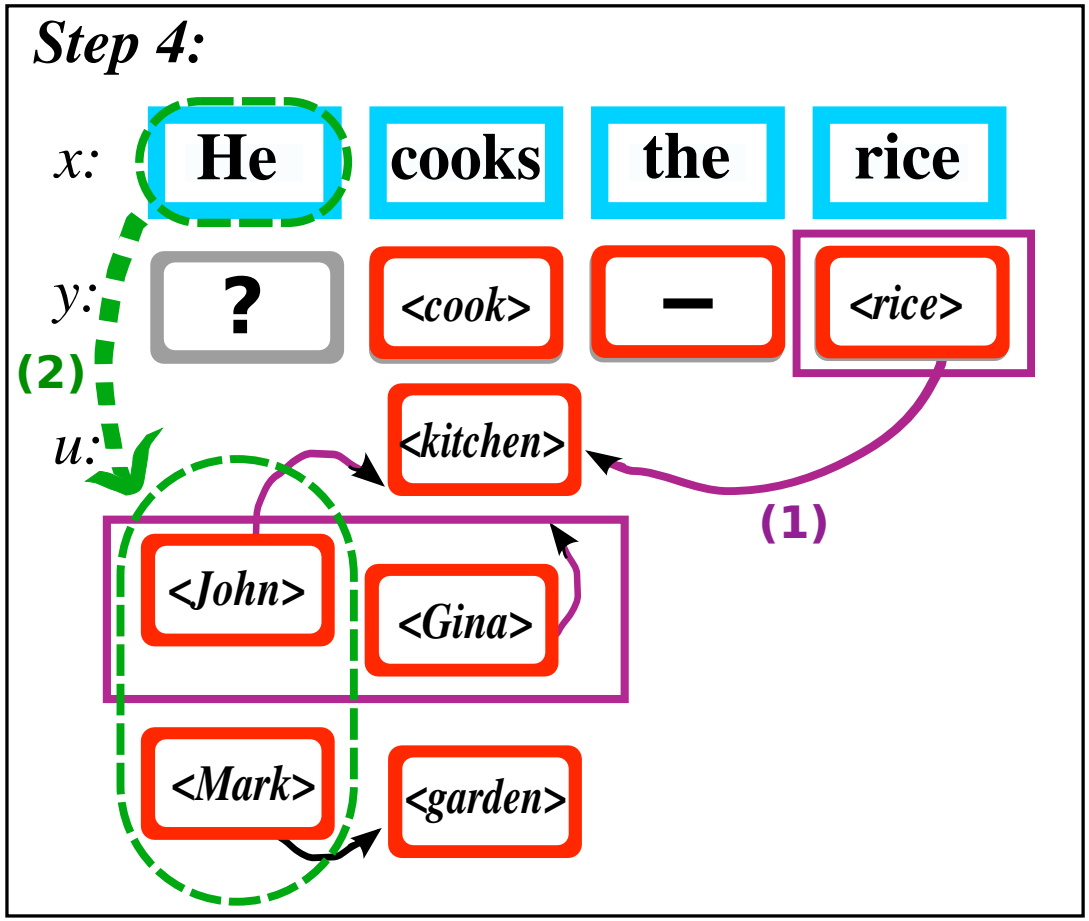
Again...

Disambiguation Example



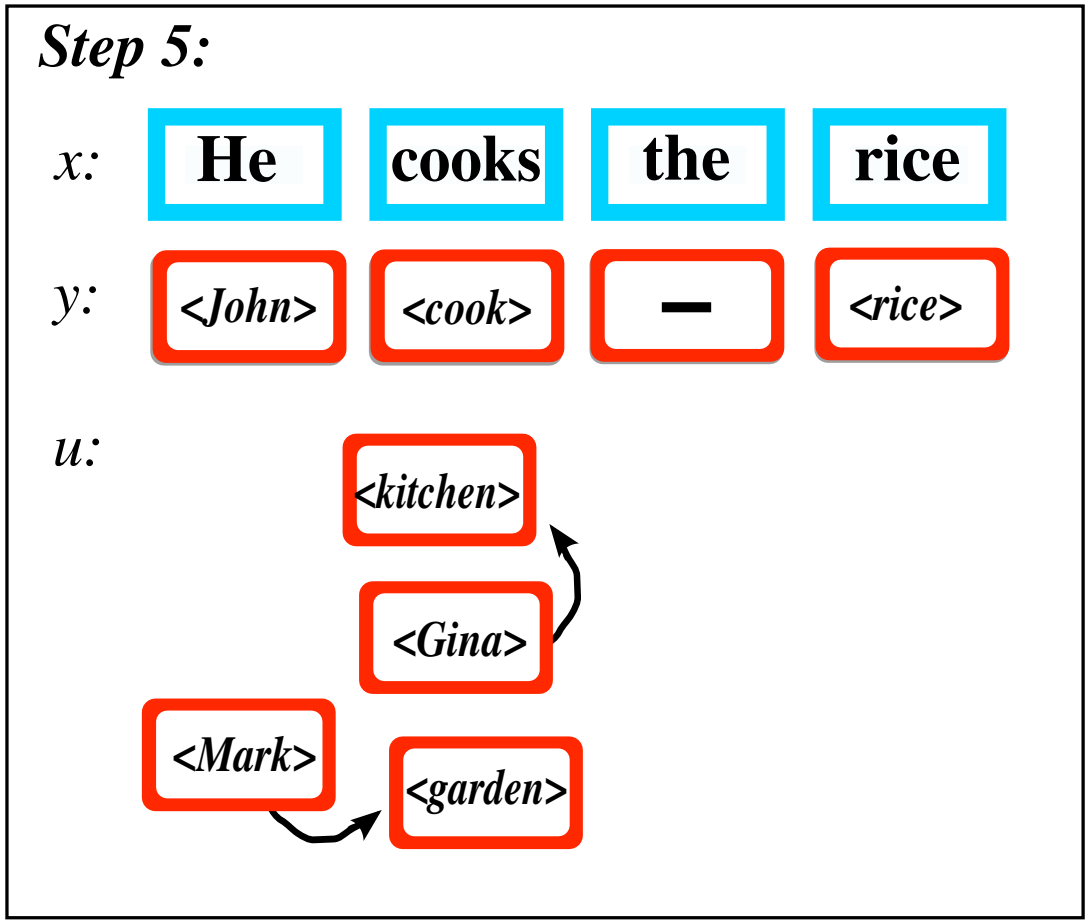
Here is a problem.

Disambiguation Example



Label "He" requires two rules which are never explicitly given.

Disambiguation Example



"John" is the only male in the kitchen!

Concept Labeling is Challenging

- Solving ambiguities requires to use rules based on linguistic information and available universe knowledge.
- **But** these rules are never made explicit in training.

→ A concept labeling algorithm has to learn them.

- No engineered features for describing words/concepts are given.

→ A concept labeling algorithm has to discover them from raw data.

Part II

Learning Algorithm

Global Structured Inference

We use a matching score :

$$\hat{y} = f(x, u) = \text{argmax}_{y'} \quad g(x, y', u),$$

$g(\cdot)$ is a scoring function which should be large if concepts y' are consistent with both the sentence x and the current state of the universe u.

Due to the complexity of the tagging problem, a complete argmax computation could be very slow. . .

Greedy ‘‘Order-free’’ Inference

Inference algorithm:

1. For all the positions not yet labeled, predict what the corresponding concept would be (using the scoring function).
2. Select the pair (position, concept) you are the most confident in. *(hopefully the least ambiguous)*
3. Remove this position from the set of available ones.
4. Collect all universe-based features of this concept to help label remaining ones.
5. Return to 1.

Training using a variant of LaSO [Daumé & al.,'05]

Scoring Function

Our score combines two functions $g_i(\cdot)$ and $h(\cdot) \in \mathbb{R}^N$ which are neural networks that could potentially encode similarities.

$$g(x, y, u) = \sum_{i=1}^{|x|} g_i(x, y_{-i}, u)^\top h(y_i, u)$$

- $g_i(x, y_{-i}, u)$ is a sliding-window on the text *and* neighboring concepts centered around i^{th} word $\rightarrow$ embeds to N dim-space.
- $h(y_i, u)$ embeds the i^{th} concept & its relations to N dim-space.
- Dot-product: confidence that i^{th} word labeled with concept y_i.

Encoding World knowledge

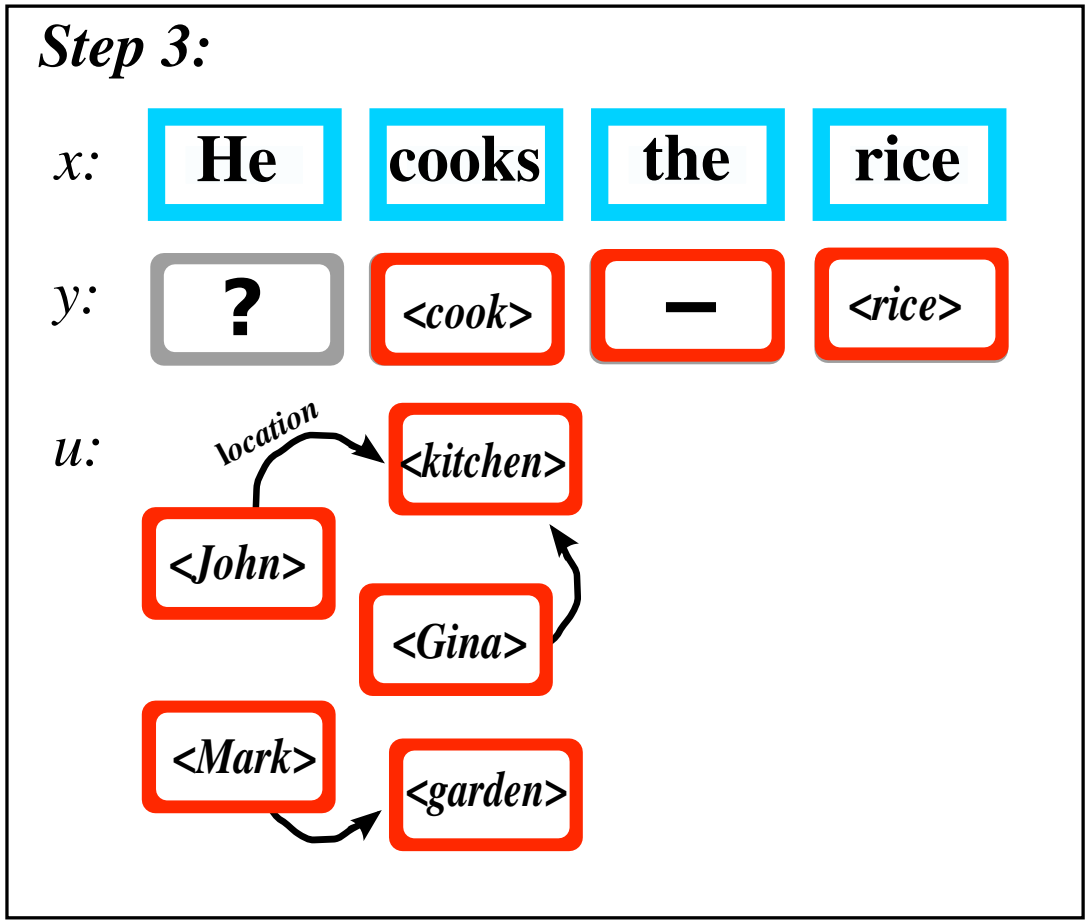
- For each concept y of the universe, we learn a vector mapping $C(y)$ of dimension d with a "Lookup Table".

- A concept and its current relations are encoded with a concatenation of such mappings.

- Examples:
  <milk>: located in *<kitchen>* & contained by *<john>*
  $\longrightarrow$ represented by $\bar{C} = (C(<\textit{milk}>),\ C(<\textit{kitchen}>),\ C(<\textit{john}>))$.
  <gina>: located in *<bedroom>*
  $\longrightarrow$ represented by $\bar{C} = (C(<\textit{gina}>),\ C(<\textit{bedroom}>),\ C(<->))$.
  <cook>:
  $\longrightarrow$ represented by $\bar{C} = (C(<\textit{cook}>),\ C(<->),\ C(<->))$.

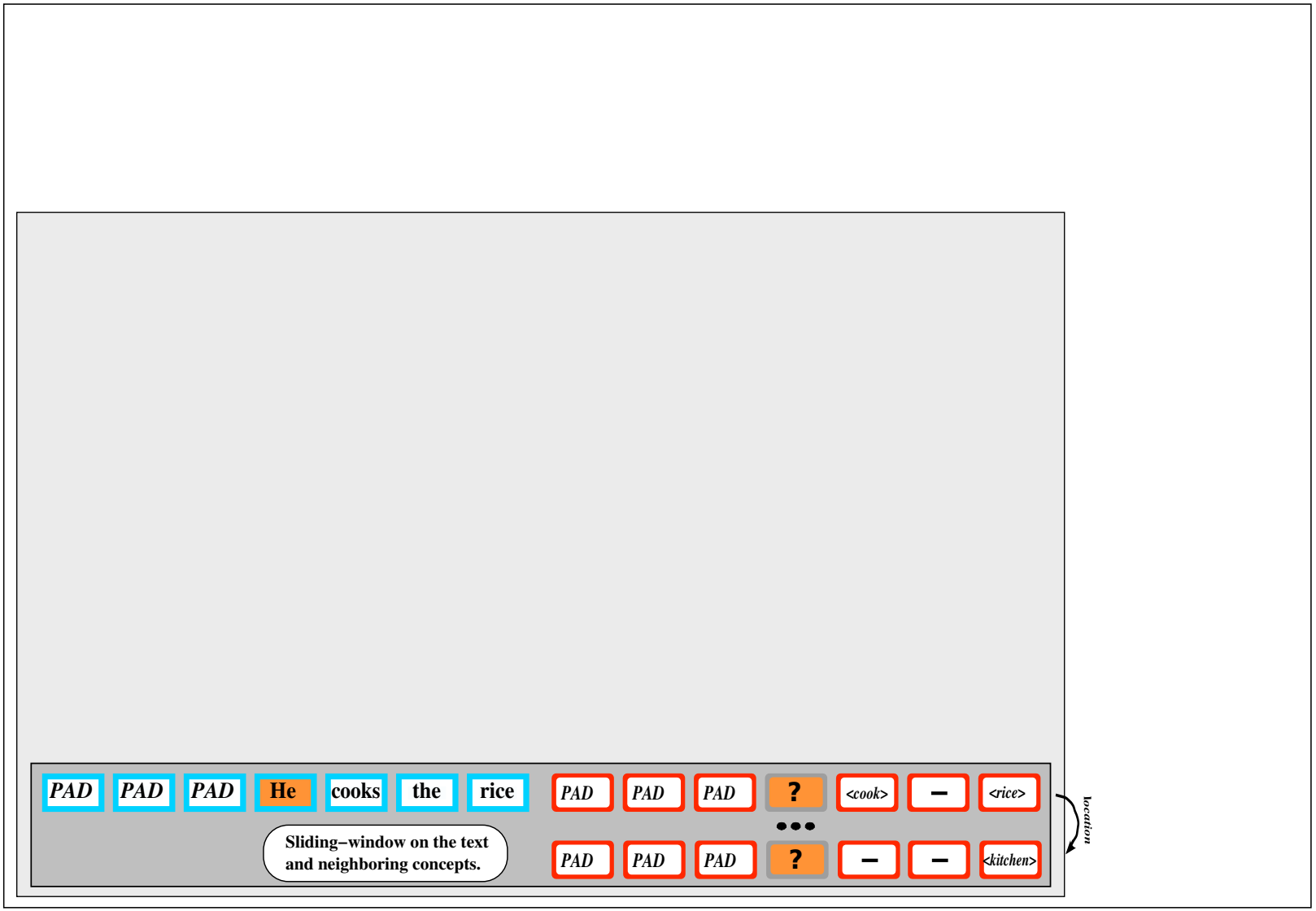
Scoring Illustration

Let's get back to our previous example:



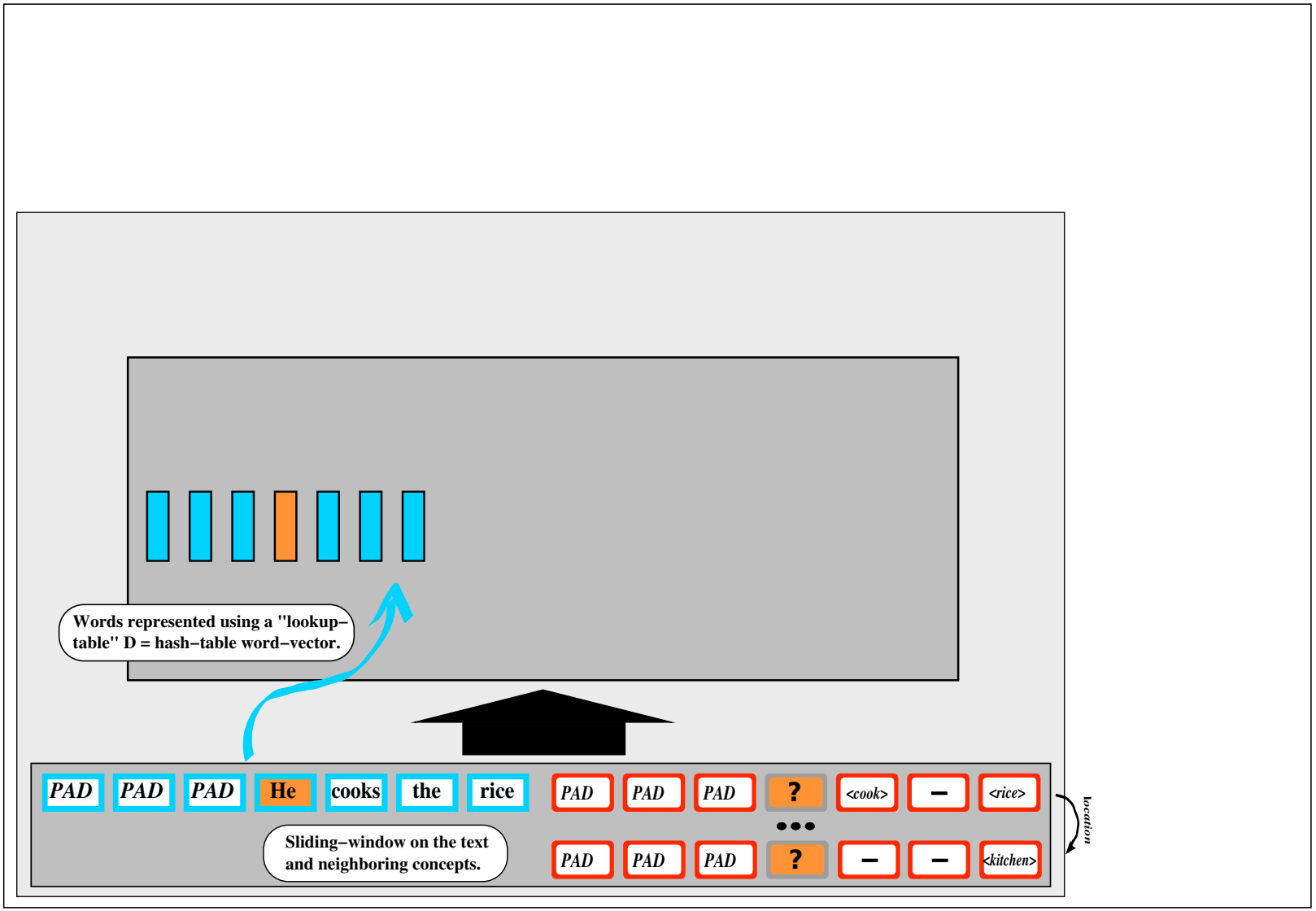
Scoring Illustration

Step 0: Set the sliding-window around the 1^{st} word.



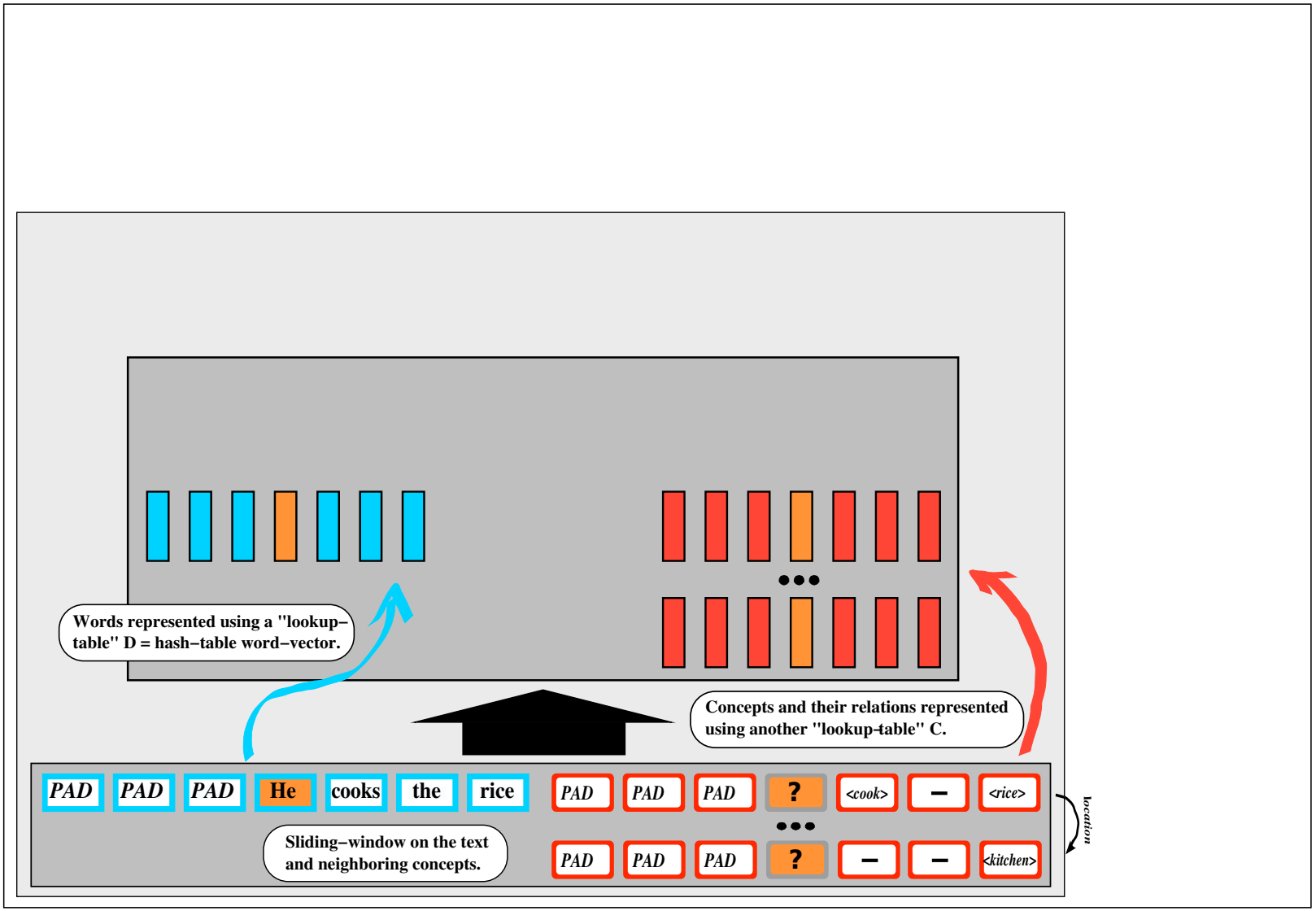
Scoring Illustration

Step 1: Retrieve words representations from the "lookup table".



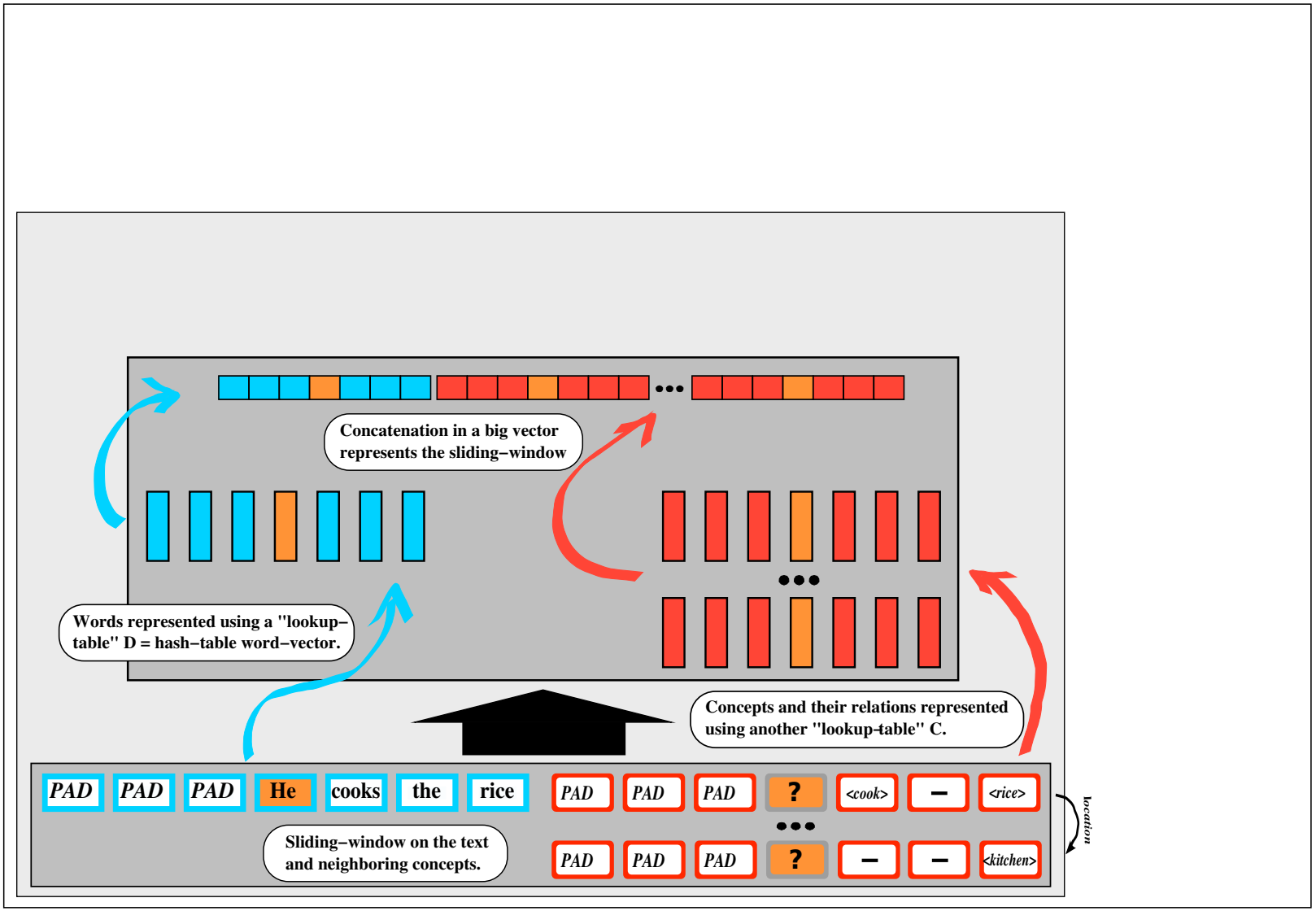
Scoring Illustration

Step 2: Similarly retrieve concepts representations.



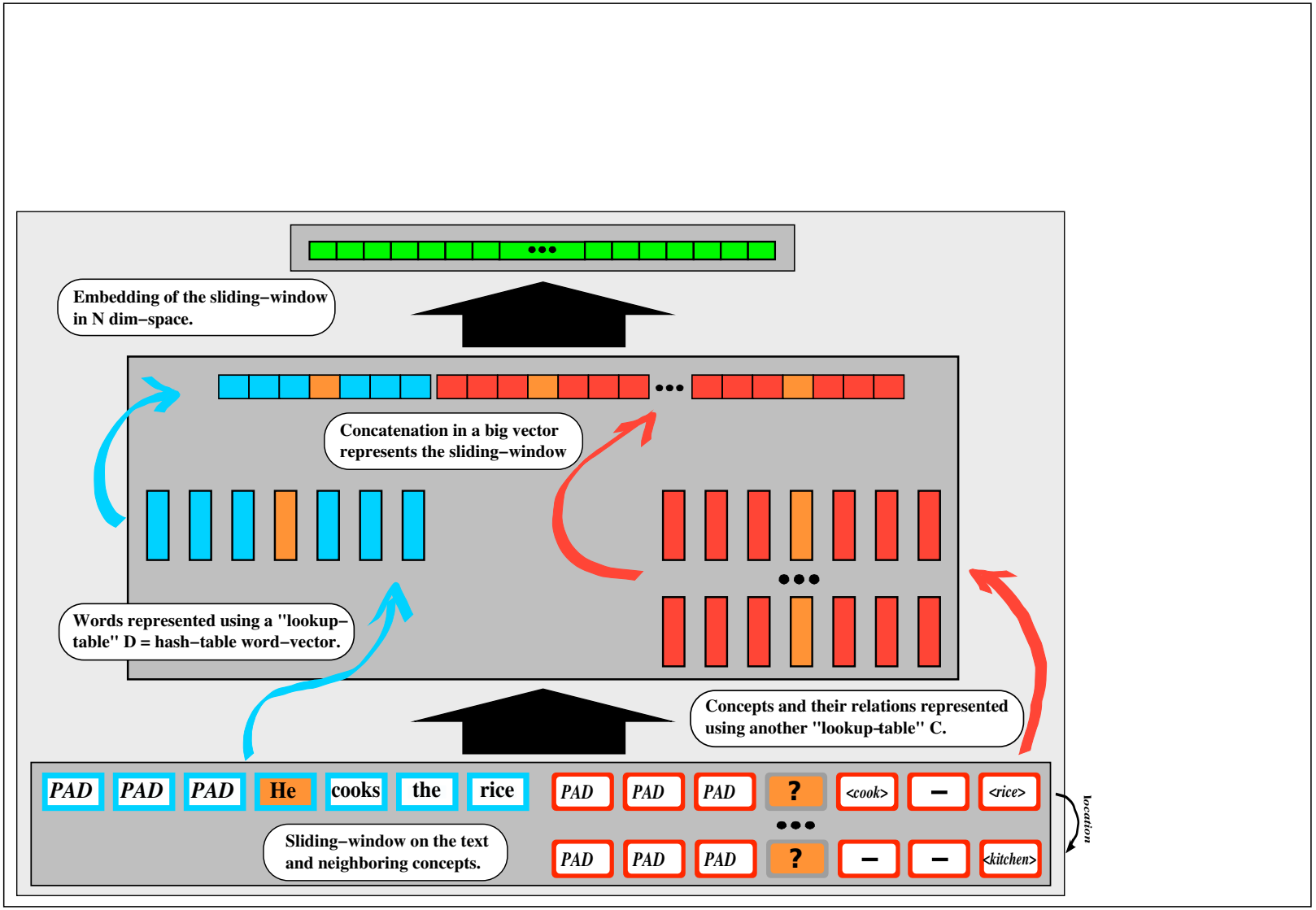
Scoring Illustration

Step 3: Concatenate vectors to obtain window representation.



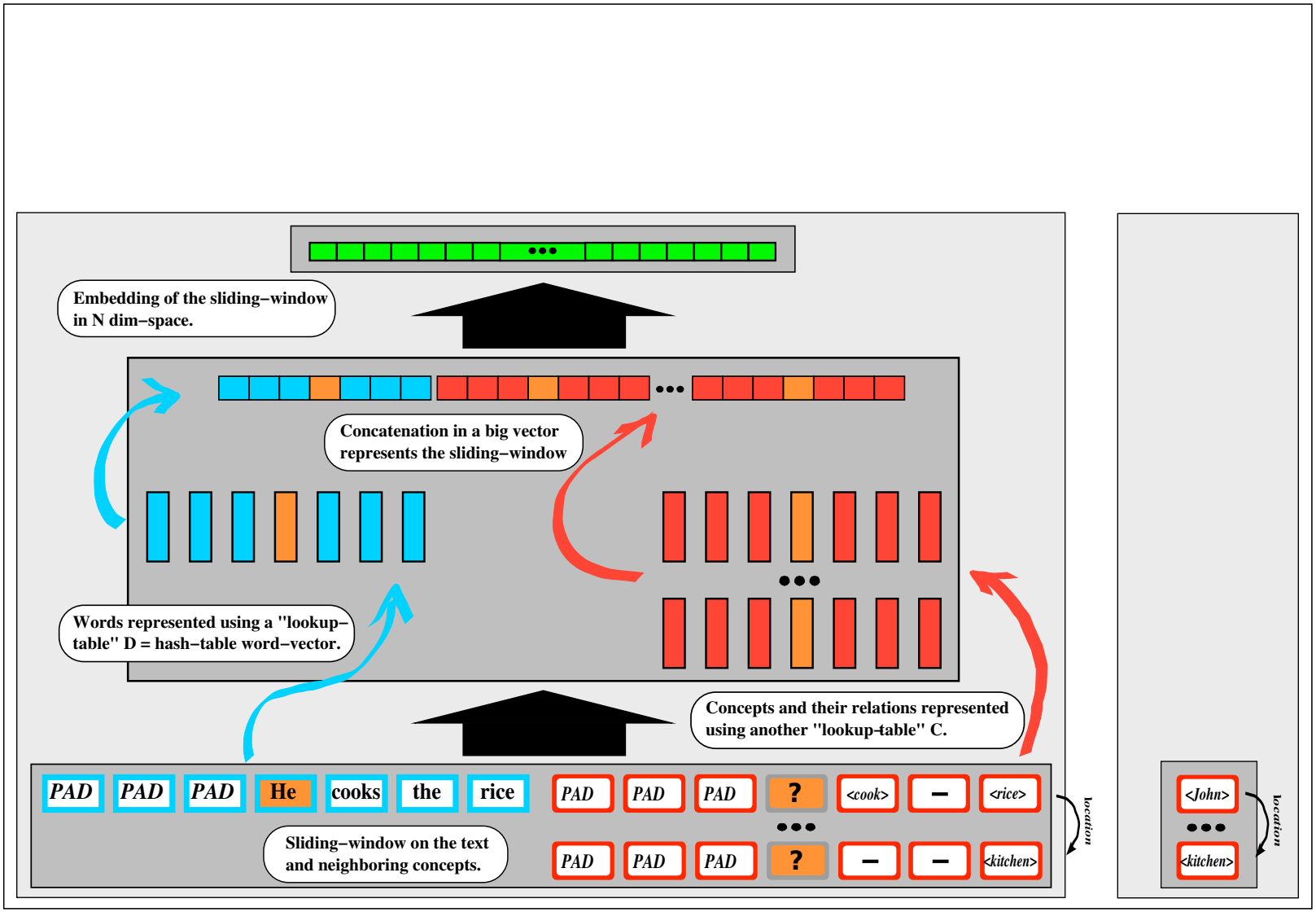
Scoring Illustration

Step 4: Compute $g_1(x, y_{-1}, u)$.



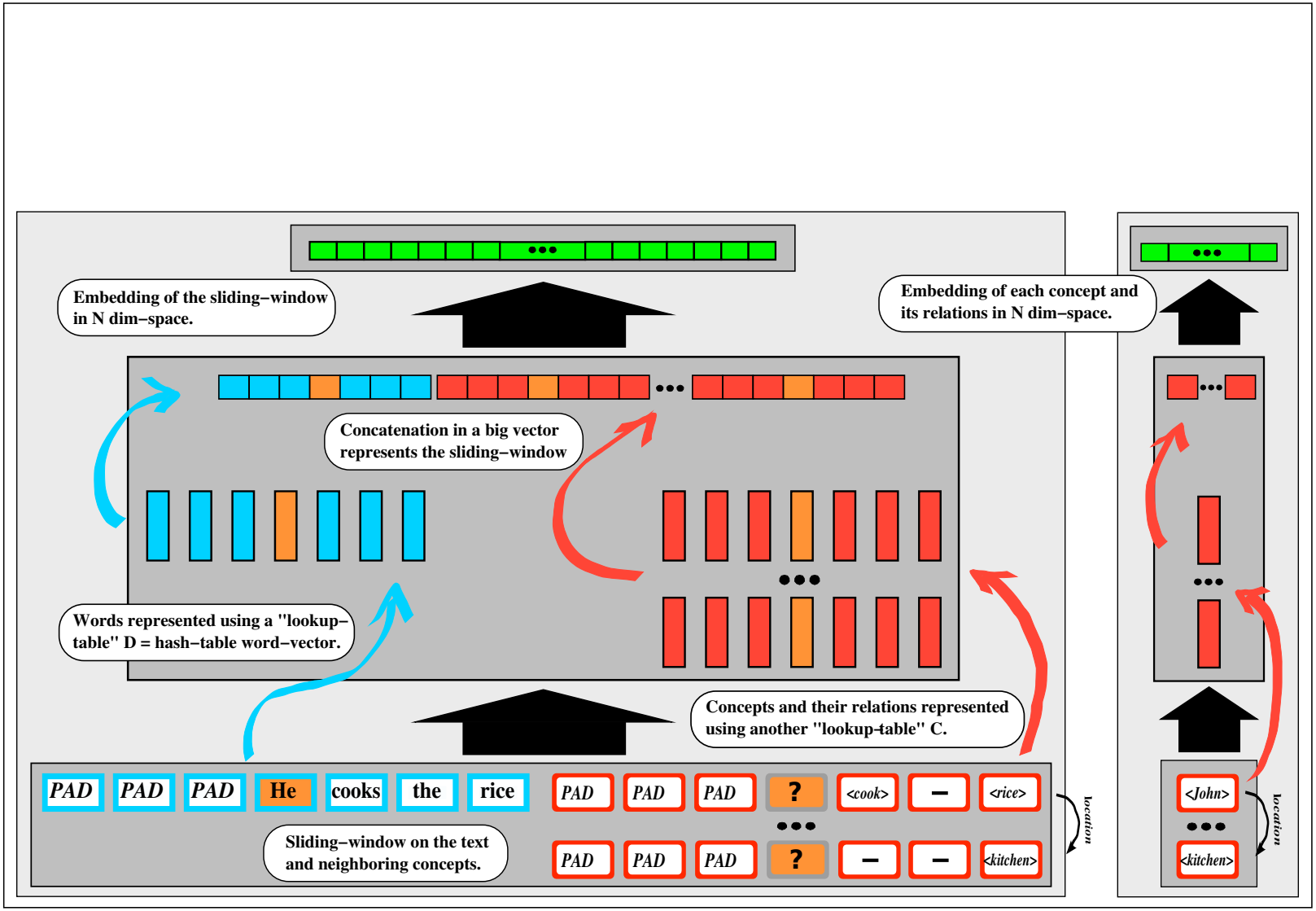
Scoring Illustration

Step 5: Get the concept <*John*> and its relations.



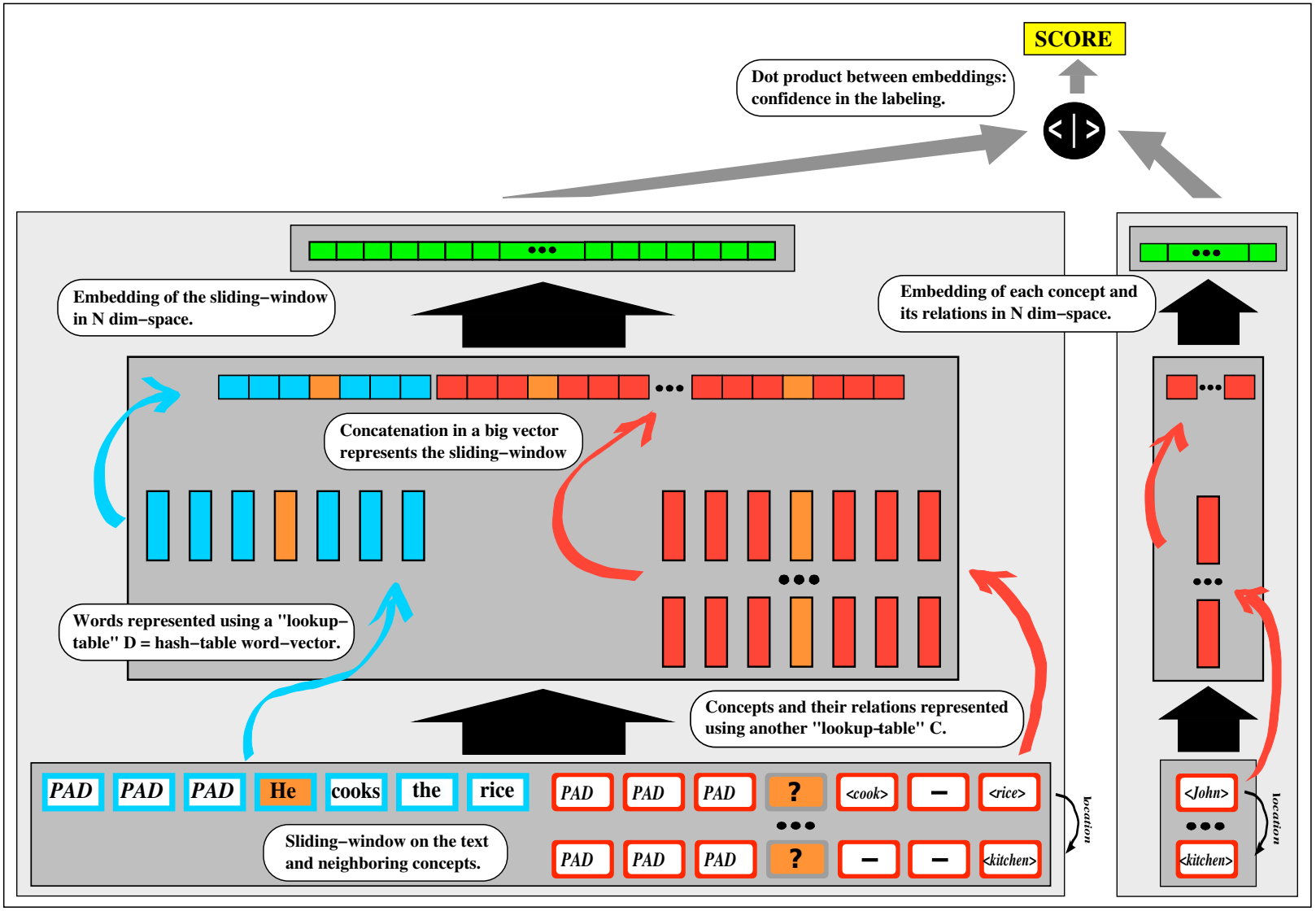
Scoring Illustration

Step 6: Compute $h(<John>, u)$.



Scoring Illustration

Step 7: Finally compute the score: $g_1(x, y_{-1}, u)^\top h(<John>, u)$.



Part III

Experiments

Generate Data by Simulation

1. Create a universe miming a house with 82 concepts: 15 verbs, 10 actors, 15 small objects, 6 rooms. . .
2. Run a simulation algorithm that generates training triples.

. . .
x: the father gets some yoghurt from the sideboard y: {<*John*>, <*get*>, <*yoghurt*>, <*sideboard*>}
x: he sits on the chair y: {<*Mark*>, <*sit*>, <*chair*>}
x: she goes from the bedroom to the kitchen y: {<*Gina*>, <*move*>, <*bedroom*>, <*kitchen*>}
x: the brother gives her the toy y: {<*Mark*>, <*give*>, <*toy*>, <*sister*>}
. . .

→ Generation a dataset of 50,000 training triples and 20,000 testing triples (≈55% ambiguous), without any human annotation.

Experimental Results

- Different tagging strategies.
- Different supervision levels: strong or weak.
- Different amounts of *universe* knowledge: no knowledge, knowledge about *containedby*, *location*, or both.

Method	Supervision	Features	Train Err	Test Err
SVM$_{struct}$	strong	$x + u$ (*loc*, *contain*)	18.68%	23.57%
NN$_{LR}$	strong	$x + u$ (*loc*, *contain*)	5.42%	5.75%

Experimental Results

- Different tagging strategies.
- Different supervision levels: strong or weak.
- Different amounts of *universe* knowledge: no knowledge, knowledge about *containedby*, *location*, or both.

Method	Supervision	Features	Train Err	Test Err
SVM_{struct}	strong	$x + u$ (*loc*, *contain*)	18.68%	23.57%
NN_{LR}	strong	$x + u$ (*loc*, *contain*)	5.42%	5.75%
NN_{OF}	strong	x	32.50%	35.87%
NN_{OF}	strong	$x + u$ (*contain*)	15.15%	17.04%
NN_{OF}	strong	$x + u$ (*loc*)	5.07%	5.22%

Experimental Results

- Different tagging strategies.
- Different supervision levels: strong or weak.
- Different amounts of *universe* knowledge: no knowledge, knowledge about *containedby*, *location*, or both.

Method	Supervision	Features	Train Err	Test Err
SVM_{struct}	strong	$x + u$ (*loc*, *contain*)	18.68%	23.57%
NN_{LR}	strong	$x + u$ (*loc*, *contain*)	5.42%	5.75%
NN_{OF}	strong	x	32.50%	35.87%
NN_{OF}	strong	$x + u$ (*contain*)	15.15%	17.04%
NN_{OF}	strong	$x + u$ (*loc*)	5.07%	5.22%
NN_{OF}	strong	$x + u$ (*loc*, *contain*)	**0.0%**	**0.11%**

→ More world knowledge & OF leads to better generalization.

Experimental Results

- Different tagging strategies.
- Different supervision levels: strong or weak.
- Different amounts of *universe* knowledge: no knowledge, knowledge about *containedby*, *location*, or both.

Method	Supervision	Features	Train Err	Test Err
SVM_{struct}	strong	$x + u$ (*loc*, *contain*)	18.68%	23.57%
NN_{LR}	strong	$x + u$ (*loc*, *contain*)	5.42%	5.75%
NN_{OF}	strong	x	32.50%	35.87%
NN_{OF}	strong	$x + u$ (*contain*)	15.15%	17.04%
NN_{OF}	strong	$x + u$ (*loc*)	5.07%	5.22%
NN_{OF}	strong	$x + u$ (*loc*, *contain*)	**0.0%**	**0.11%**
NN_{OF}	weak	$x + u$ (*loc*, *contain*)	**0.64%**	**0.72%**

→ Learning with weak supervision is *almost* as efficient.

Features Learnt by the System

- Our model learns representations of concepts.
- Nearest neighbors in this vector space:

Query Concept	Closest Concepts
<*Gina*>	<*Francoise*>, <*Maggie*>
<*Mark*>	<*Brian*>, <*John*>
<*football*>	<*toy_car*>, <*videogame*>
<*chocolate*>	<*salad*>, <*milk*>
<*desk*>	<*bed*>, <*table*>
<*livingroom*>	<*kitchen*>, <*garden*>
<*get*>	<*sit*>, <*give*>

- Similar concepts are close to each other.
- Such information is never given explicitly.

Summary

- *Simple*, but *general framework* for language grounding based on the task of *concept labeling*.
- *Scalable*, *flexible learning algorithm* that can learn without hand-crafted rules or features and under weak supervision.
- Simulation validates our approach and shows that learning to disambiguate with world knowledge is possible.

Next step: train a character "living" in a "computer game world" to learn language from scratch i.e. *from interactions alone*.

Thank You

Thank you for your attention.

Talking to Computers

"Computers are being used today to take over many of our jobs. They can perform millions of calculations in a second, handle mountains of data, and perform routine office work much more efficiently and accurately than humans.

But when it comes to telling them what to do, they are tyrants. They insist on being spoken to in a special computer language, and act as though they can't even understand a simple English sentence."

Terry Winograd – 1971

in *Procedures as a Representation for Data in a Computer program for Understand Natural Language*

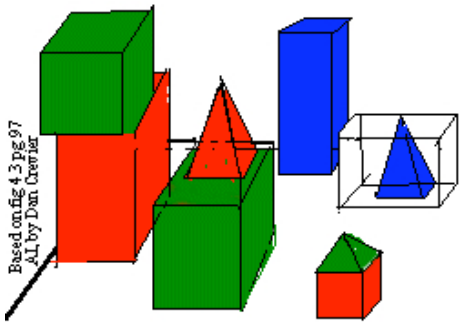
(Some of the) Previous Work

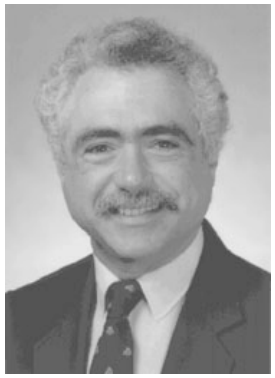
No use of world knowledge as input (only natural language):

- Mapping language with visual reference: [Winston '76], [Thibadeau '86], [Siskind '96], [Yu & Ballard '04], [Barnard & Johnson '05], [Fleischman & Roy '07].
- Mapping from sentences to meaning in formal language: [Zettlemoyer & Collins, '05], [Wong & Mooney, '07], [Chen & Mooney '08]
- Example applications:
  (i) word-sense disambiguation (from images),
  (ii) generate Robocup commentaries from actions,
  (iii) convert questions to database queries.

SHRDLU & Block Worlds

- SHRDLU: early natural language understanding computer program. [Winograd, '72],[Bobrow & Winograd, '76]
- Use both language and world knowledge as input.





- Great success of AI $\rightarrow$ **great hopes**.
- **No later success on more realistic situations**.
- **Problem:** SHRDLU involves hand-coding in 2 ways,
  (1) World model (block world).
  (2) **Mapping natural language to world**.

From Concept Labeling to Semantics

- Concept labeling is not sufficient for semantic interpretation.
- Just add **Semantic Role Labeling**:

He	cooks	the	rice		
<*John*>	<*cook*>	-	<*rice*>		
ARG1	REL	-	ARG2	$\longrightarrow$	<*cook*>(<*John*>, <*rice*>)

- The system can **update its own world representation** and carry on **story understanding**.
- For example:
  "John went to the kitchen and Mark stayed in the living room."
  "He cooked the rice and served dinner."

Benchmarking

Task: for a NL sentence, choose the corresponding action among several alternatives: weak supervision and noisy NL.

Dataset: Robocup [Chen & Mooney '08].

Method	Matching F1-score
Random	0.465
Wasper	0.530
Krisper	0.645
Wasper-gen	0.650
NN_{WEAK}	**0.669**

$\rightarrow$ NN_{WEAK} trains well on NL under weak supervision.

Simulation Algorithm

A. An universe is initialized i.e. concepts and relations are created.

B. The simulation algorithm is run with:

1. **Generate a new event**, $(v, a) = \mathit{event}(u)$.
   Generates verb + set of args $\rightarrow$ a *coherent* action given the universe. *E.g. actors change location and pick up, exchange objects. . .*

2. **Generate a training triple**, i.e. (x,y)=$\mathit{generate}(v, a)$.
   Returns a sentence and concept labeling pair given a verb + args. *This sentence should describe the event.*

3. **Update the universe**, i.e. $u = \mathit{exec}(v)(a, u)$.