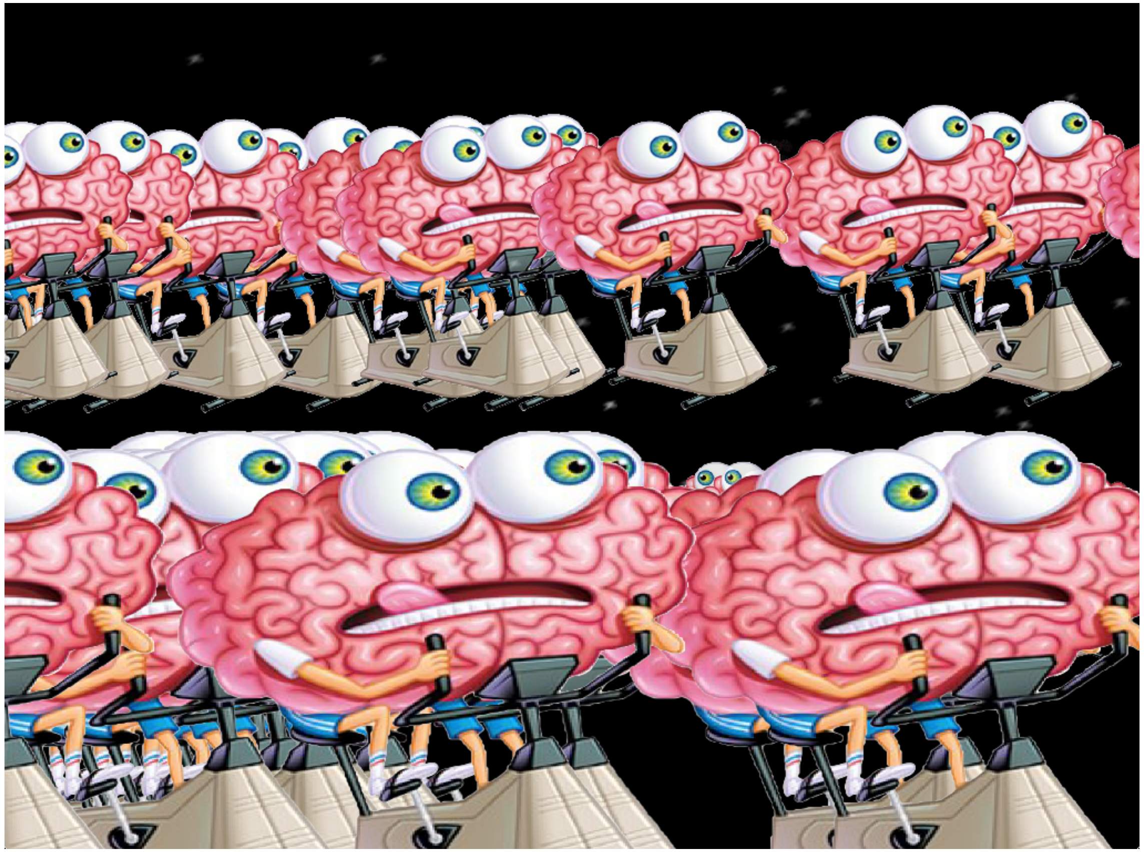
How Can We Learn Situated Language?$^{(*)}$



$(*)$ Answer not included.

Jason Weston (Google, NY)
Antoine Bordes & Nicolas Usunier (Université de Technologie de Compiègne)
Ronan Collobert (IDIAP, Switzerland).

The Dream

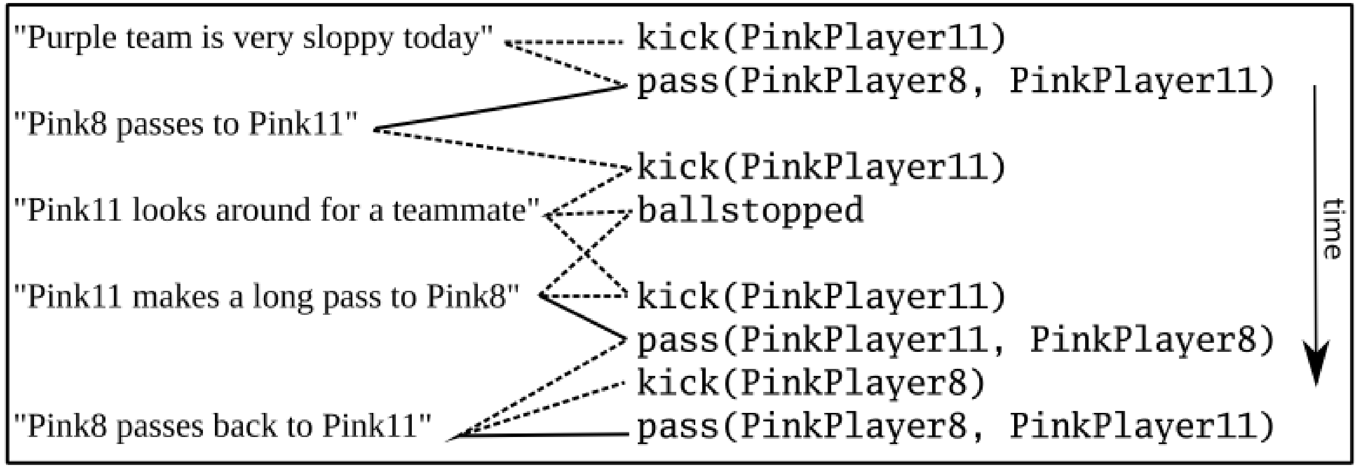
GOAL: A learner begins with little or no knowledge in an environment where language is used between actors to fulfill tasks.

Over time it *learns* what the language "means" in the sense that:

1. It can answer questions (e.g. location/state of object/person.)
2. Perform actions you ask it to.
3. Understand the goals of other actors (desired location/state)?

Training Signal

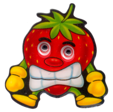
1. Can it learn through observation alone? i.e. just learning correspondence between modalities?



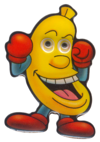
!!! WARNING: More common case not so correlated as above? !!!

2. Will it be much better / learn faster if it can ask questions? (how?)
3. Will it be much better if it can act in the environment? (how?)

Grounding NLP: Why?



Quite a lot of NLP work solves labeling tasks e.g. POS, chunking, parsing, SRL, MT, . . . ignoring world knowledge via grounding.



We understand language because it has a deep connection to the world it is used in/for →

“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 close to the river.

Here, our learner needs a world model (memory) that is dynamic.

The Environment

Humans use (designed) language as a tool to communicate about our physical reality (or metaphysical considerations of it).

 Planet Earth = tricky:

vision, speech, motor control + *language understanding.*

 Multi-user game (e.g. on the internet) = easier.

Simplest version = text adventure game. Good test-bed for ML?

Would be great if game players were interested in teaching our models (Tamagotchi ['96]).

The Learning Signal : text adventure game

Represent all atomic actions in the game as concepts
(`get`, `move`, `give`, `shoot`, `...`).

Represent all physical objects in the game as concepts
(`character1`, `key1`, `key2`, `...`).

(Can consider this signal as a pre-processed version of a visual signal.)
Assumes 'vision' is solved : non-noisy world knowledge

Open domain: Variable concepts + Variable vocab.
New concepts: new characters, compound definitions (e.g. "family").

Closed domain: Fixed concepts + Variable vocab.
E.g. `axe1` → "the big axe", "weapon", "it".

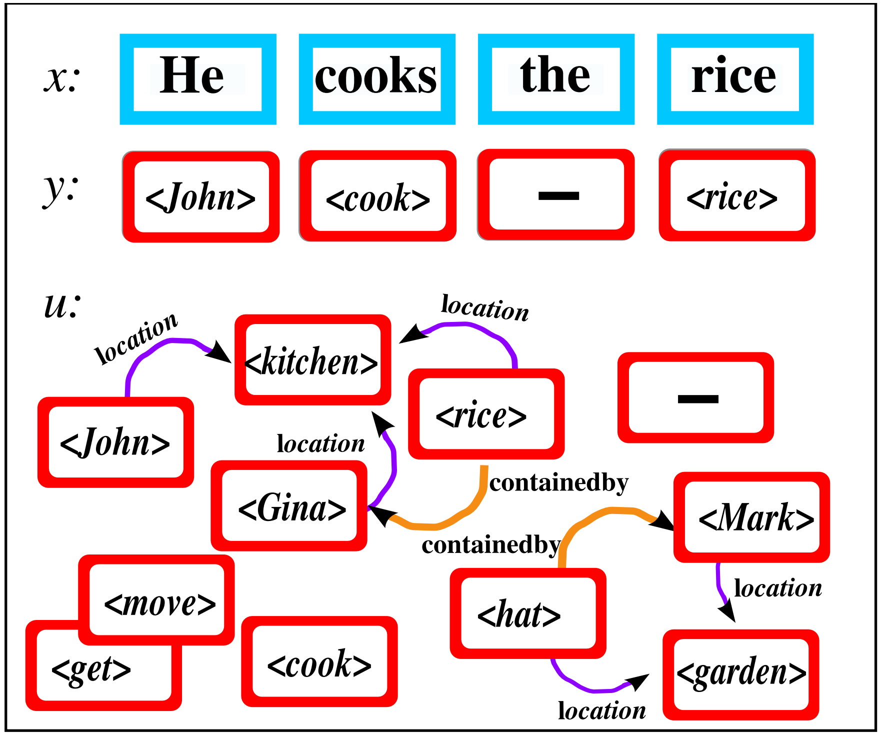
Closed domain is simpler, but maybe still challenging?

The Concept Labeling Task: (something concrete)

Definition: Map any natural language sentence $x \in \mathcal{X}$ to its labeling in terms of *concepts* $y \in \mathcal{Y}$, where y is a sequence of concepts.

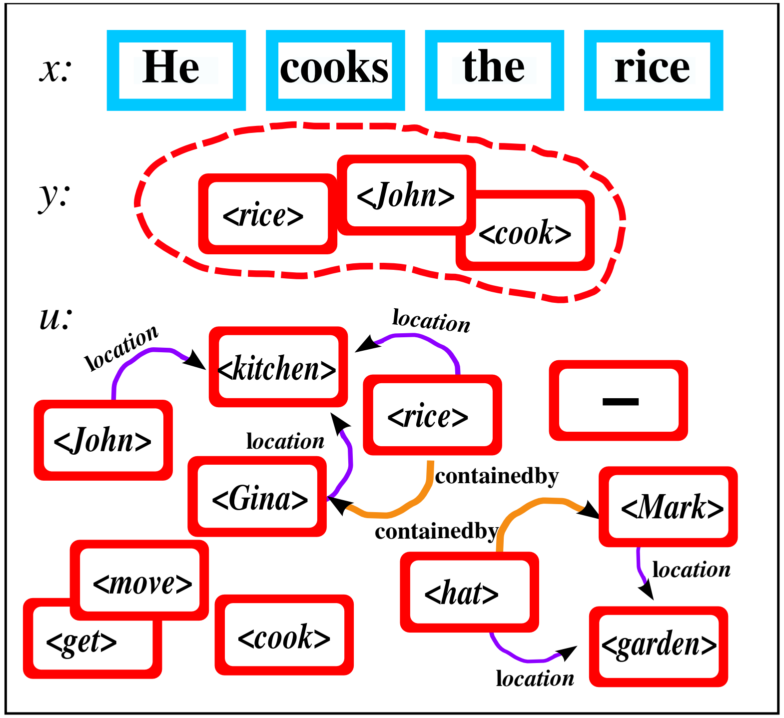
Training data triples $\{\mathbf{x}_i, \mathbf{y}_i, \mathbf{u}_i\}_{i=1,\ldots,m}$ where $\mathbf{u}_i$ is the current state the world ("universe").

Universe $=$ set of concepts and their relations to each other.



Example of Weak Concept Labeling

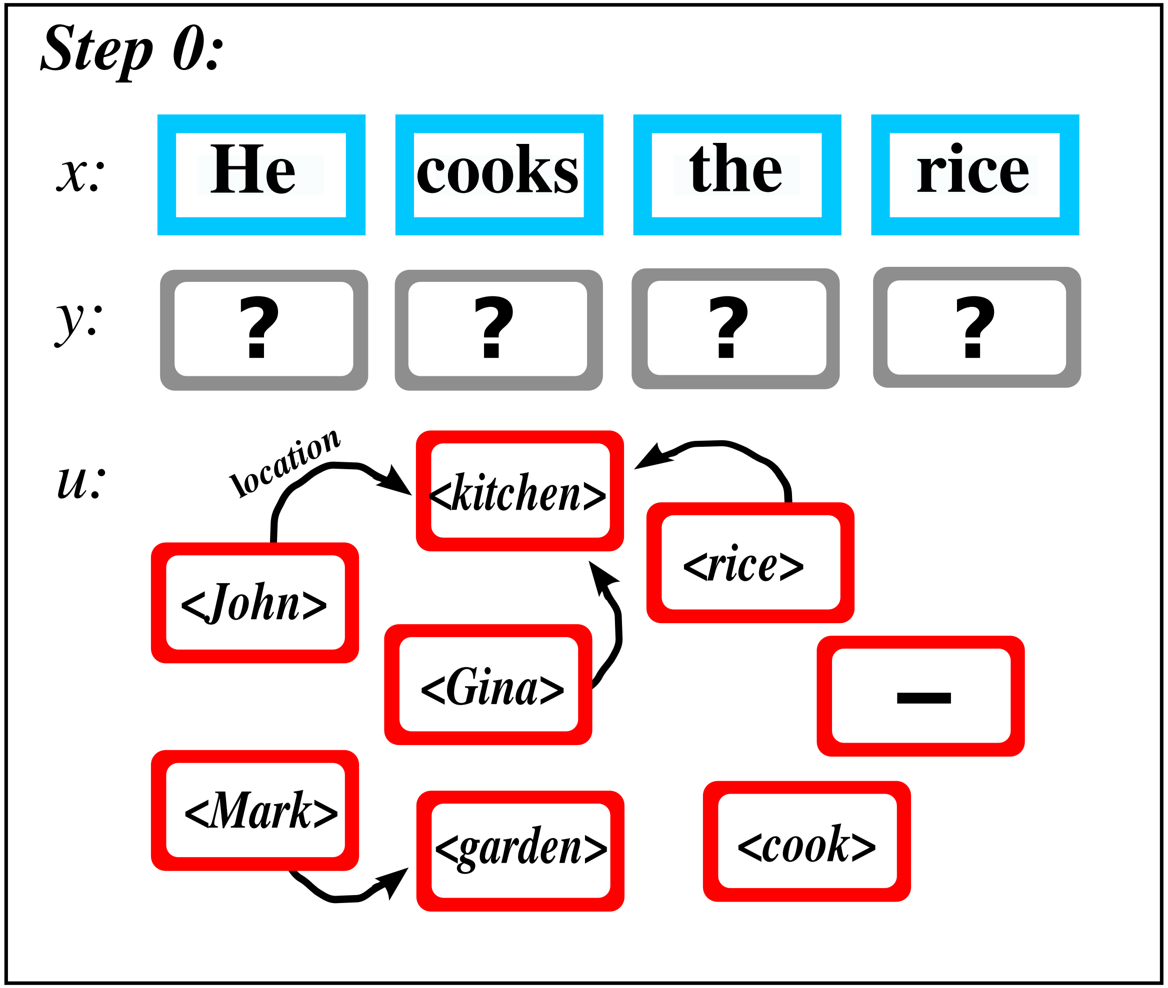
Same example, but "alignment" is not given in training signal.



This is slightly(!) more realistic: a child sees actions and hears sentences must learn correlation.

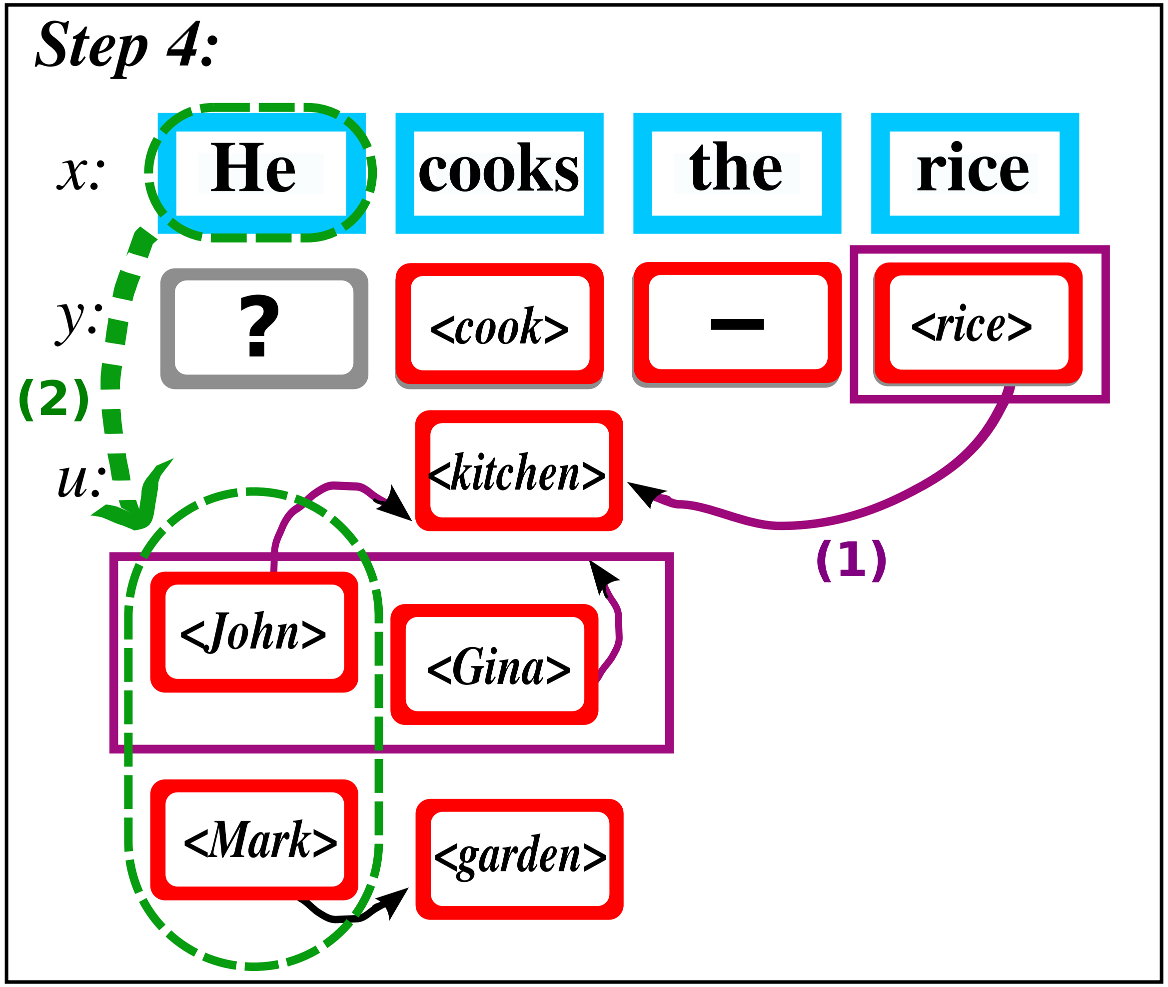
Even harder: the bag of possible concepts is larger, e.g. the set of events that occurred nearby, recently. Uncorrelated sentences, too.

Disambiguation Example



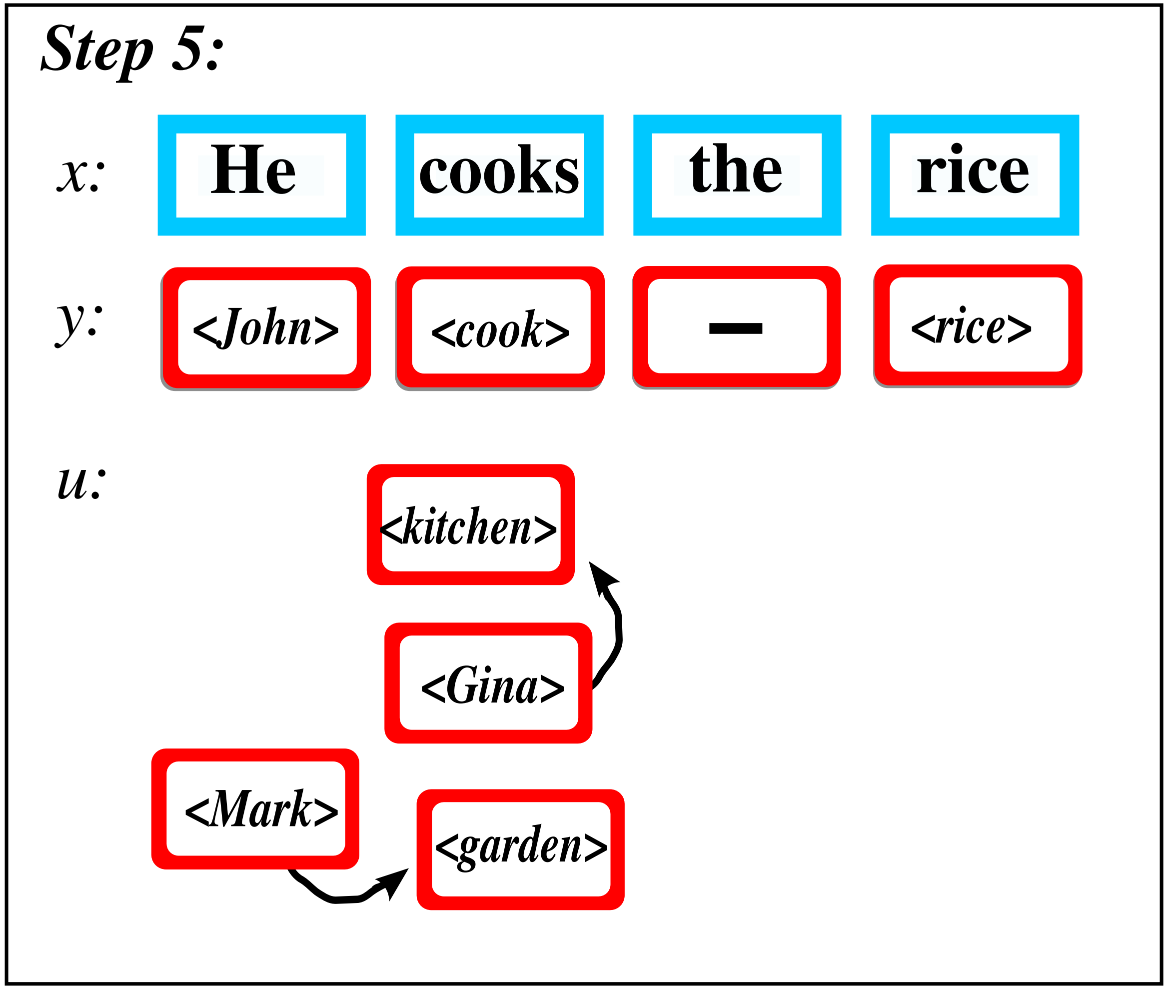
Start with the least ambiguous words..

Disambiguation Example



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

Disambiguation Example



John is the only male in the kitchen!

(Some of the) Previous Work

- Blocks world, KRL [Winograd, '72],[Bobrow & Winograd, '76]
- Ground language with visual reference, e.g. in blocks world [Winston '76],[Feldman et al. '96] or more recent works [Fleischman & Roy '07],[Barnard & Johnson '05],[Yu & Ballard '04],[Siskind'00].
- Map from sentence to meaning in formal language [Zettlemoyer & Collins, '05], [Wong & Mooney, '07], [Chen & Mooney '08]
- Other more recent works!!
- All the stuff mentioned over the last three days!!

Note: datasets like Robocup data or "Ambig-childworld" (Kate & Mooney, '07) don't use world knowledge ($\mathbf{u}$).

Using World knowledge can resolve ambiguities

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. . .)

Concept labeling: more general than word-sense disambiguation, co-reference resolution, or named-entity recognition . . .

Concept Labeling: resolving ambiguities

The main difficulty of concept labeling → ambiguous words

- Mislabeling destroys any subsequent semantic interpretation.
- Ambiguities we consider:
  - Location-based: can be solved using *locations* of the concepts: (contained-by or located relations)

    "father picked *it* up" or "*he* got the coat in the hall"

    "the *milk* in the closet' or "the *one* in the closet"
  - Category-based: can be solved using semantic categorization:

    "*He* cooks the rice in the kitchen';

    "John drinks the *orange*" and "John ate the *orange*".

Labeled Data generated by the Simulation

Simple "adventure game" simulation: a house with 58 concepts: 15 verbs, 10 actors, 27 objects, 6 rooms, generate training data with:

1. Generate event: coherent action (verb+args) given universe.
2. Generate example: (sentence, concept label, universe) triple.
3. Update the universe.

...

x:	the father gets some yoghurt from the sideboard
y:	- *<father>* *<get>* - *<yoghurt>* - - *<sideboard>*
x:	he sits on the chair
y:	*<brother>* *<sit>* - - *<chair>*
x:	she goes from the bedroom to the kitchen
y:	*<mother>* *<move>* - - *<bedroom>* - - *<kitchen>*
x:	the brother gives the toy to her
y:	- *<brother>* *<give>* - *<toy>* - *<sister>*

...

“Shared Representation” Model

To score the concepts we could use:

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

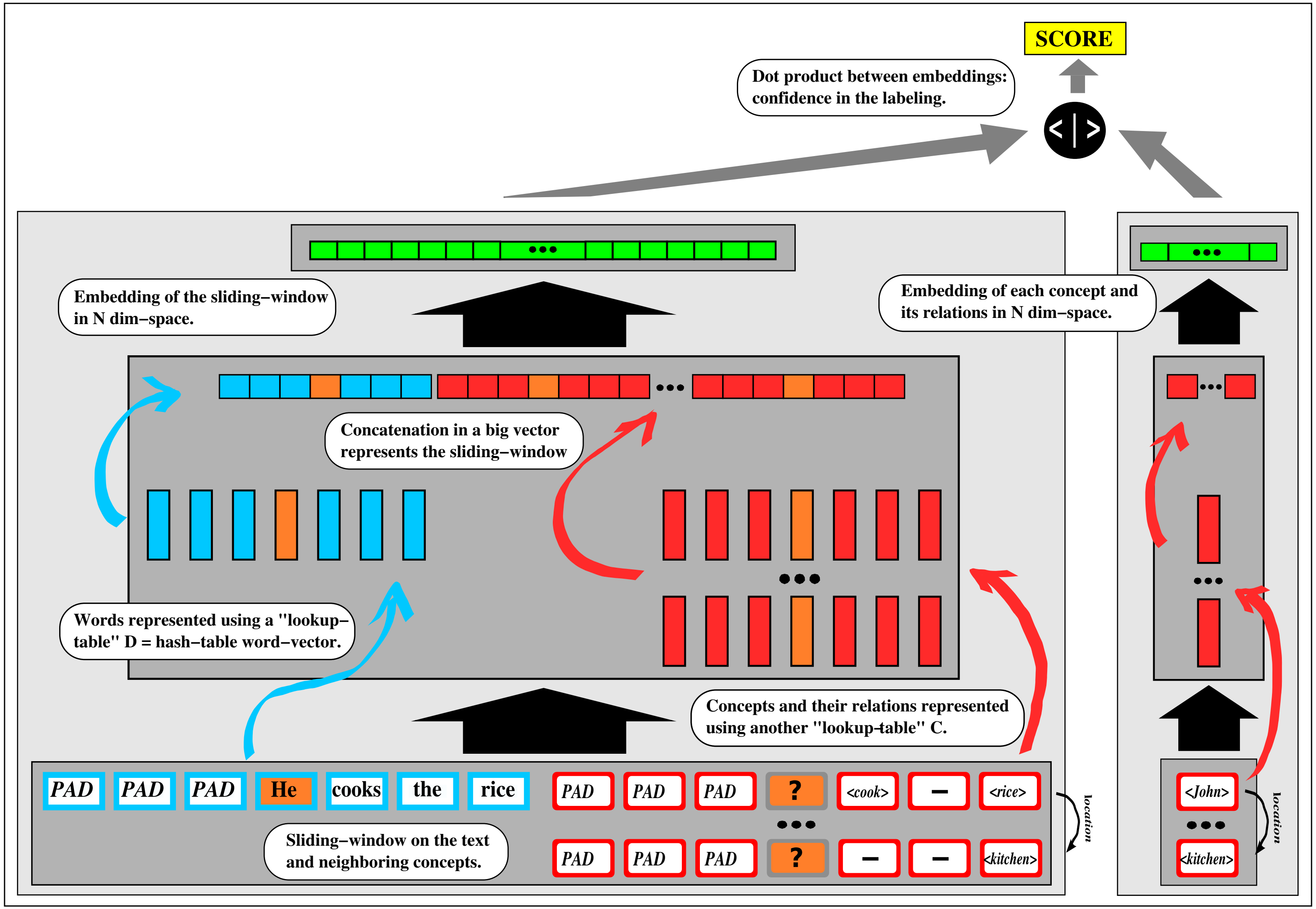
Our model combines two functions which are neural networks:

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

- $g_i(x, y_{-i}, u)$ 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 to N dim-space.
- Dot-product: confidence that i^{th} word labeled with concept y_i.

Scoring Illustration

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



Greedy "Order-free" Inference

(approximate the argmax)

Adapted from LaSO (Learning As Search Optimization) [Daumé & al.,'05].

Inference algorithm:

1. For all the positions not yet labeled, predict the most likely concept.
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. Loop.

Train the System

- Online training i.e. prediction and update for each example.
- At each greedy step, if a prediction $\hat{y}^t$ is incorrect, several updates are made to the model to satisfy:
  - Strong supervision: we want any incorrect partial prediction to be ranked below all correct partial labeling.
    $\rightarrow$ Note: "Order-free" is not directly supervised.
  - Weak supervision: rank anything (unused) in the "bag" higher than something not in the bag.
- All updates performed with SGD + Backpropagation.

Experimental Results

Method	Features	Train Err	Test Err
SVM$_{struct}$	x	42.26%	42.61%
SVM$_{struct}$	$x + u$ (loc, contain)	18.68%	23.57%
NN$_{multi}$	x	35.80%	36.97%
NN$_{LR}$	x	32.80%	35.80%
NN$_{LR}$	$x + u$ (loc, contain)	5.42%	5.75%
NN$_{OF}$	x	32.50%	35.87%
NN$_{OF}$	$x + u$ (contain)	15.15%	17.04%
NN$_{OF}$	$x + u$ (loc)	5.07%	5.22%
NN$_{OF}$	$x + u$ (loc, contain)	**0.0%**	**0.11%**
NN$_{WEAK}$	$x + u$ (loc, contain)	**0.64%**	**0.72%**

- Different tag strategies: learning "least ambiguous first" (OF) best.
- Different amounts of *universe* knowledge: no knowledge, knowledge about *containedby*, *location*, or both. More = better.

Same algorithm on Robocup performs ok (0.67 F1) vs. e.g. Krisper (0.645 F1) and Wasper-Gen (0.65 F1) . . . but u not used.

What's Missing (Answer: LOTS!)

1. Answering questions. Simple version: learn map from text to binary labeling of all concepts and relations. Then, can answer things like: "where is my hat?", "who cooked the rice?".
2. Executing instructions: learn map from text to action. Pretty much have this with SRL.
3. Learning what an action does: learn map from action to change in $\mathbf{u}$ (in our framework, doesn't seem that hard.)
4. Semi-supervised: multi-task with unlabeled task, also doesn't seem that hard. This will help train the word vectors.
5. Open domain.
6. Partially observed universe.
7. Subsymbolic mapping + representation ?
8. More challenging data: more relations, noisy, human-labeled . . .

However, will this be a completely unified system, or just a collection of parts..? :(