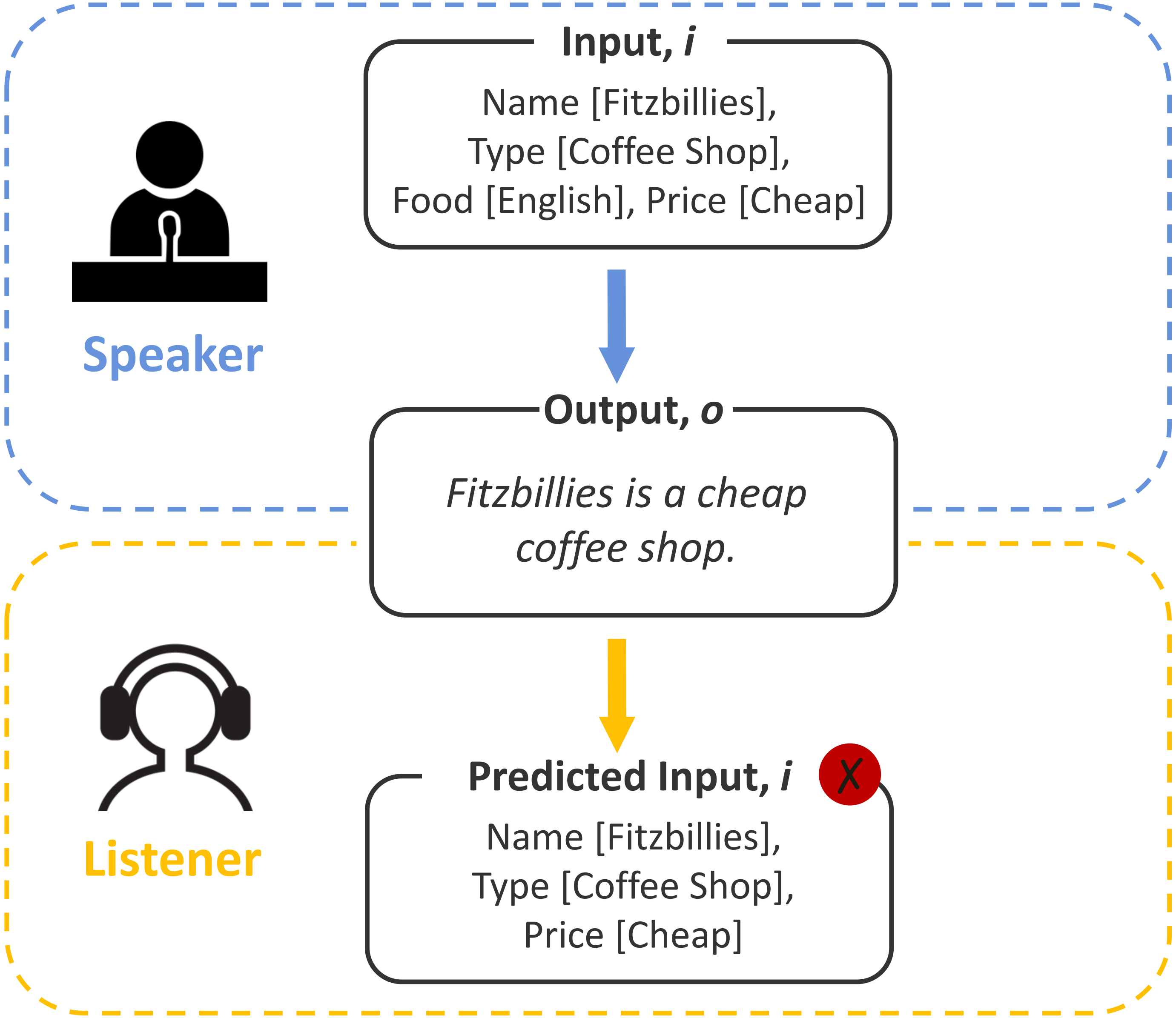
Pragmatically Informative Text Generation

Berkeley
N L P

Sheng Shen, Daniel Fried, Jacob Andreas, and Dan Klein

Why Might Generation Need Pragmatics?



Input, *i*

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

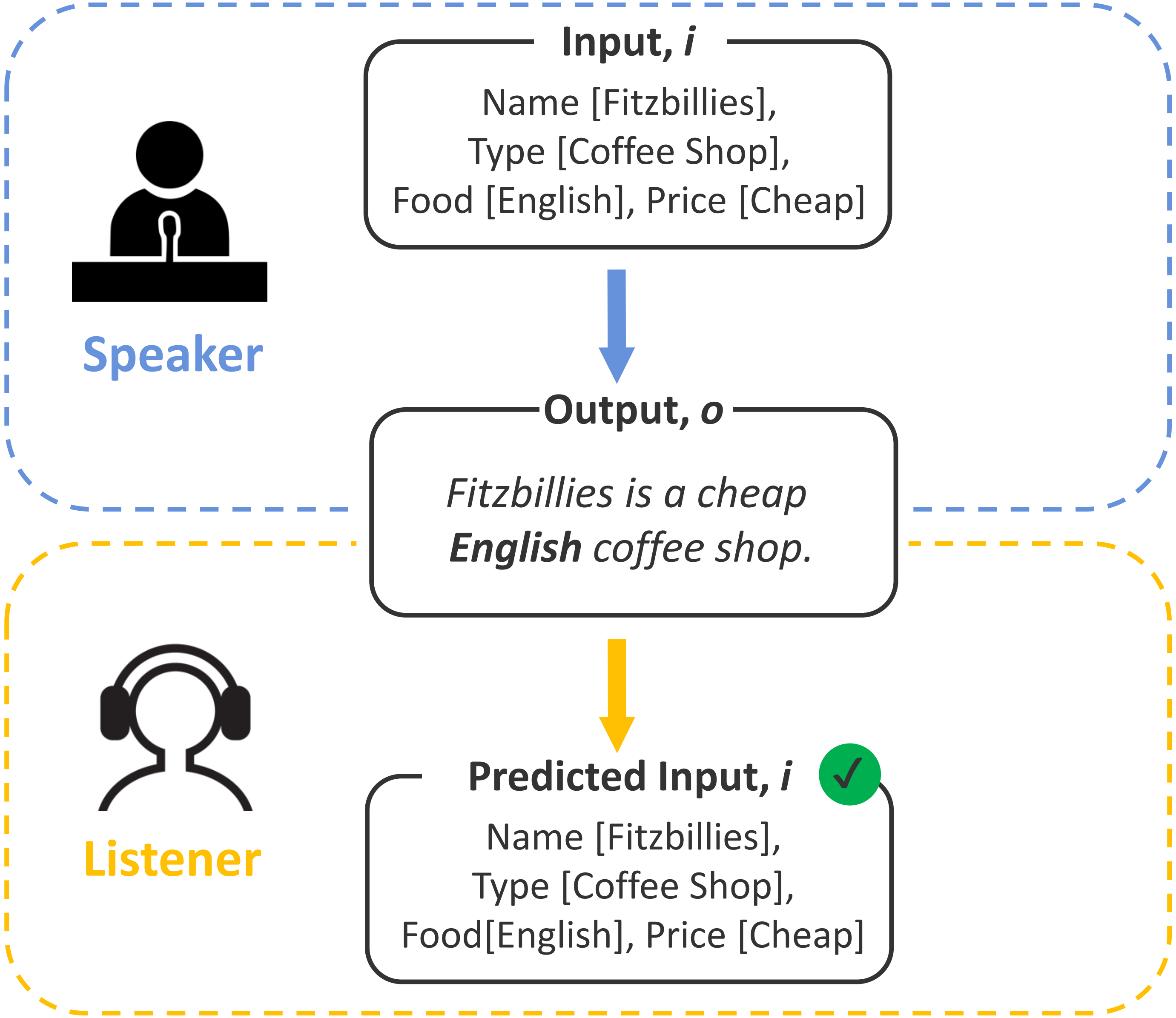
Output, *o*

Fitzbillies is a cheap coffee shop.

Predicted Input, *i* X

Name [Fitzbillies],
Type [Coffee Shop],
Price [Cheap]

Why Might Generation Need Pragmatics?



Input, *i*

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Speaker

Output, *o*

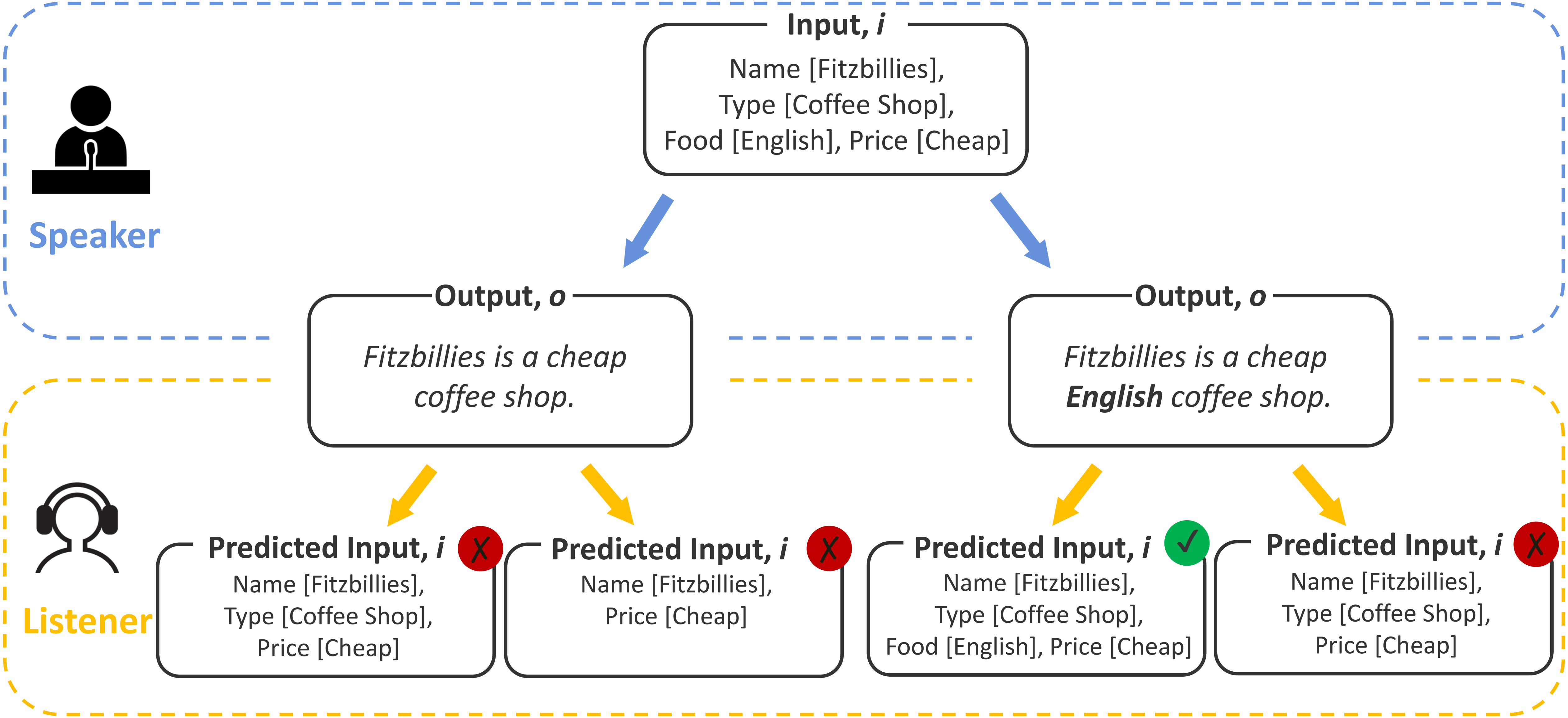
Fitzbillies is a cheap
***English** coffee shop.*

Listener

Predicted Input, *i* ✓

Name [Fitzbillies],
Type [Coffee Shop],
Food[English], Price [Cheap]

Generation as a Pragmatic Game



Speaker

Input, *i*

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Output, *o*

Fitzbillies is a cheap coffee shop.

Output, *o*

*Fitzbillies is a cheap **English** coffee shop.*

Listener

Predicted Input, *i* ✗

Name [Fitzbillies],
Type [Coffee Shop],
Price [Cheap]

Predicted Input, *i* ✗

Name [Fitzbillies],
Price [Cheap]

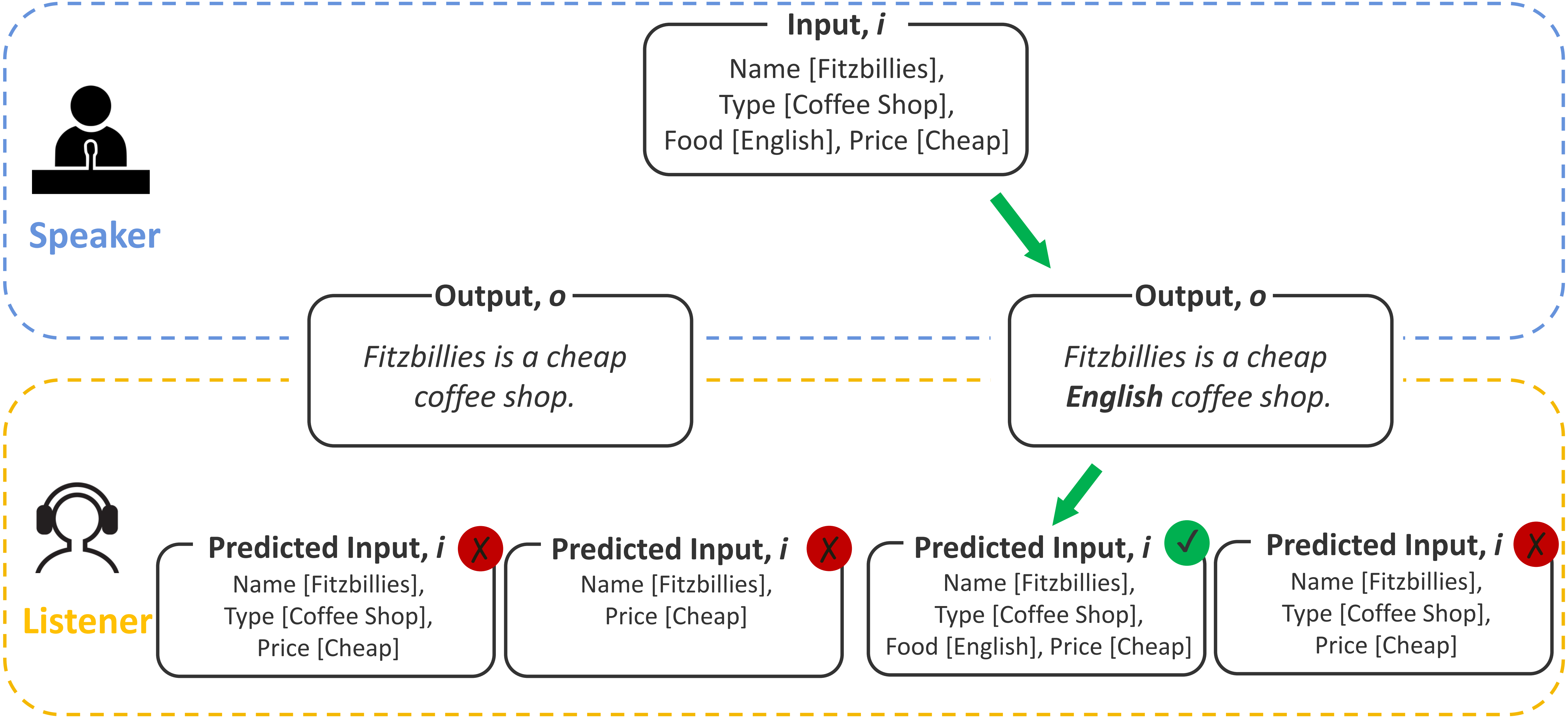
Predicted Input, *i* ✓

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Predicted Input, *i* ✗

Name [Fitzbillies],
Type [Coffee Shop],
Price [Cheap]

Generation as a Pragmatic Game



Speaker

Input, *i*

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Output, *o*

Fitzbillies is a cheap coffee shop.

Output, *o*

*Fitzbillies is a cheap **English** coffee shop.*

Listener

Predicted Input, *i* ✗

Name [Fitzbillies],
Type [Coffee Shop],
Price [Cheap]

Predicted Input, *i* ✗

Name [Fitzbillies],
Price [Cheap]

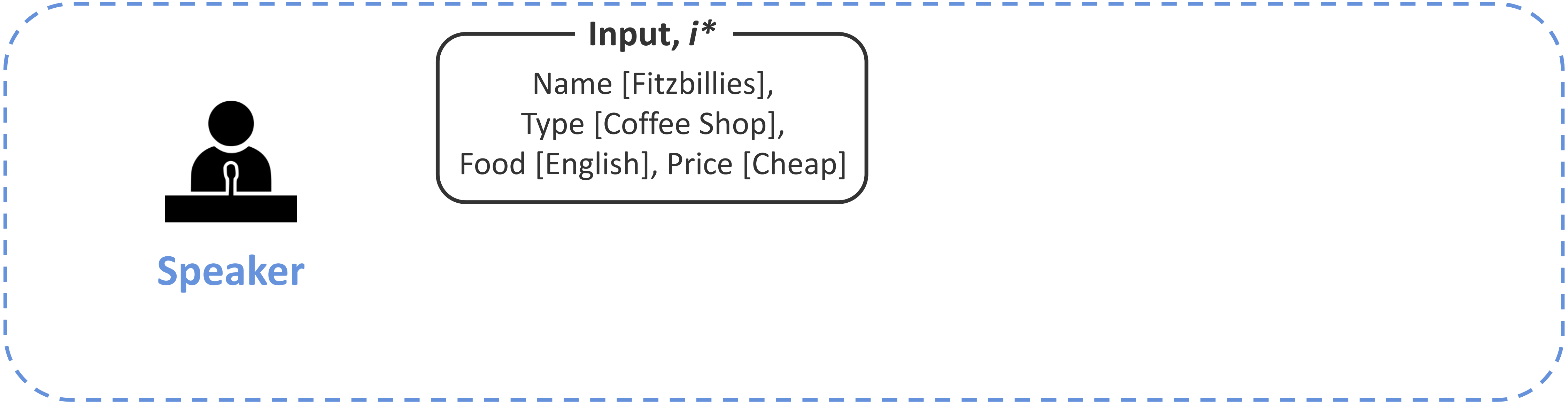
Predicted Input, *i* ✓

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Predicted Input, *i* ✗

Name [Fitzbillies],
Type [Coffee Shop],
Price [Cheap]

Generating Pragmatic Output Text

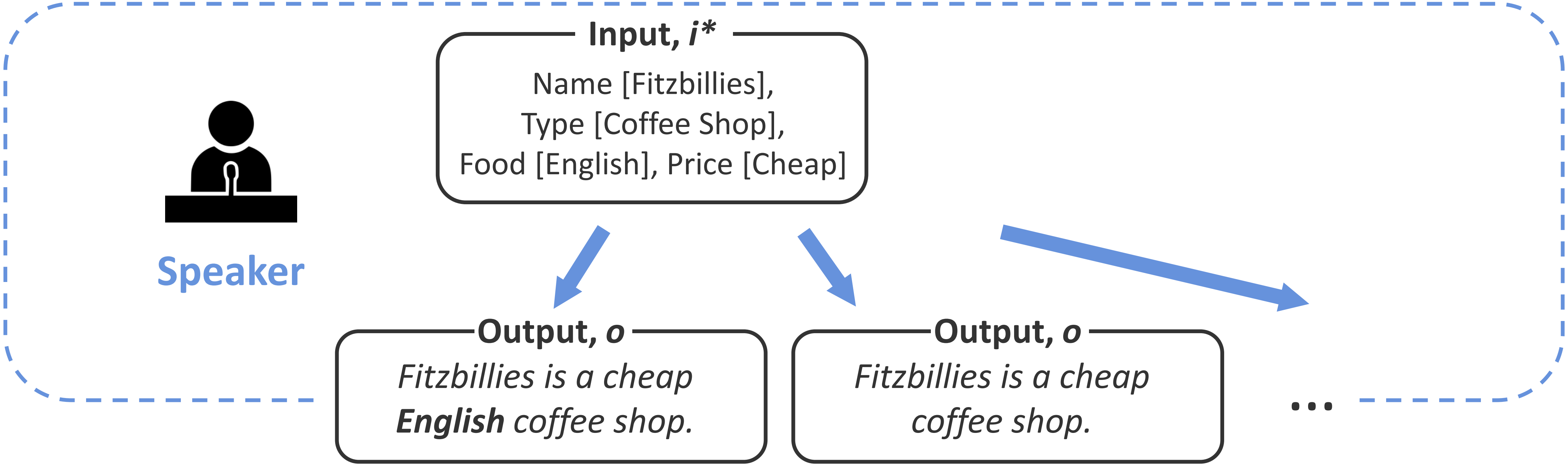


Input, *i**

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Speaker

Generating Pragmatic Output Text



Input, *i**

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Speaker

Output, *o*

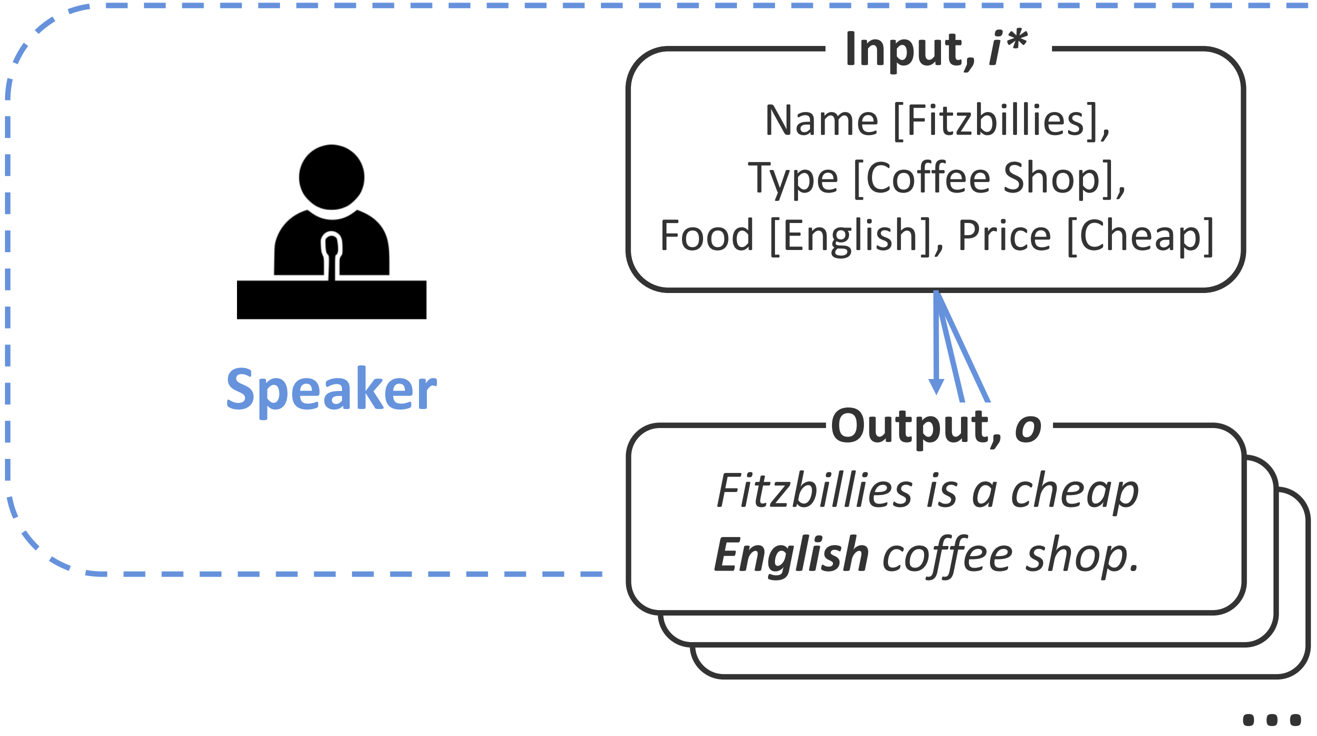
*Fitzbillies is a cheap **English** coffee shop.*

Output, *o*

Fitzbillies is a cheap coffee shop.

...

Generating Pragmatic Output Text

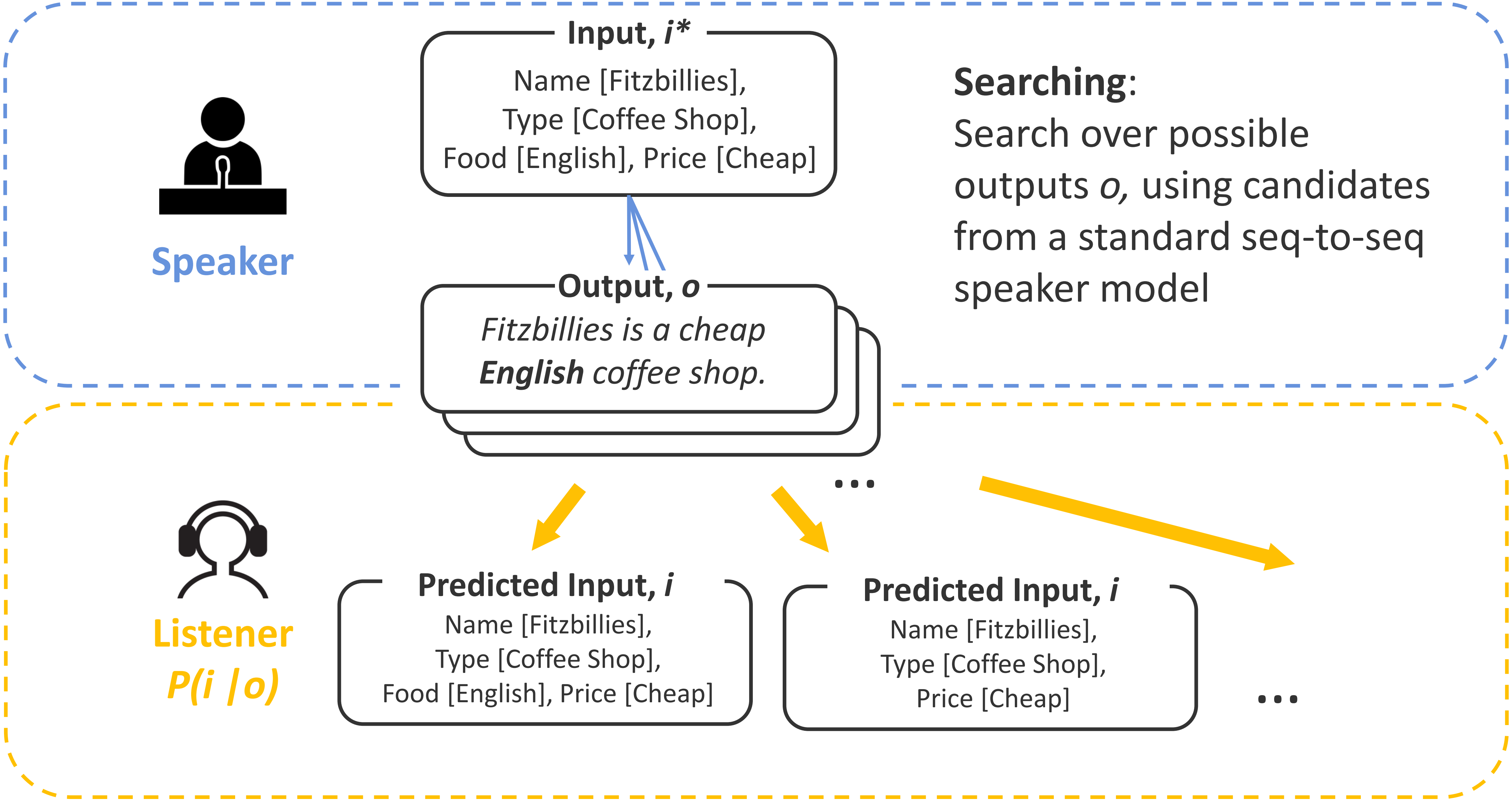


Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Fitzbillies is a cheap ***English*** *coffee shop.*

Searching:
Search over possible outputs *o*, using candidates from a standard seq-to-seq speaker model

Generating Pragmatic Output Text



Speaker

Input, *i**

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Output, *o*

Fitzbillies is a cheap
***English** coffee shop.*

...

Searching:
Search over possible outputs *o*, using candidates from a standard seq-to-seq speaker model

Listener
P(i |o)

Predicted Input, *i*

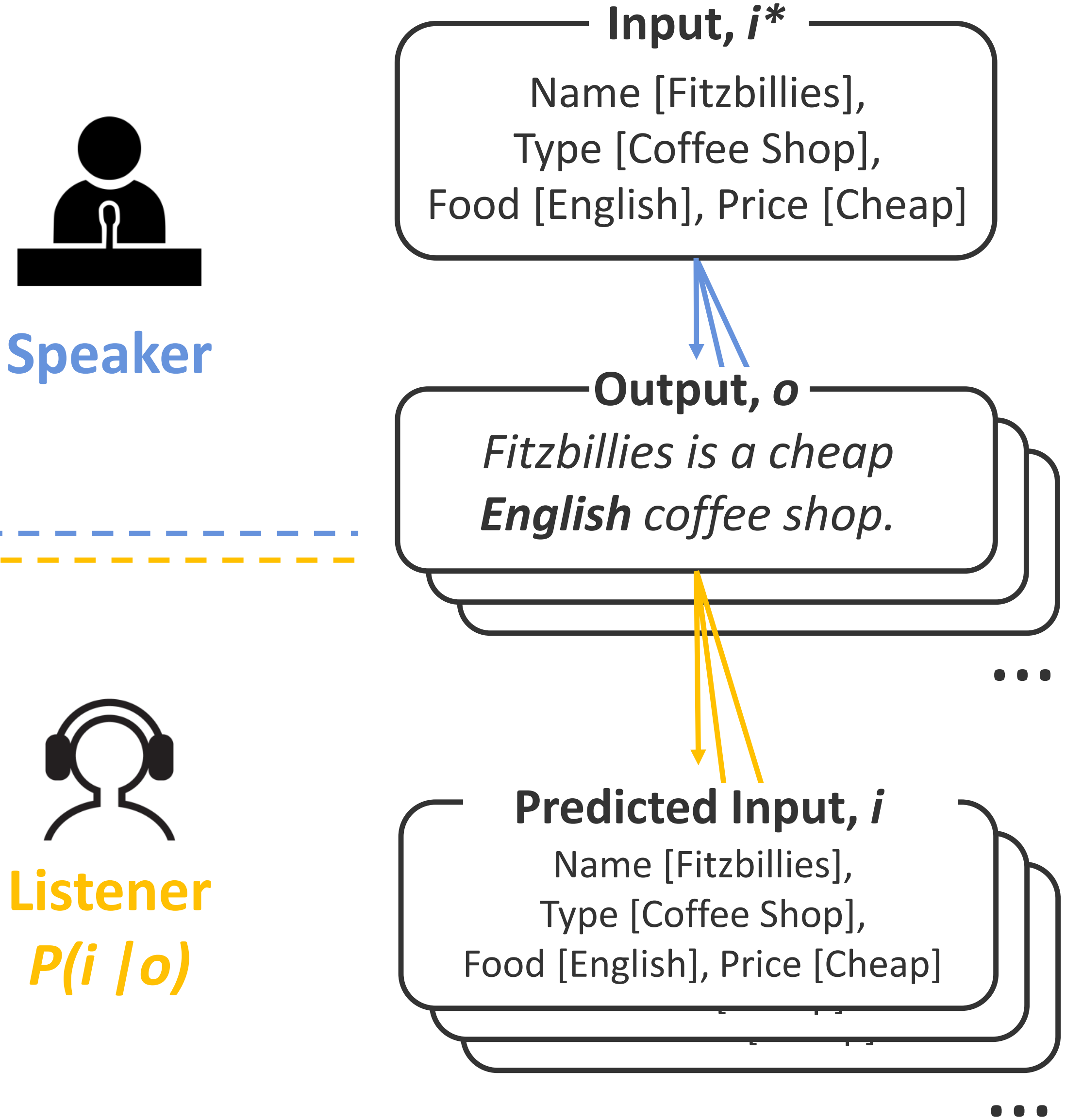
Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Predicted Input, *i*

Name [Fitzbillies],
Type [Coffee Shop],
Price [Cheap]

...

Generating Pragmatic Output Text



Speaker

Input, *i**

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

Output, *o*

*Fitzbillies is a cheap **English** coffee shop.*

...

Listener

P(i |o)

Predicted Input, *i*

Name [Fitzbillies],
Type [Coffee Shop],
Food [English], Price [Cheap]

...

Searching:
Search over possible outputs *o*, using candidates from a standard seq-to-seq speaker model

Scoring:
Choose an output with maximum listener probability, ***P(i* | o)***

How to Construct the Listener?

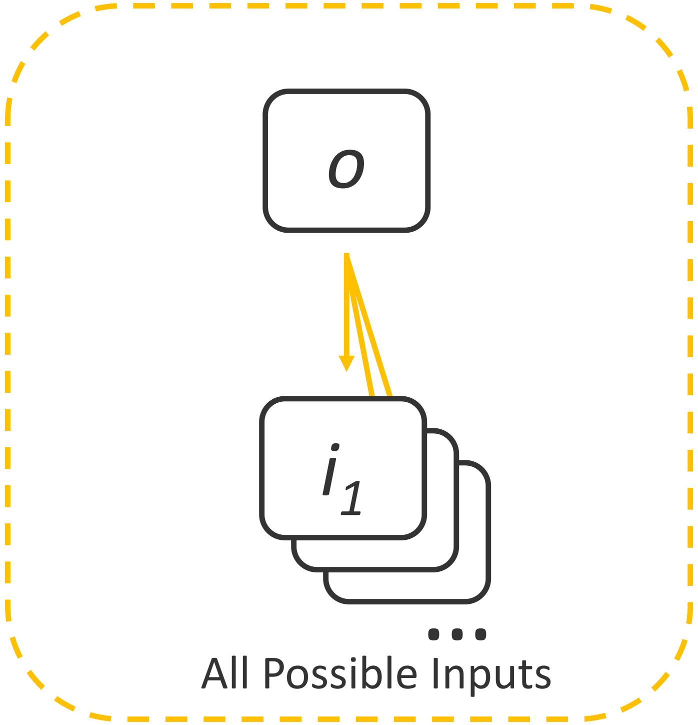
Reconstructor-Based

Train a separate **Listener** model to give a distribution over any possible inputs.

or

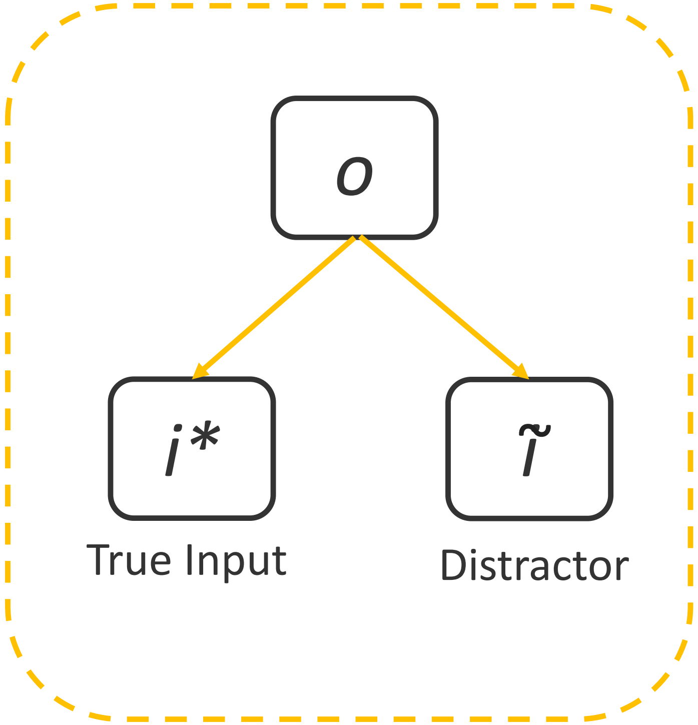
Distractor-Based

Construct a context-appropriate *distractor* input that **Listener** needs to distinguish the true input from.



Listener

P(i |o)



Past Work on Pragmatic Generation

Convey All Relevant Info

[Grice 1970, Horn 1984,
Dušek and Jurčíček 2016,
Li et al. 2016,
He et al. 2016,
Fried et al. 2018,
Cohn-Gordon et al. 2019, ...]

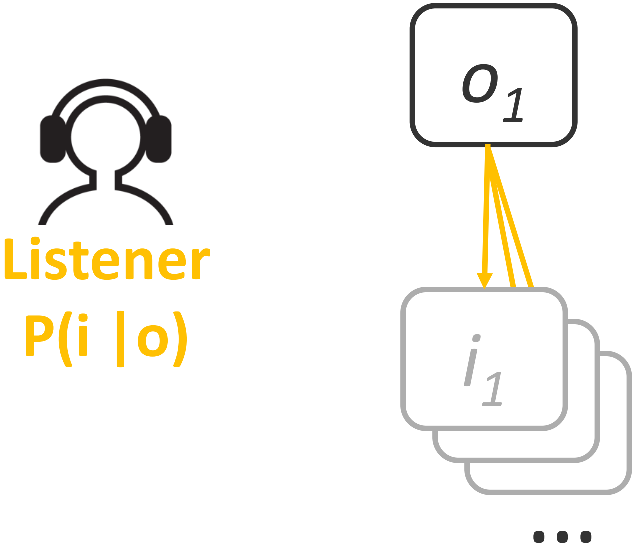
Motivates Reconstructor

Be Informative in Context

[Golland et al. 2010,
Frank and Goodman 2012,
Mao et al. 2015,
Andreas and Klein 2016,
Vedantam et al. 2018,
Cohn-Gordon et al. 2018, …]

Motivates Distractor

Reconstructor-Based Pragmatics



Listener

P(i |o)

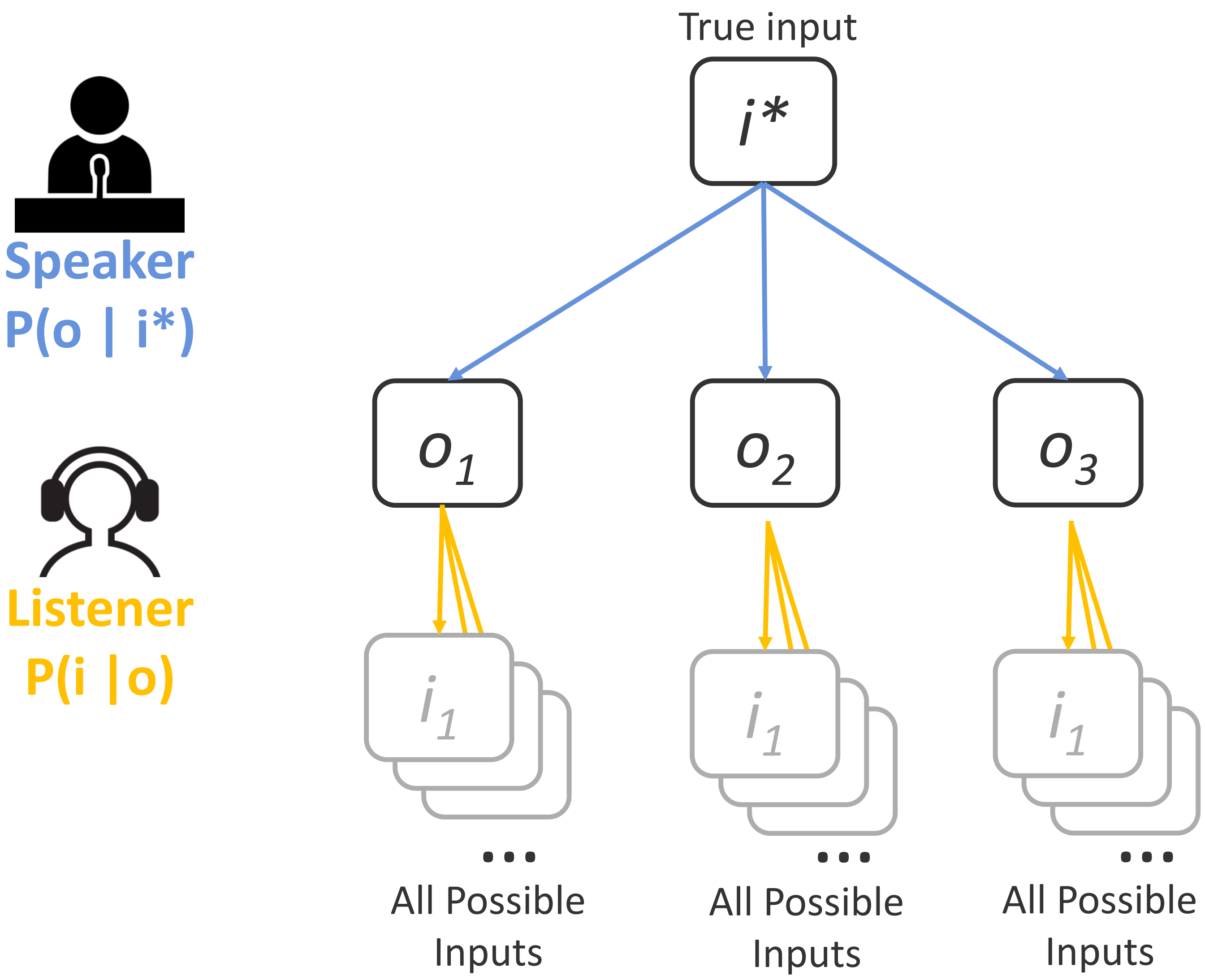
o_1

i_1

…

All Possible Inputs

Reconstructor-Based Pragmatics



Speaker

P(o | i*)

Listener

P(i |o)

True input

$i*$

o_1 o_2 o_3

i_1 i_1 i_1

…

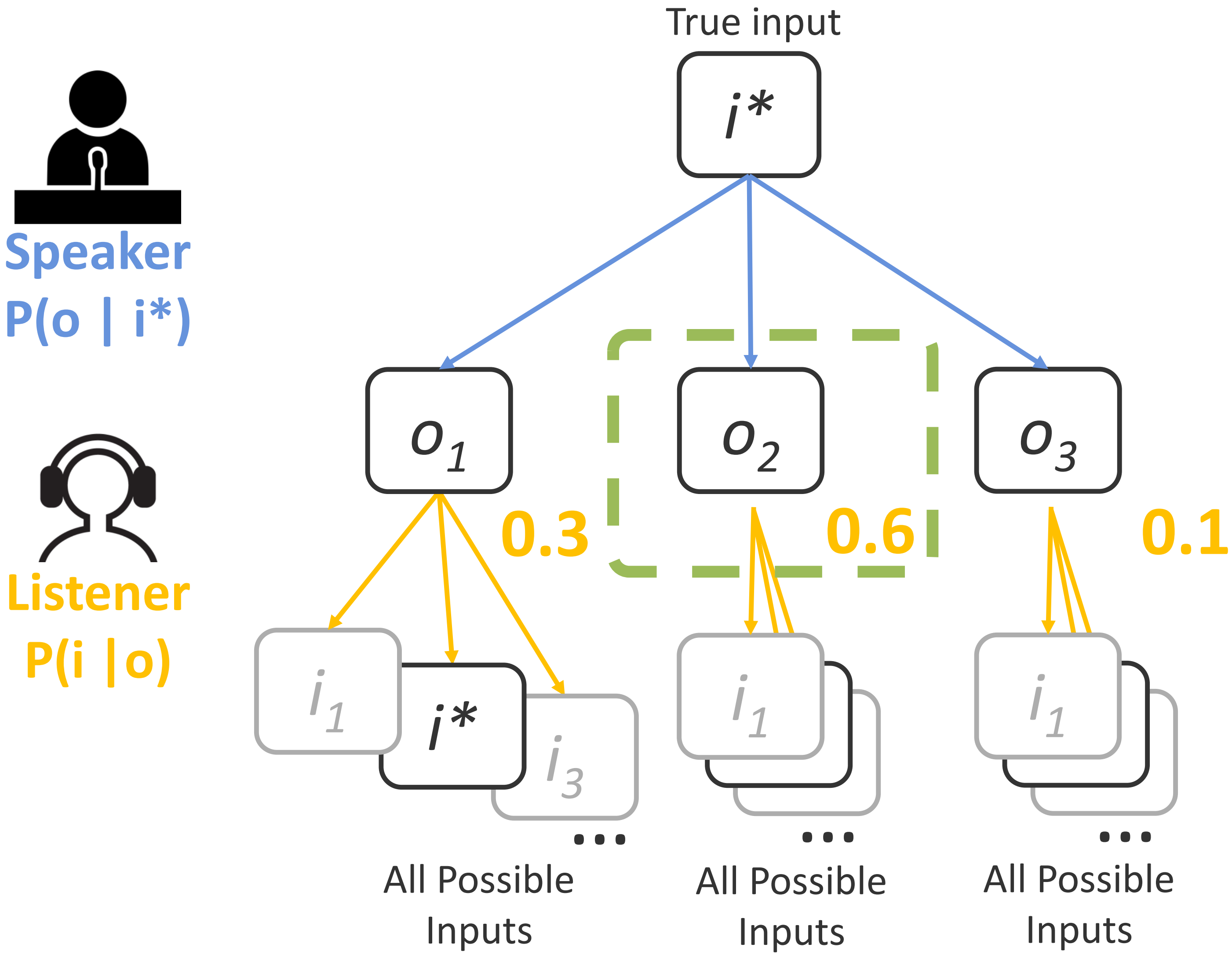
All Possible Inputs

All Possible Inputs

All Possible Inputs

Searching:
Obtain candidate outputs by beam search in P(o | i*)

Reconstructor-Based Pragmatics

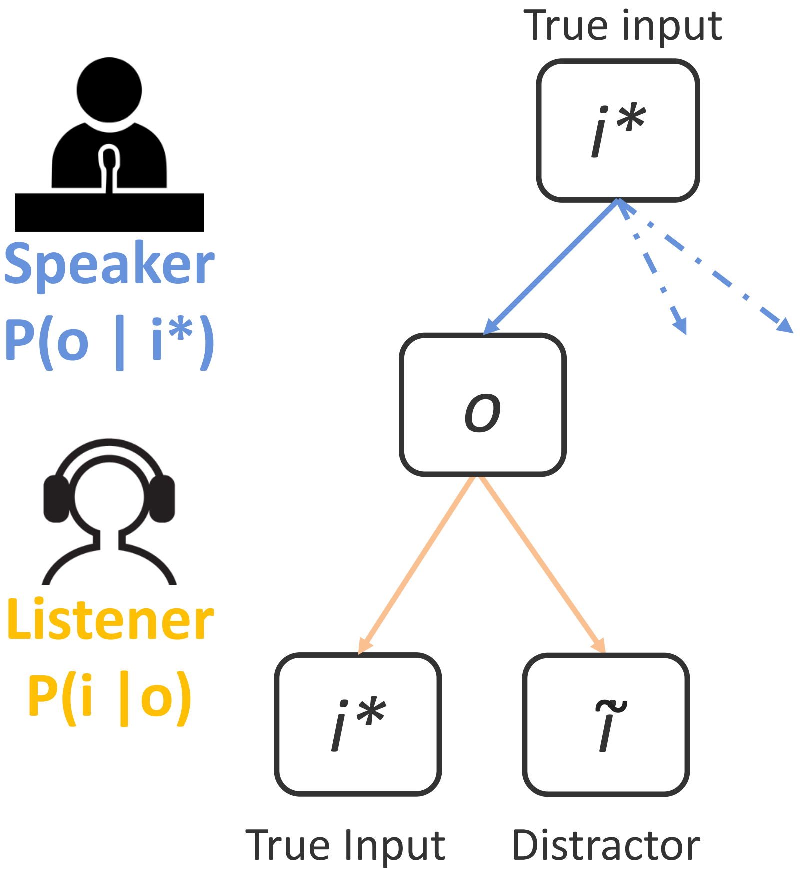


Searching:
Obtain candidate outputs by beam search in P(o | i*)

Scoring:
Score and select o using P(i* | o)

Distractor-Based Pragmatics

When we use a Listener can only produce the true input and a distractor, we can define the Listener using the Speaker and Bayes' rule:



Searching:
Obtain candidate outputs by beam search in **P(o | i*)**

Given by seq-to-seq Speaker

Use a uniform prior

$$P(i^*|o) = \frac{P(o|i^*)P(i^*)}{\sum_{i' \in \{i^*, \tilde{\imath}\}} P(o|i')P(i')}$$

Scoring:
Choose output by argmax_o **P(i* | o)**

Distractor-Based Pragmatics

$$P(i^*|o) = \frac{P(o|i^*)}{\sum_{i' \in \{i^*, \tilde{\imath}\}} P(o|i')}$$

Possible Outputs
(search over these)

o_1: *Fitzbillies is a cheap coffee shop.*

o_2: *Fitzbillies is a cheap* ***English*** *coffee shop.*

...

Inputs

True Input, *i**
Name [Fitzbillies],
Eat Type [Coffee Shop],
Food[English], Price[Cheap]

Distractor, $\tilde{\imath}$
Name [Fitzbillies],
Eat Type [Coffee Shop],
Price[Cheap]

	o_1	o_2	...
True Input, *i**	**0.4**	0.2	...
Distractor, $\tilde{\imath}$	**0.8**	0.05	...

Distractor-Based Pragmatics

$$P(i^*|o) = \frac{P(o|i^*)}{\sum_{i' \in \{i^*, \tilde{\imath}\}} P(o|i')}$$

Possible Outputs

(search over these)

Inputs

	o_1 *Fitzbillies is a cheap coffee shop.*	o_2 *Fitzbillies is a cheap **English** coffee shop.*	…
True Input, i^* Name [Fitzbillies], Eat Type [Coffee Shop], Food[English], Price[Cheap]	0.33	**0.8**	…
Distractor, $\tilde{\imath}$ Name [Fitzbillies], Eat Type [Coffee Shop], Price[Cheap]	**0.66**	0.2	…

Distractor-Based Pragmatics

$$P(i^*|o) = \frac{P(o|i^*)}{\sum_{i' \in \{i^*, \tilde{i}\}} P(o|i')}$$

Possible Outputs

(search over these)

o_1

Fitzbillies is a cheap coffee shop.

o_2

Fitzbillies is a cheap ***English*** *coffee shop.*

…

Inputs

True Input, *i**

Name [Fitzbillies],
Eat Type [Coffee Shop],
Food[English], Price[Cheap]

0.33 **0.8** …

Choose argmax o as the pragmatic output!

Distractor, $\tilde{i}$

Name [Fitzbillies],
Eat Type [Coffee Shop],
Price[Cheap]

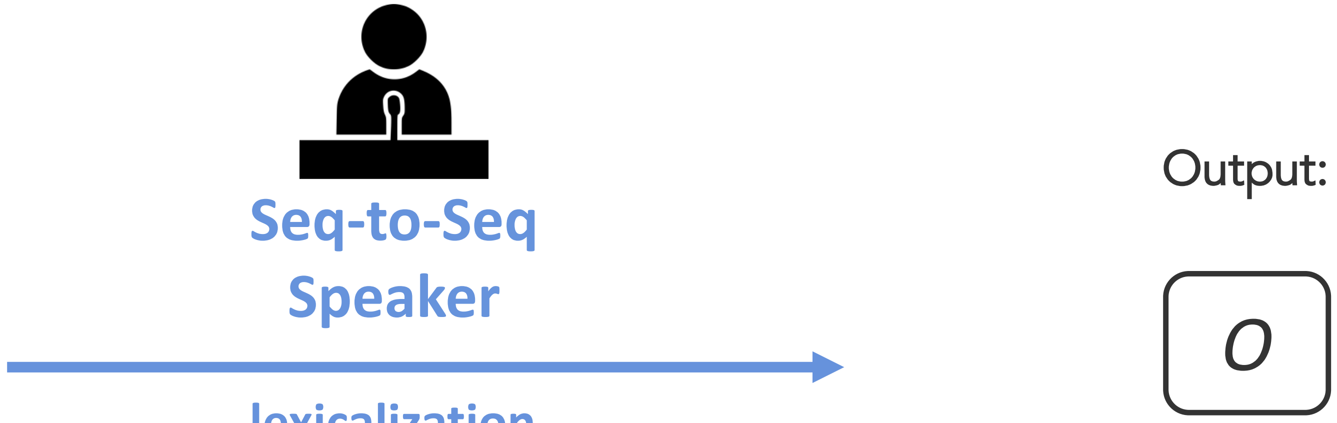
0.66 **0.2** …

In practice: do the search and normalization incrementally, word-by-word. [Cohn-Gordon et al. 2018.]

Generation from Meaning Representations

Input:

```
Name[Fitzbillies],

EatType[Coffee Shop],

PriceRange[Cheap],

Area[Riverside],

Food[English]
```



Seq-to-Seq Speaker

lexicalization

[Puzikov and Gurevych, 2018]

Output:

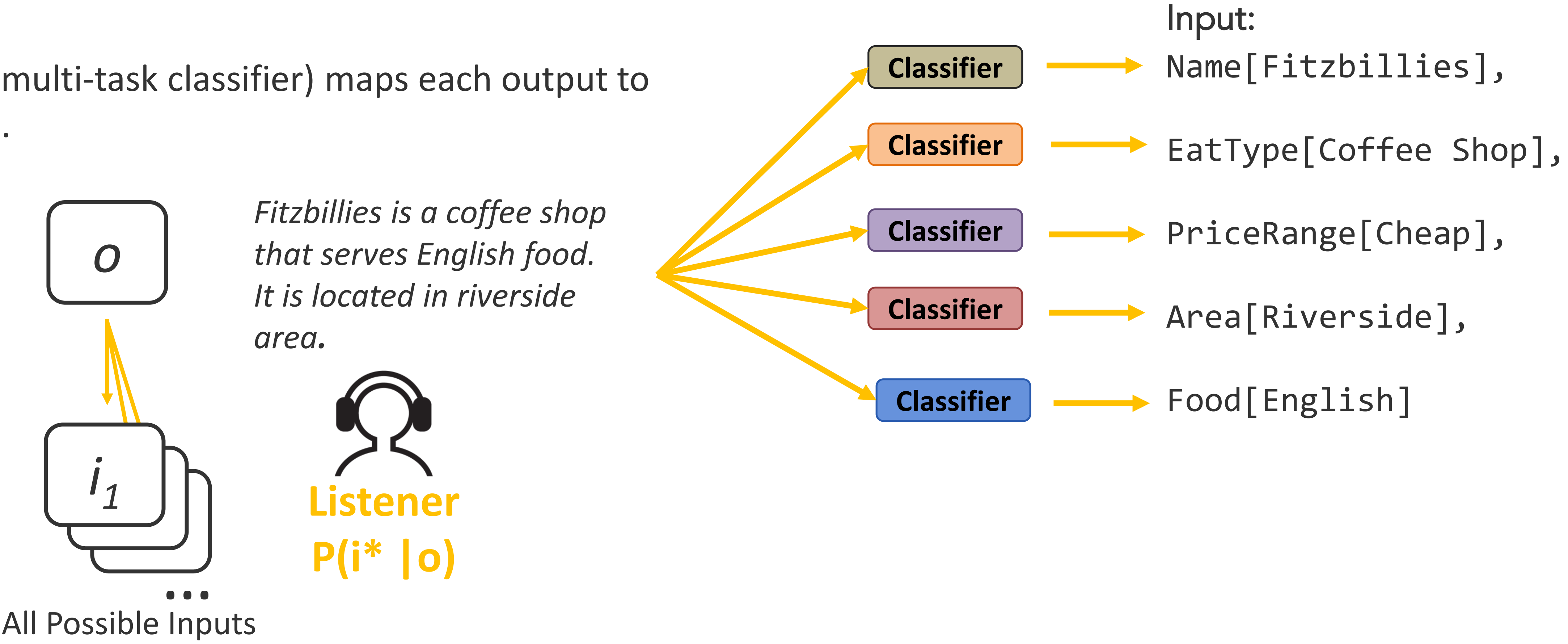
O

Fitzbillies is a coffee shop that serves English food. It is located in riverside area.

Generation from Meaning Representations

Reconstructor:

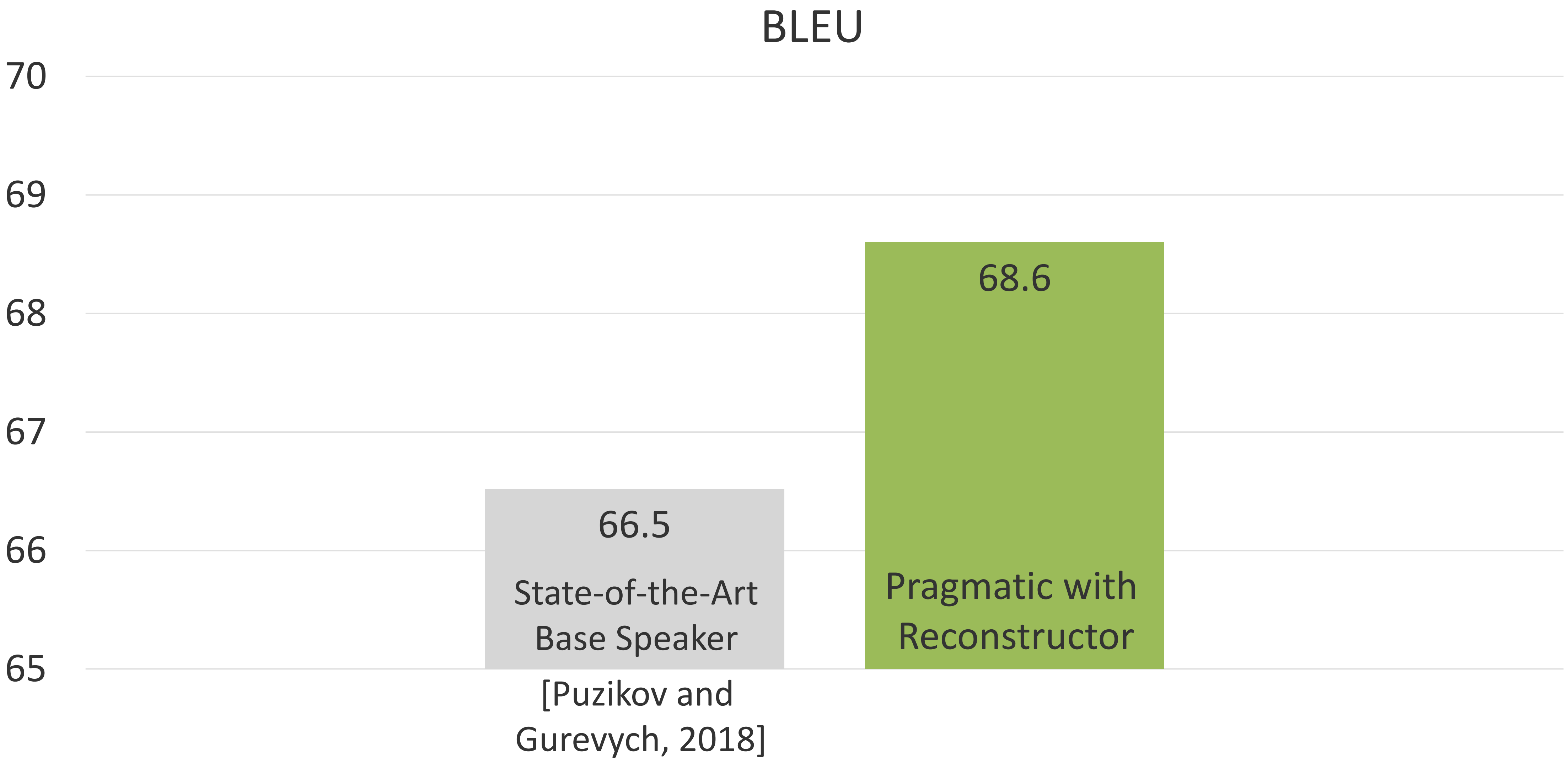
S^R (a multi-task classifier) maps each output to input.



Fitzbillies is a coffee shop that serves English food. It is located in riverside area.

Input:
Name[Fitzbillies],
EatType[Coffee Shop],
PriceRange[Cheap],
Area[Riverside],
Food[English]

Generation from Meaning Representations

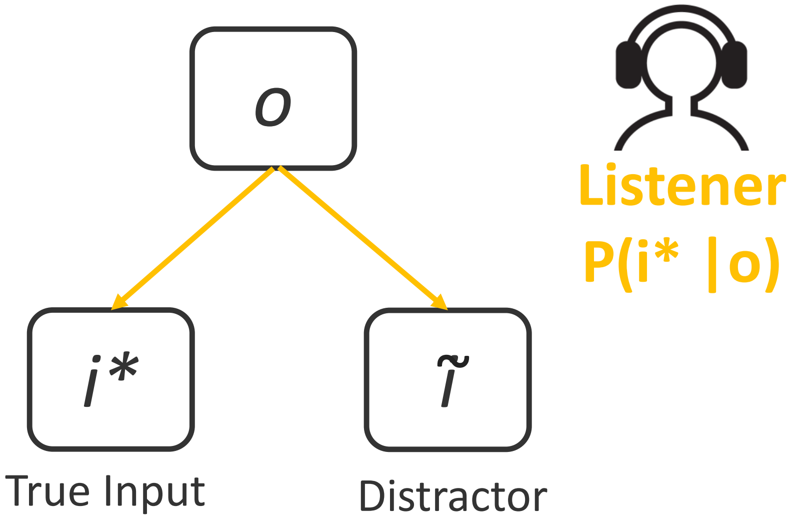


Generation from Meaning Representations

Distractor:

S^D is based on the MR that masks out other attributes.

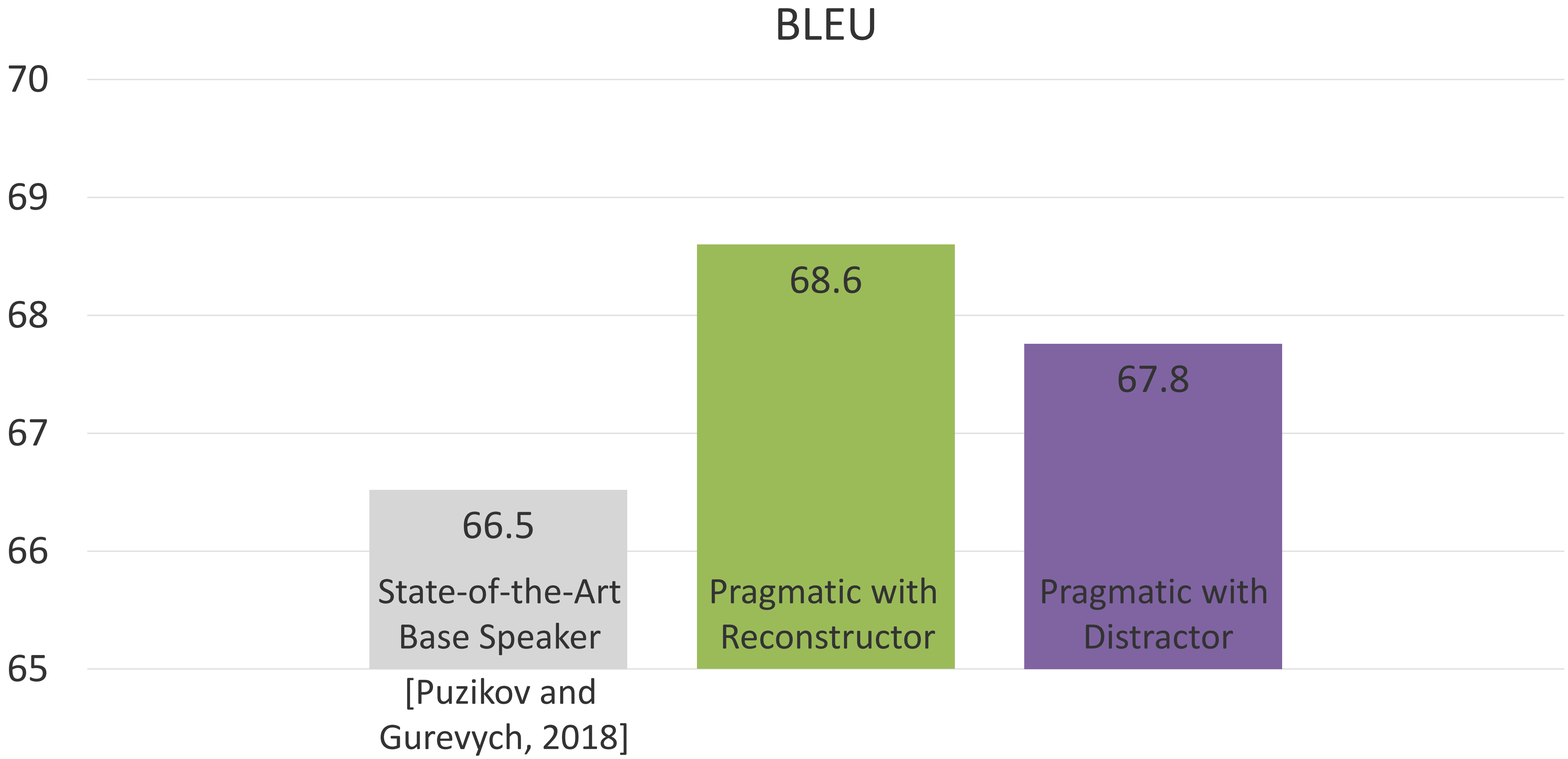
Eg: `Near[Burger King]`



Input:

```
Name[Fitzbillies],

EatType[Coffee Shop],

PriceRange[Cheap],

Area[Riverside],

Food[English]
```

Generation from Meaning Representations

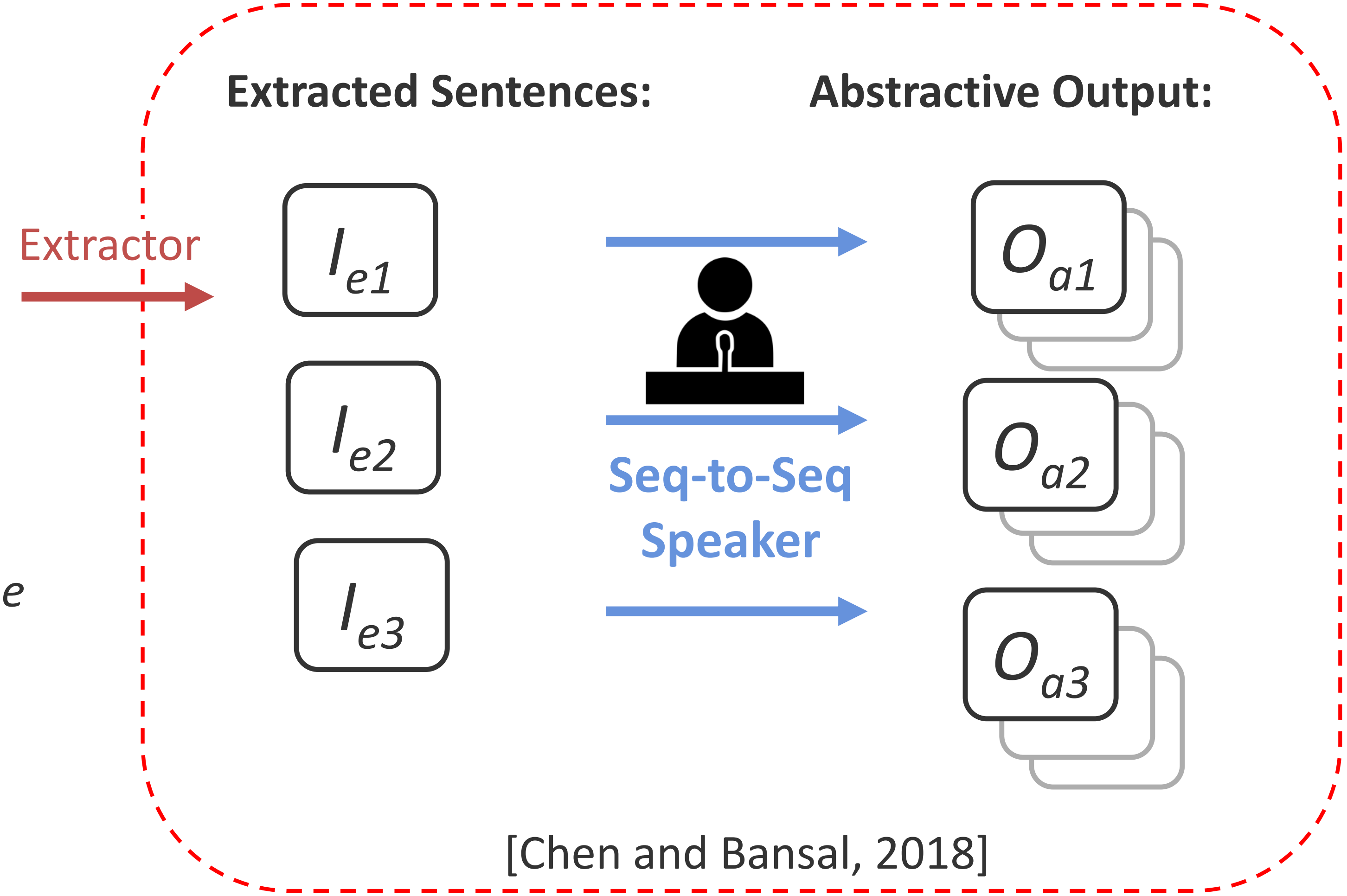


Abstractive Summarization

Long Document:

It is the primary reason all four English teams - Liverpool , Chelsea , Arsenal and Manchester City - were eliminated from the Champions League before the quarter-final draw.

…



[Chen and Bansal, 2018]

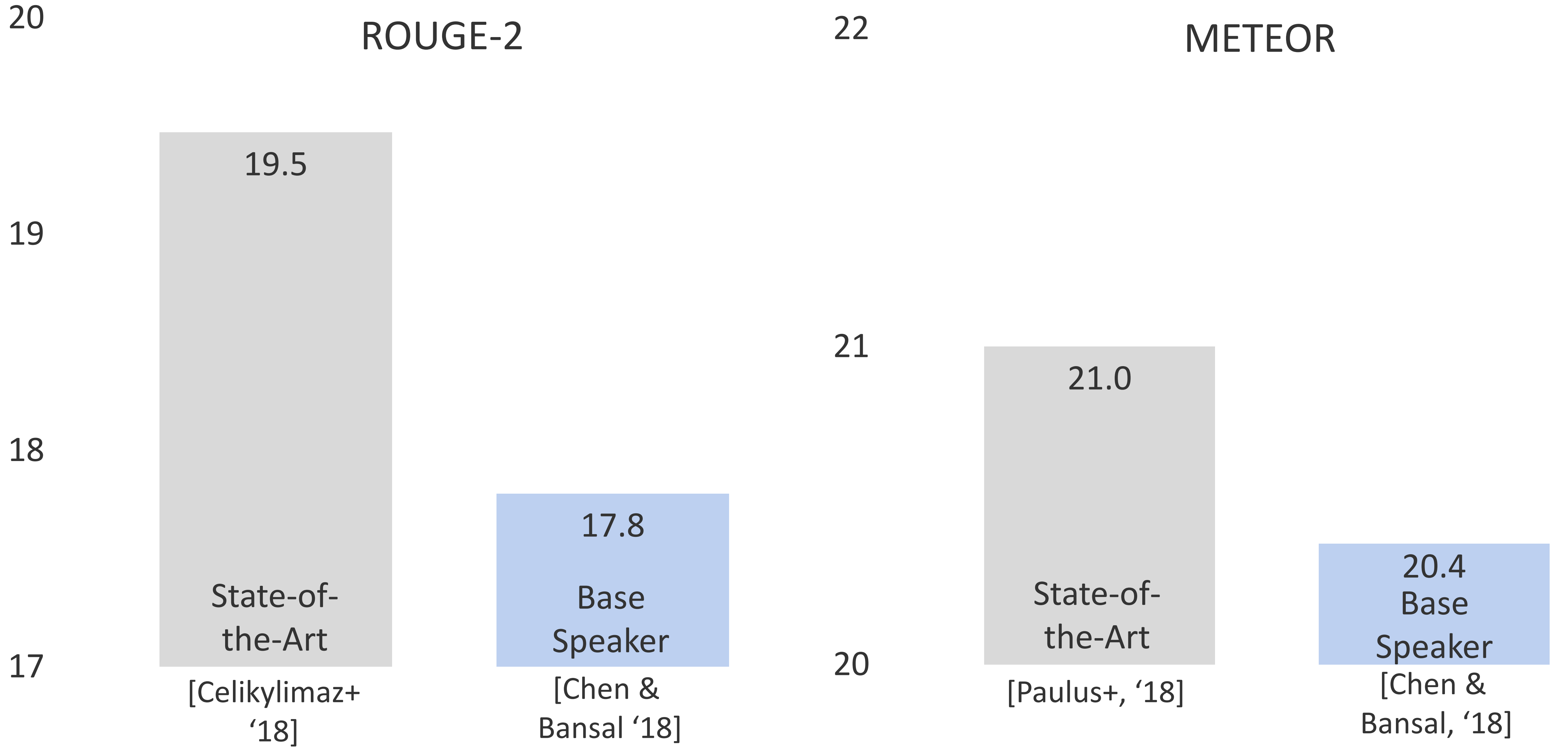
Final Output:

1. Manchester City became the latest team to be eliminated from Europe;

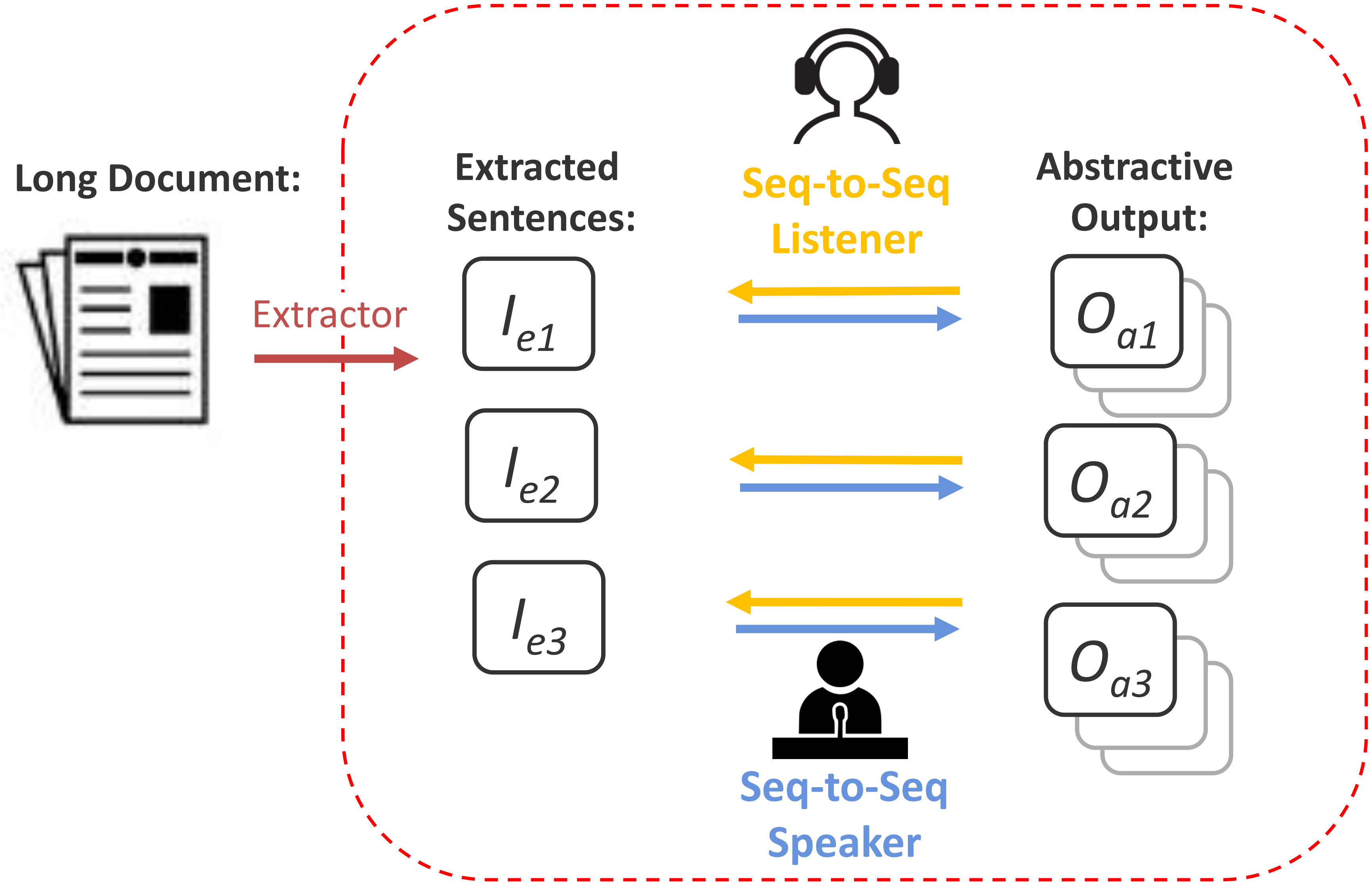
2. City were dumped out of the Champions League last 16 by Barcelona.

3. …

Abstractive Summarization



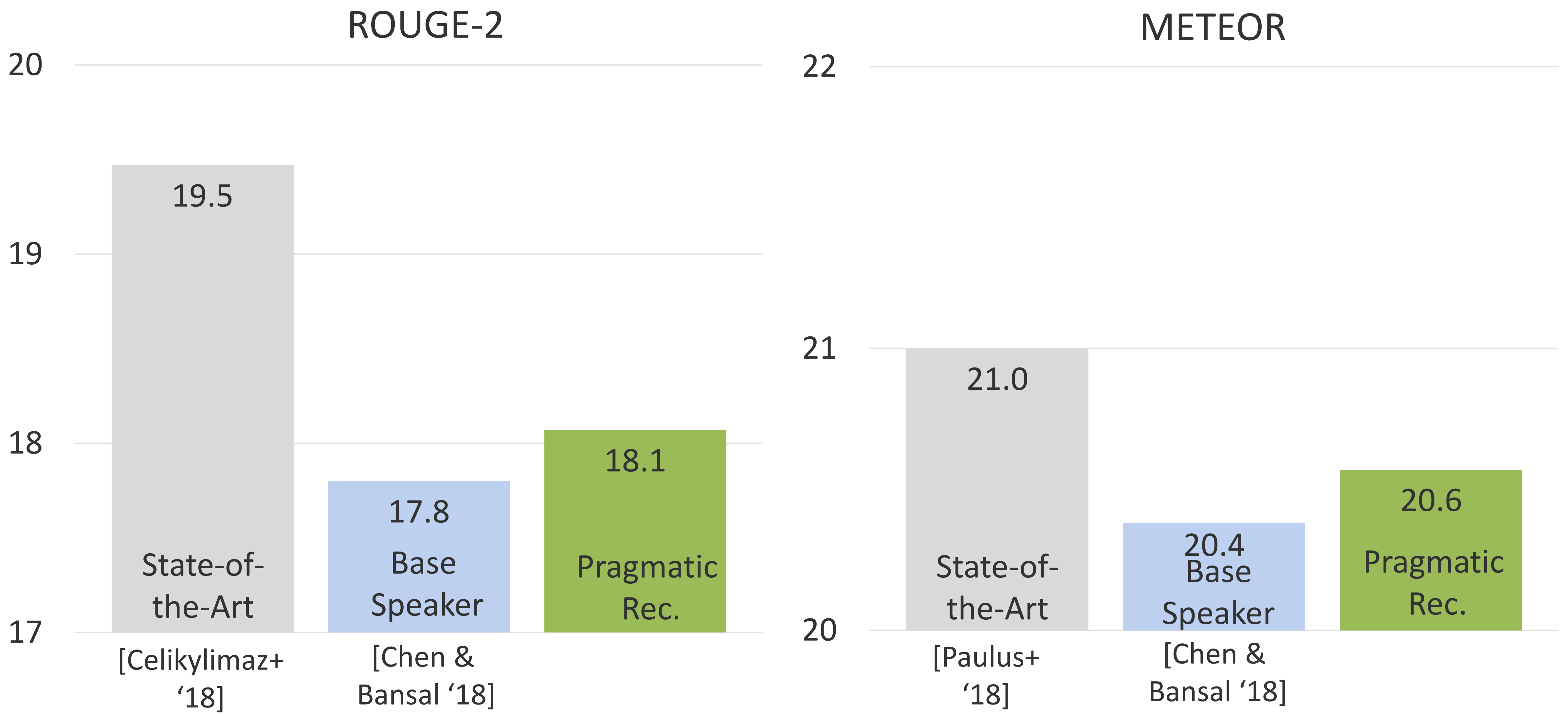
Abstractive Summarization



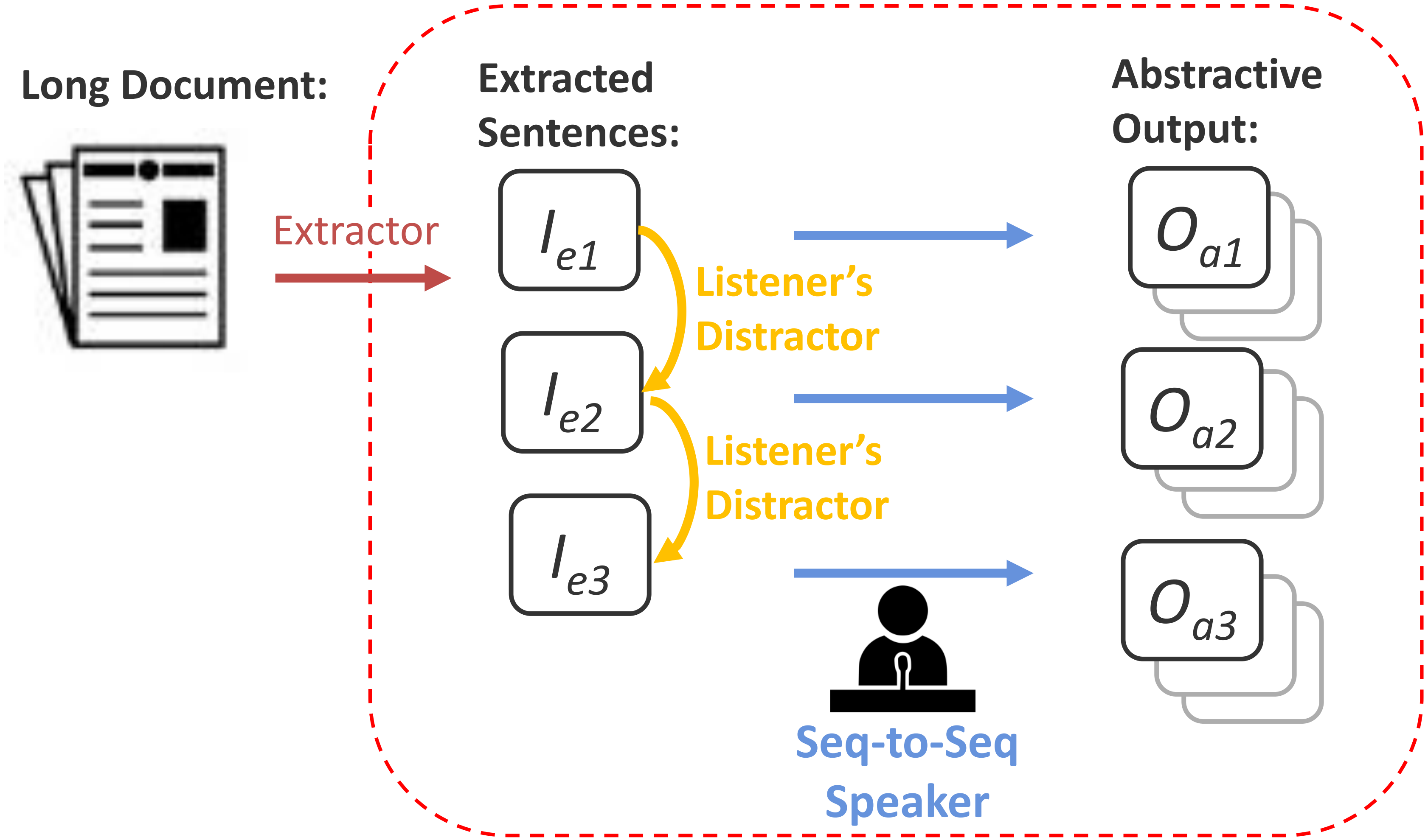
Reconstructor:

S^R (seq-to-seq model) maps abstractive outputs to extractive inputs.

Abstractive Summarization



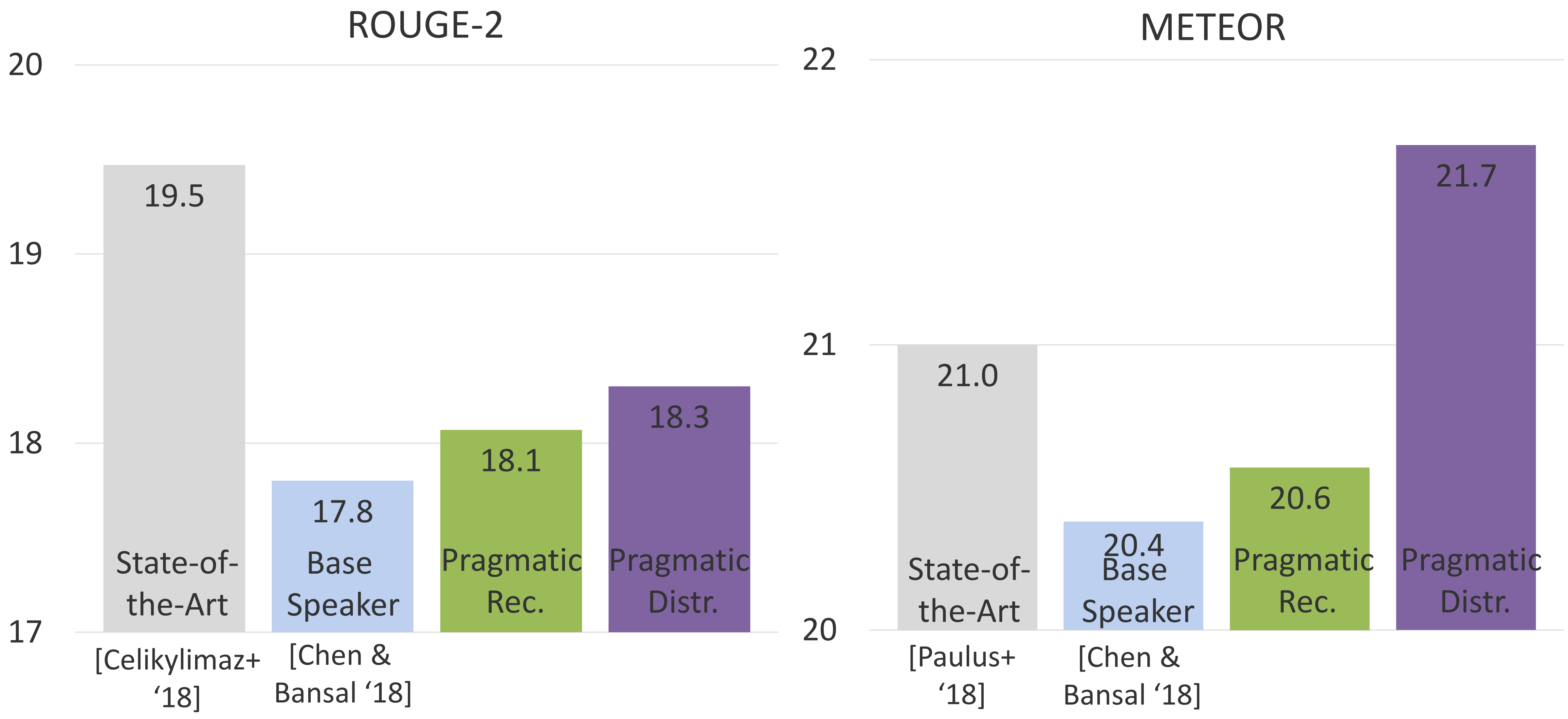
Abstractive Summarization



Distractor:

For a given extracted sentence, use the next extracted sentence as the distractor.

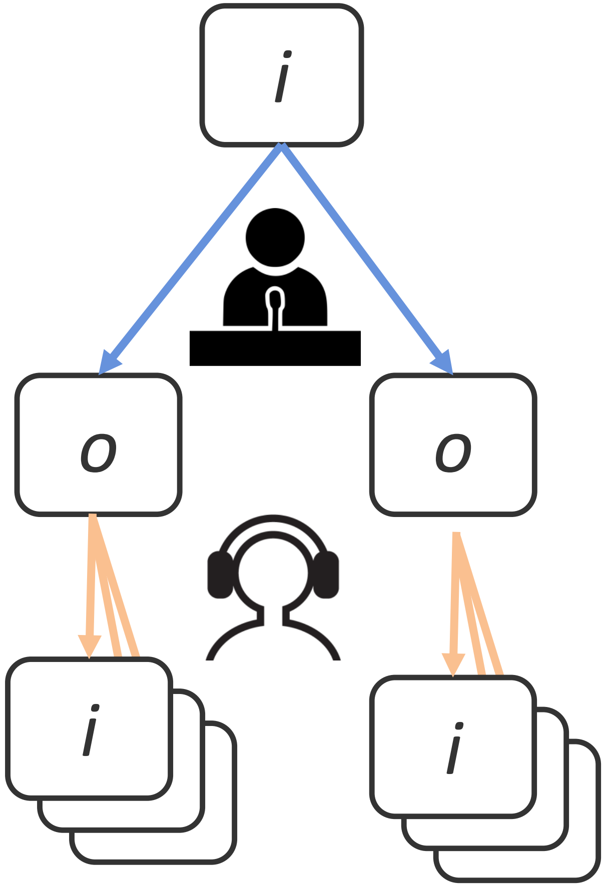
Abstractive Summarization



Conclusions

- Modeling generation as a speaker-listener game leads to more adequate and informative outputs

- Computational pragmatics produces improvements for general text generation tasks



Thanks!

Berkeley

N L P

Our code is publicly available at

`https://github.com/sIncerass/prag_generation`