Knowledge-Driven Slot Constraints for Goal-Oriented Dialogue Systems

Piyawat Lertvittayakumjorn*, Daniele Bonadiman, Saab Mansour
pl1515@imperial.ac.uk, dbonadim@amazon.com, saabm@amazon.com
* Work done while interning at Amazon

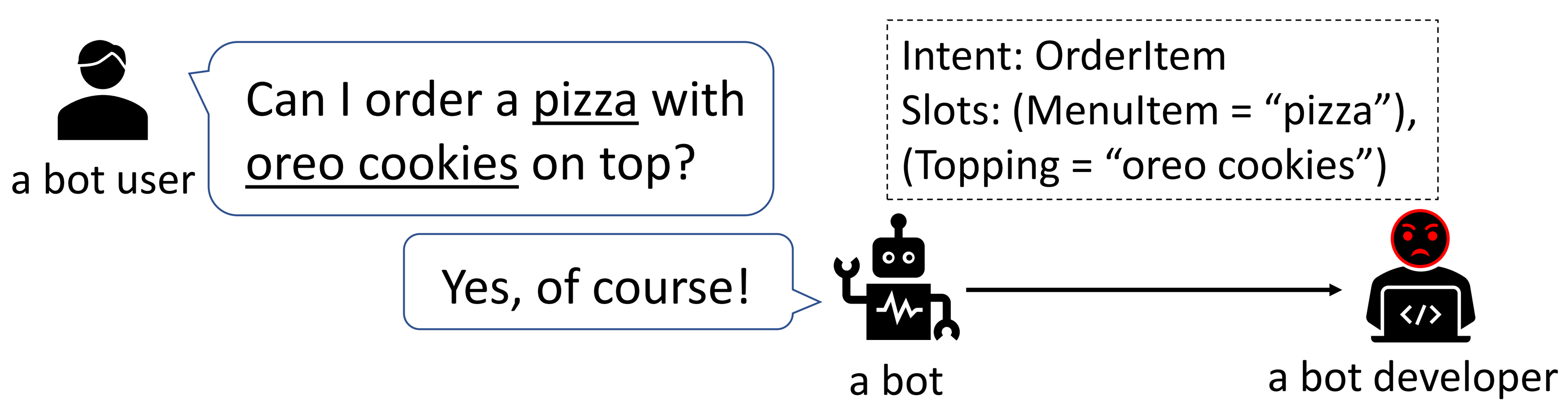
Goal-Oriented Dialogue Systems

- Users provide information through slot values to achieve specific goals.
- The NLU component performs intent classification (IC) and slot labelling (SL)

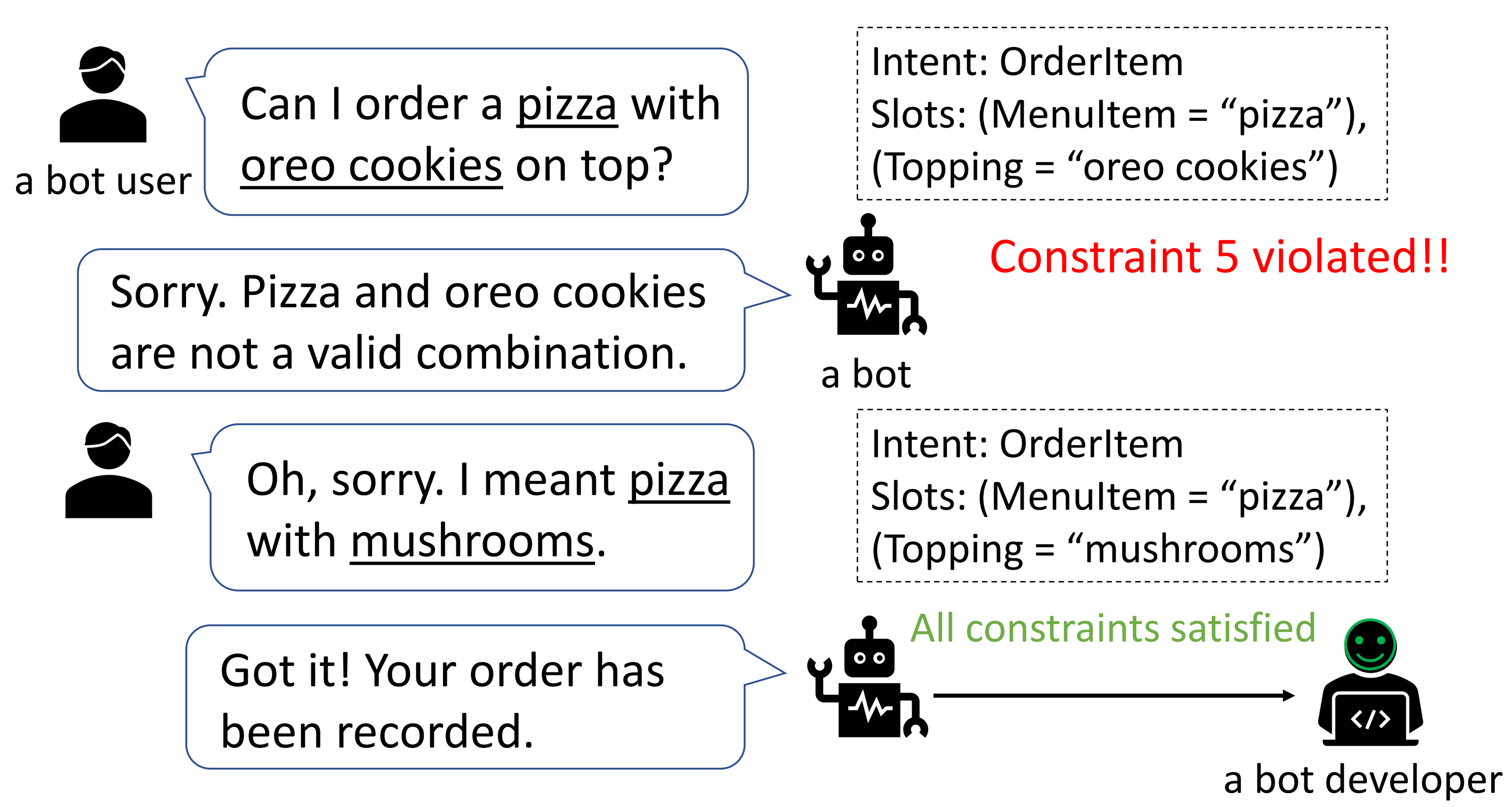
- Intent: GetAllergenInfo
- Slots: (AllergenType = “dairy”), (MenuItem = “Cream cheese bagel”)

Motivation: Invalid slot combinations

- Some combinations of slot values are not valid for the task based on the business logic



- Wouldn’t it be better?



Contributions

- Formal representation of slot constraints and the constraint violation detection task
- Benchmarking data for the task, focusing on constraints on custom slot types
- Three approaches for detecting constraint violations with experiments

Constraint Representation

(A) Input utterance: Please add one XL fries to my order.
Basic NLU output (Intent classification & Slot labelling):
- **Intent**: AddToOrder
- **Slot labels**: Please add [one:Quantity] [XL:MenuItemSize] [fries:MenuItem] to my order.

Dialogue state: d = (AddToOrder, {Quantity: 1, MenuItem: ‘Fries’, MenuItemSize: ‘extra large’})

(B) Constraint $c = (c_i, c_S, c_l)$ with c_i = [AddToOrder], c_S = (MenuItem, MenuItemSize), and c_l = ((MenuItem, =, ‘Cheese burger’) AND (MenuItemSize, in, [‘small’, ‘medium’, ‘large’]))
OR ((MenuItem, =, ‘Lasagna’) AND (MenuItemSize, in, [‘medium’, ‘large’]))
OR ((MenuItem, =, ‘Fries’) AND (MenuItemSize, in, [‘medium’, ‘large’, ‘extra large’]))
OR ((MenuItem, =, ‘Pulled pork’) AND (MenuItemSize, in, [‘small’, ‘medium’]))

- A dialogue state d violates a constraint c if and only if $d_{intent} \in c_i$ and $c_S \subseteq d_{slots}$ but d does not satisfy c_l.

Slot Constraint Violation Detection Task

- Given: a bot schema with constraints, a current utterance, and a conversation history
- Predict: whether the current state of conversation violates any constraints or not and which constraints are violated

Approaches

- **Deterministic Pipeline Approach**
  - IC/SL: JointBERT (Chen et al., 2019)
  - (Open) Entity Linking: Also predict ‘None’ if the slot value cannot be linked to any known entity
  - Deterministic satisfiability check

Approaches (Cont’)

- **Probabilistic Pipeline Approach**
  - We use the probability distribution (via softmax) over the candidate entities (including None) to represent the slot value.
  - Violation score = 1 – Σ Prob of all valid entity combinations
- **End-to-End Approach**
  - MultilabelBERT (# classes = # constraints)
  - Applying a linear layer (with sigmoid function) on top of the embedding vector of [CLS]
  - Learn from training data with violation labels

Experiments & Results

- We modified two domains, insurance and fast food (turn-level annotation), of the MultiDoGO dataset (Peskov et al., 2019) to support violation detection.

Method (Threshold)	Insurance						Fast food					
	Conver. correct	Turn correct	Turn IoU	F1	Precision	Recall	Conver. correct	Turn correct	Turn IoU	F1	Precision	Recall
Deterministic Pipeline Approach (DP)												
Exact match	81.6	89.9	92.4	71.7	62.1	**85.0**	30.7	45.0	59.6	**59.7**	49.1	**76.1**
Bijaccard	74.9	85.6	88.4	39.2	70.6	27.1	39.4	52.2	63.0	51.5	**69.8**	40.8
Levenshtein	73.4	84.6	87.8	40.9	63.3	30.2	34.5	48.5	60.3	51.7	64.2	43.3
NLI	72.8	84.3	87.8	43.6	63.1	33.4	36.7	49.6	59.4	46.2	64.4	36.0
NLI (0.8)	80.5	89.6	91.9	70.1	62.6	79.6	36.7	48.3	61.9	58.2	54.4	62.4
Average	74.3	85.0	88.2	42.3	67.3	30.8	**39.9**	**52.6**	63.3	50.8	68.5	40.3
Average (0.5)	82.2	90.4	92.5	71.6	63.9	81.4	37.4	50.2	63.5	59.5	54.5	65.4
Probabilistic Pipeline Approach (PP)												
Bijaccard	74.1	84.8	88.4	44.6	66.9	33.5	37.7	50.8	62.7	52.4	67.3	42.9
Levenshtein	73.7	84.6	88.0	44.3	63.8	33.9	31.9	46.2	58.4	51.2	62.0	43.5
NLI	70.7	83.1	86.8	44.0	58.7	35.2	34.3	47.0	58.3	49.0	62.3	40.3
NLI (0.8)	70.2	83.8	86.4	60.9	52.6	72.3	36.5	47.9	61.6	58.4	54.7	62.8
Average	73.7	84.7	88.2	45.1	64.4	34.6	35.0	48.7	60.8	52.8	64.0	45.0
Average (0.5)	75.4	85.8	89.3	52.5	57.8	48.1	38.2	50.8	**63.8**	59.0	55.6	63.0
End-to-End Approach (EE)												
End-to-End BERT	**83.9**	**92.1**	**93.4**	**75.1**	**76.2**	74.1	33.3	52.0	62.4	57.4	60.0	55.1

- The pipeline approaches have access to constraints and are more explainable, but prone to error accumulation. Meanwhile, the end-to-end approach is less cumbersome but learns only from data, i.e., have no access to constraints yet