Augmenting Task-Oriented Dialogue Systems with Relation Extraction

Andrew Lee,1 Zhengu Chen,2 Kevin Leach,3 Jonathan Kummerfeld 1

1 University of Michigan
2 Bloomberg
3 Vanderbilt University

ajyl@umich.edu, Zhenguo.Chen@colorado.edu, kevin.leach@vanderbilt.edu, jkummerf@umich.edu

Abstract

The standard task-oriented dialogue pipeline uses intent classification and slot-filling to interpret user utterances. While this approach can handle a wide range of queries, it does not extract the information needed to handle more complex queries that contain relationships between slots. We propose integration of relation extraction into this pipeline as an effective way to expand the capabilities of dialogue systems. We evaluate our approach by using an internal dataset with slot and relation annotations spanning three domains. Finally, we show how slot-filling annotation schemes can be simplified once the expressive power of relation annotations is available, reducing the number of slots while still capturing the user’s intended meaning.

1 Introduction

Dialogue platforms like Watson, Rasa, and Dialogflow have enabled a dramatic increase in the development and use of dialogue systems by bringing together NLP models, data collection and curation, and scalable deployment (Meteer et al. 2019). These platforms all follow the same general framework for interpreting queries, shown in Figure 1 consisting of (1) classification models for domain or intent identification and (2) slot-filling or entity recognition models to identify relevant entities in a query. While these two models can handle a wide range of queries, they are unable to handle more complex queries (Aghajanyan et al. 2020; Davidson, Yu, and Yu 2019; Gupta et al. 2018). Specifically, when a query contains multiple slots with semantic relations, such as those shown in Table 1, intent classification and slot-filling cannot easily capture the necessary information.

Meanwhile, Relation Extraction (RE) has been effectively applied to a range of data sources, such as news articles, encyclopedia entries, and blog posts, and even conversational data (Yu et al. 2020). Recently, conversational semantic parsing (Aghajanyan et al. 2020; Cheng et al. 2020; Andreas et al. 2020) has also built large end-to-end models to extract semantic relations. While these representations are expressive, they have not been integrated into dialogue system platforms because of the difficulty of collecting new domain-specific data.

Figure 1: The standard architecture of deployed dialogue systems. We show how adding relation extraction can enable support for more complex queries.

In this paper, we show how relation extraction can be incorporated into task-oriented dialogue systems. RE is a lightweight but powerful way to extend the capabilities of a dialogue system. In particular, suitable data can be rapidly annotated without extensive training, making domain-specific development feasible.

To evaluate our idea, we use an internal conversational dataset spanning three domains containing multiple slots and relations. We investigate the accuracy, scalability, and generalizability of a relation extraction model within a dialogue system. Finally, we demonstrate the benefits of including an RE model in terms of the dialogue system’s expressive power, meaning the system’s ability to derive new semantic information that was not seen during training. Adding relation extraction to the task-oriented dialogue pipeline will broaden the range of queries these systems can handle and improve their robustness to variation in the ways people express relations in their queries.

2 Related Work

We give a brief overview in various ways in which semantic relations can be extracted in a dialogue setting.

2.1 Relation Extraction

Relation Extraction (RE) models predict labeled links between spans of text. They have been incorporated into a range of downstream applications, including question-answering, ontology population, and knowledge base construction. These have involved applying RE at a range of
scales: sentence-level, bag-level, document-level, and few-shot (Han et al. 2019). Our task is a sentence-level task, considering one dialogue utterance at a time.

Prior work has shown a variety of techniques for learning to do the task, from hand-crafting rules (Hearst 1992), to supervised learning (Wang et al. 2016; Miwa and Bansal 2016), and weak or distant learning (Mintz et al. 2009; Hoffmann et al. 2011; Zeng et al. 2015; Lin et al. 2016). Modeling approaches have also varied, with neural networks dominating recently (Goswami et al. 2020; Christopoulou, Miwa, and Ananiadou 2021). While there is an abundance of choices for models, to demonstrate the benefit of incorporating any RE model, we consider two neural approaches, an LSTM model and a transformer.

2.2 Task-oriented Dialogue Systems

A range of architectures have been explored for task-oriented dialogue systems. Work on modular structures such as TRINDI (Larsson and Traum 2000) and DIPPER (Bos et al. 2003) proposed ways to connect a variety of models with a structured representation of information states. At the other end of the spectrum, end-to-end neural approaches have been explored (Feng et al. 2020; Ham et al. 2020; Hossemini-Asl et al. 2020). However, labeled dependency accuracy is only 78%, while the AMR annotations are for human-robot commands and AMRL obliges to a strict ontology. While Gupta et al. (2018)’s data is the closest to ours, they also take an end-to-end approach for extracting slots and intents. Finally, there has been work on systems that are domain general, such as the TRIPS parser (Allen et al. 2018). These have been applied to biomedical text and blocks world experiments, but not to the task-oriented setting we consider, and doing so would require substantial work to map the domain-general ontology to a specific need.

2.3 Conversational Semantic Parsing

Recently, researchers have turned to applying semantic parsing to build dialogue representations (Aghajanyan et al. 2018; Davidson, Yu, and Yu 2019), and abstract meaning representation (Bonial et al. 2020), such as the Alexa Meaning Representation Language (AMRL) (Kollar et al. 2018). However, labeled dependency accuracy is only 78%, while the AMR annotations are for human-robot commands and AMRL obliges to a strict ontology. While Gupta et al. (2018)’s data is the closest to ours, they also take an end-to-end approach for extracting slots and intents. Finally, there has been work on systems that are domain general, such as the TRIPS parser (Allen et al. 2018). These have been applied to biomedical text and blocks world experiments, but not to the task-oriented setting we consider, and doing so would require substantial work to map the domain-general ontology to a specific need.

Table 1: Examples of challenging utterances across three different domains that contain relations between slots. The output of classification and slot-filling models is not sufficient to correctly handle the examples above, and without incorporating a statistical relation extraction model, developers of a dialogue system need a workaround such as the one we describe in Section 3.1.
3 Limitations of Current Systems

While the combination of intent classification and slot-filling can handle a wide range of queries, they do not always provide enough information to correctly understand a user’s query. Table 1 includes examples of complex queries with relations that the NLU module must be able to recognize to correctly interpret the query. For instance, consider the two example utterances for the gaming domain. The utterances share the same intents and slots, but the addition of the word ‘a’ and the change from plural to singular changes the meaning. In the first, the user is asking for “fire swords” and “fire shields”, while in the second they want “fire swords” and any type of “shield”. As the table shows, relations can capture this difference, by having an edge “fire ↔ shields” in the first, but not the second. Not understanding the distinction in this example may lead to a fairly benign error, but the same problem appears in all applications of dialogue systems. Even with effective intent and slot detection, without accurate relation extraction, the system may incorrectly execute a task.

3.1 Current Approach

In most modular dialogue systems, a user’s query is passed through intent classification and slot-filling models, followed by back-end applications or external APIs, such as querying a database, aggregating results, or executing an order. Given such a system, without a statistical relation extraction model, developers are left with writing rules in the back-end application to capture and process relations between slots.

Within the back-end module, developers can implement heuristics to infer the relations between slots. For instance, a developer could use the slot type, slot values, or slot position to determine the relations (e.g., in a food ordering system, if a numeric slot is immediately followed by a food item slot then make that number specify the quantity of the food item). However, this approach has several shortcomings:

**Accuracy.** The information captured by classifier and slot-filling models may not be sufficient to infer slot relations. Table 1 shows several cases where this is the case, as discussed earlier. Handling these subtle cases would require complex rules in the back-end application, which would not be robust, causing errors in other cases.

**Scalability.** Heuristic relation extraction requires custom development per use case—updating the slots in the slot-filling model would require re-design and re-implementation. This makes it hard to scale such heuristics to multiple tasks or domains.

**Generalizability.** The developer must exhaustively consider and develop all possible slot combinations and relations. This is not generalizable, and complex rules and code would become difficult to maintain as the number of slots and relationships grows.

These shortcomings limit the scope of dialogue systems. As a result, developers on conversational platforms build simpler systems that only handle straightforward queries. Incorporating relation extraction breaks this limitation, enabling support for more complex queries.

4 Relation Extraction Module in a Dialogue System Framework

We integrate relation extraction into a dialogue system by running it after the classification and slot-filling steps. The RE model takes their predictions as input and predicts relations in the user’s utterance. Once the RE model predicts the relations among the slots, all of the extracted information, intent, slots and relations, are passed to the back-end application.

We consider several approaches to RE, from rule-based methods to neural models. While there may be more sophisticated models for RE, we simply use an attention-BiLSTM model and a transformer to demonstrate that incorporating any RE model allows us to handle more complex queries.

**Neural.** We consider two simple approaches that have been applied to relation extraction in other domains: an attention-BiLSTM model (Zhou et al., 2016) and a transformer model (Alt, Hübner, and Hennig, 2019). Both have been the state-of-the-art at one point on the popular SemEval 2010 Task 8 dataset (Hendrickx et al., 2010). For slot-filling we use a BiLSTM (Mesnil et al., 2014).

Like many sentence-level relation extraction approaches (Liu et al., 2018), these two models support classifying the correct relation between two entities within an utterance. However, in a conversational setting, a single utterance can have multiple slots (see Table 1). We use these models to enumerate all the pairs of slots in the query utterance, and treat each slot pair as an independent relation extraction task. For a given slot pair, we annotate the slots within the raw text utterance by adding special tokens (BEGIN_SLOT, END_SLOT) as well as the slot’s label before and after the slot tokens. This enumeration step is not required for models that can identify the correct relation among multiple slots, such as Liu et al. (2018).

**Rule-Based.** We also consider a baseline that is indicative of the approach used when there is no machine learning based relation extraction module in the NLU pipeline (Section 3.1). This heuristic was designed to be generic enough to be applicable across all three of the domains we consider. The only domain-specific information is a set of rules in which each slot is identified as a modifier and/or modified slot (or neither). For each modifier slot type, we identify a list of valid modified slot types and a relation type the slot pair entails. Using this list, we iterate over slots that appear in a query utterance. For each modifier slot \( x \), we determine which (if any) of the other slots \( Y \) are valid modified slots, according to the configured rules. If a single modified slot \( y \in Y \) exists, we assign the specified relation between \( x \) and \( y \). When multiple \( y_i \in Y \) exist, we select the slot in \( Y \) that appears nearest to \( x \) in the utterance.

5 Evaluation

In this section, we describe our internal dataset and how we measure the performance of our proposed approach.
5.1 Conversational Data with Relation Annotation

The dataset that was used for our evaluation consists of utterances repurposed from slot-filling data in a production dialogue system. The relations amongst slots were annotated manually using a custom annotation tool. Details about the dataset, including the number of utterances, the domains, slots, and relations are described in our supplementary material.

For some of the domains, extra crowdsourced data or new slot types were added to introduce more examples of relations. In all cases, the relations were manually annotated.

Stocks. This data was originally collected for a heuristic RE approach (Section 3.1). As a result, the types of queries handled in the task were intentionally chosen to be solvable without relation extraction. This makes it a relatively difficult case for our approach to show improvements. There are two versions of this dataset: one with the original slot labeling scheme with no relation annotation, and one with the same utterances but a new slot labeling scheme as well as relation annotations (described in Section 6).

Food-Order. These utterances were repurposed from an existing slot-filling dataset, with added crowdsourced data to include more utterances with relations and to provide a challenging benchmark. When collecting new data, we provided workers with examples and asked them to write paraphrases, then replaced slot values with new options from a list of food menu items to add diversity to the slots. To ensure the utterances were diverse we sampled them in a two step process. First, we grouped utterances based on the pattern of slots they contain (e.g., [ QUANT, PLUS ] for “Give me { QUANT 3 } { PLUS burgers }”). Next, we randomly selected a set of slot patterns and randomly picked a set of utterances for each pattern.

Gaming. This dataset was built similarly to the food-ordering dataset, but with extra slots and example utterances added prior to crowdsourcing.

5.2 Results

We seek to develop RE models that are highly accurate, scale well to complex utterances, and generalize well to sentence structures not seen in training.

To measure these characteristics, we evaluate the $F_1$ score and exact match score across several experiments. The exact match score indicates the percentage of utterances in which every relation in the utterance is correctly predicted. These scores measure the RE models’ predictions in isolation (they are provided with oracle slot-filling and intent classification output). For each neural network, we used the hyperparameters specified in the original work across all experiments (we verified they were effective using a development set in each domain). We also use the development dataset for early stopping during training.

Accuracy. Table 2 presents $F_1$ and exact match scores for all three approaches on all three datasets, averaged over five runs. The consistently high $F_1$ and exact match scores indicate that these models are appropriate in task-oriented dialogue systems. Moreover, while additional developer effort could increase the heuristic performance, our approach requires no such effort.

Scalability. We consider scalability to be our ability to identify relations in increasingly complex utterances. First, we split the test set based on the number of slots per utterance. We then measure each model’s performance on each subset of the test data. The $F_1$ and exact match scores averaged across five runs are presented in Figure 2. As the test utterances increase in complexity, the rule-based approach quickly deteriorates in performance, while the neural-network approaches maintain robust performance.

Generalizability. Lastly, we follow the query-split idea from Finegan-Dollak et al. (2018) to evaluate the generalizability of the RE approaches by testing on sentence structures not seen during training. For each dataset, we created new train-test splits based on slot patterns: the order in which slot labels appear in an utterance. We grouped all of the utterances by their patterns and randomly placed each group in either the train or test set. This meant that none of the patterns of slots in test set utterances are seen in training.

We repeated this experiment seven times in each domain for each RE approach. The resulting $F_1$ scores are shown in Figure 3. We prefer a model that consistently generalizes well (i.e., tightly distributed $F_1$ scores). While the neural networks seem to generalize to new slot patterns better than the heuristic approach, some of the domains have a wide range in $F_1$ scores across all three RE approaches. We hypothesize that either (1) some slot patterns have difficult relations to identify, or (2) some slot patterns are crucial for the model to see during training to generalize well.

Note that the stocks dataset was originally crafted to perform well with a heuristic RE approach (Section 5.1), and thus performs similarly to the other RE approaches in all three experiments.

6 Expanding Expressive Power Using Slots And Relations

Once we have a pipeline that includes a relation extraction model in addition to intent classification and slot-filling, we

|                  | Food          | Gaming       | Stocks        |
|------------------|---------------|--------------|---------------|
| $F_1$            | $F_1$         | $F_1$        | $F_1$         |
| EM               | EM            | EM           | EM            |
| Heuristic        | 88            | 79           | 98            | 94            |
| Att-BiLSTM       | 96            | 92           | 97            | 93            |
| Transformer      | 97            | 94           | 98            | 96            |
|                  | 99            | 98           | 99            | 98            |

Table 2: The $F_1$ and exact match scores for each RE approach for three domains. The exact match score is the percentage of utterances in which all relations were correctly identified. There are 227, 114, and 189 test utterances for each domain. Bold indicates the highest score in each column.
Labeling Schemes

| Slot-Based       | Show me all the companies in Europe outside of Germany. |
|------------------|--------------------------------------------------------|
| Relation-Based   | Show me all the companies in Europe outside of Germany. |

| Slot-Based       | Show me the EBITDA of companies that have a market cap over a million dollars and revenue less than 2 million? |
|------------------|------------------------------------------------------------------------------------------------------------------|
| Relation-Based   | Show me the EBITDA of companies that have a market cap over a million dollars and revenue less than 2 million? |

Table 3: A standard labeling scheme for slot filling (slot-based), compared to a new labeling scheme with relations incorporated (relation-based). The standard slot-filling scheme requires each slot to capture both the meaning of the slot tokens as well as the meaning of their context. The new approach allows the labeling scheme for slot-filling to be simplified, while the relation extraction model now captures the contextual information.

can revisit previous design decisions. In particular, to get around the lack of a relation extraction model, many slot-filling schemes are designed with variations on slots that actually capture contextual information. In this section, we explore how slot-filling schemes can be simplified without a decrease in performance, once a relation extraction model is available.

Consider the examples in Table 3. The first example uses location_inside and location_outside slots to distinguish locations to include or exclude, while the second uses filter_amount_above and filter_amount_below. Note that these slots are capturing not only the meaning (location, amount) of the slot value, but also the context (inside versus outside, above versus below).

Instead, we propose a simplified slot-filling labeling scheme in which we eliminate contextual meanings per slot. For example, we could simplify the slots above to location and amount slots, and then capture relations between those slots with new modifier tokens within the utterances. By using an RE model to identify these logical relations, we can simplify the overall slot-filling model. Moreover, we can potentially increase the model’s generalizability—by learning logical modifiers and existing relationships they have within tokens, the model can predict relations between previously unseen pairs of logical modifiers and slots.

This new labeling scheme allows us to treat slots as operands and relations as operators. These operands and operators give us more expressive power by representing any utterance as a logical expression. For instance, any location slot that does not have a relation with any negation_modifier would be equivalent to the location_inside slot, and those that do have a relation...
would be equivalent to the location_outside slot.

This expressive power also gives the NLU module flexibility to be improved, because we no longer need training data for every possible combination of a slot and relation. For instance, imagine a dialogue system that has been trained for a location_outside slot and a location_inside slot. However, once this system is deployed in production, a user may ask about “neighboring locations” (“What are the top companies in the area surrounding Boston?”). Similarly, while a dialogue system knows to recognize filter_amount_above or filter_amount_below slots, a user may ask about an amount that is “equal to”, “greater than or equal to”, or “less than or equal to” a certain value. This requires new training utterances to be collected to cover every possible variation for each slot, and hinders the flexibility and scalability of slot-filling models. On the other hand, if the slot-filling model is able to extract both the simplified slot and the operational tokens, and the relational models are able to extract the correct relations between new slot pairs, new semantic information can be derived.

6.1 Results: Logical Expressions

To evaluate the approach of using slots as operands and relations as operators, we first manually annotate each utterance in the test set of the stocks domain into a set of “operations.” These operations represent the tasks that the back-end application needs to execute.

Figure 4 contains examples of operations and a summary of the two query pipelines that we compare: (1) the approach without a relation extraction model, under the standard (slot-based) labeling scheme for slot-filling, and (2) our approach with a simplified (relation-based) slot-filling model, followed by a RE model. The operation annotations allow us to compare the end-to-end performances of the two approaches (the $F_1$ scores across the three different NLU models in the two query pipelines are not directly comparable, as they are each learning different tasks). In this case, the exact match score indicates the percentage of utterances in which all the correct operations were derived.

**Accuracy.** Table 4 presents the $F_1$ scores and the exact match scores of the two approaches. The use of logical expression has a higher end-to-end performance, and given the similarity in performance between the two slot-filling models, the improvements are most likely coming from the high performance of the RE model.

**Scalability.** In this case we consider scalability to be the system’s ability to be flexible by increasing its expressive power. We demonstrate the scalability of our approach by building a new test set in the stocks domain in which all examples contain a new slot sector_outside that never occurs in the training data. However, the training data contains two similar slots, one with the same slot type (sector) and another that involves negation (location_outside). This allows the slot-filling model to recognize the sector slot and for the RE model to capture the contextual meaning (negation_relation), in turn allowing the whole system to derive the previously-unseen semantic information (sector_outside).

Table 5 presents $F_1$ and exact match scores of the RE
Table 5: Accuracy comparison ($F_1$ and exact match scores) of the two query pipelines (see Figure 4) on a test set that contains new slot (sector outside) that was never seen during training. s.b. (slot-based) SF and s.b. E2E indicate the slot-filling and end-to-end performance of the slot-based query pipeline. r.b. (relation-based) SF, RE, and r.b. E2E indicate the slot-filling, RE, and end-to-end performance of the new query pipeline. The difference between the s.b. and r.b. approaches is in how the utterances are annotated (see Table 3). The attention-BiLSTM was used for the RE model in this table. Transformer accuracy is elided, but yielded similar results.

Table 6: $F_1$ scores on a relation (enchantment) seen between a new pair of slots. The RE models originally only sees examples of the enchantment relation between enchantment ↔ item slots during training, but are tested on the same relation between a new slot pair, enchantment ↔ monster. As training examples with this relation in the new slot pair are introduced, the models converge to high $F_1$ scores quickly.

Table 6 similarly demonstrates that a RE model can generalize and predict the correct relation between slot pairs that were not seen during training. In this experiment, we created a new train and test split of the gaming domain dataset, such that during training, the model sees examples of the enchantment relation between the enchantment ↔ item slot pairs, but never between the enchantment ↔ monster slot pairs. In contrast, the new test data contains the enchantment relation between the enchantment ↔ monster slot pair in every utterance. The $F_1$ scores with zero additional training data indicate that the models can still identify the correct relation between slot pairs that were never seen with a relation during training. They also converge with high accuracy with small amounts of additional training data. Similar results are shown in Table 5 for the negation relation for the relation extraction model (far right column).

7 Conclusion

In this paper we propose the addition of a relation extraction model to the NLU pipeline in modular task-oriented dialogue systems. As part of evaluating the use of relation extraction, we manually annotated an internal dataset from a production dialogue system across 3 different domains that include slot and relation annotations. We examine the benefits of incorporating a relation extraction model by presenting a set of evaluations that cover accuracy, scalability, and generalizability. We further introduce and evaluate the notion of building logical expressions with the use of slots and relations, which can lead to more expressive power, allowing the NLU module to be more flexible and scalable to future changes.

References

Aghajanyan, A.; Maillard, J.; Shrivastava, A.; Diedrick, K.; Haeger, M.; Li, H.; Mehdad, Y.; Stoyanov, V.; Kumar, A.; Lewis, M.; et al. 2020. Conversational Semantic Parsing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 5026–5035.

Allen, J. F.; Bahkshandeh, O.; de Beaumont, W.; Galescu, L.; and Teng, C. M. 2018. Effective Broad-Coverage Deep Parsing. In AAAI.

Alt, C.; Hübner, M.; and Henning, L. 2019. Improving relation extraction by pre-trained language representations. arXiv preprint arXiv:1906.03088.

Andreas, J.; Buse, J.; Burkett, D.; Chen, C.; Clausman, J.; Crawford, J.; Crim, K.; DeLoach, J.; Dorner, L.; Eisner, J.; et al. 2020. Task-Oriented Dialogue as Dataflow Synthesis. Transactions of the Association for Computational Linguistics, 8: 556–571.
Miwa, M.; and Bansal, M. 2016. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 1105–1107. Berlin, Germany: Association for Computational Linguistics.

Wang, L.; Cao, Z.; de Melo, G.; and Liu, Z. 2016. Relation Classification via Multi-Level Attention CNNs. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 1298–1307. Berlin, Germany: Association for Computational Linguistics.

Yu, D.; Sun, K.; Cardie, C.; and Yu, D. 2020. Dialogue-Based Relation Extraction. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 4927–4940.

Zeng, D.; Liu, K.; Chen, Y.; and Zhao, J. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 1753–1762.

Zhang, N.; Deng, S.; Sun, Z.; Wang, G.; Chen, X.; Zhang, W.; and Chen, H. 2019. Long-tail Relation Extraction via Knowledge Graph Embeddings and Graph Convolution Networks. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 3016–3025. Minneapolis, Minnesota: Association for Computational Linguistics.

Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; and Xu, B. 2016. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 207–212. Berlin, Germany: Association for Computational Linguistics.