Natural Language Model Re-usability for Scaling to Different Domains

Young-Bum Kim, Alexandre Rochette and **Ruhi Sarikaya**
Microsoft Corporation, Redmond, WA
ybkim,alrochet,ruhi.sarikaya@microsoft.com

Abstract

Natural language understanding is the core of the human computer interactions. However, building new domains and tasks that need a separate set of models is a bottleneck for scaling to a large number of domains and experiences. In this paper, we propose a practical technique that addresses this issue in a web-scale language understanding system: Microsoft's personal digital assistant Cortana. The proposed technique uses a constrained decoding method with a universal slot tagging model sharing the same schema as the collection of slot taggers built for each domain. The proposed approach allows reusing of slots across different domains and tasks while achieving virtually the same performance as those slot taggers trained per domain fashion.

1 Introduction

Recently there has been tremendous investment into the personal digital assistants by big technology companies (Sarikaya, 2015; Sarikaya et al., 2016). Apple's SIRI, Google Now, Microsoft's Cortana and Amazon's Alexa are examples of such systems. Natural language understanding (Gupta et al., 2006; Tur and De Mori, 2011)is at the core of these systems providing natural communication between the user and the agent. These systems support a number of scenarios including creating reminders, setting up alarms, note taking, scheduling meetings, finding and consuming entertainment (i.e. movie, music, games), finding places of interest and getting driving directions to them. The domains supported by these systems are on the order of tens (not in hundreds) and adding new domains and experiences is a scaling problem that has not been solved yet (Tur, 2006; Jeong and Lee, 2009; El-Kahky et al., 2014; Kim et al., 2015d).

The primary reason behind this is that each domain requires potentially a new schema, intents and slots extracted from the natural language query. That, in turn requires collecting and annotating new data, which is the most expensive step in terms of time and money, and building a new set of domain specific models to support the new domains and scenarios. Slot modeling in particular is the most demanding component in terms of the difficulty of annotation and modeling.

In this study, we propose a new approach that reduces the cost of scaling natural language understanding to a large number of domains and experiences without significantly sacrificing the accuracy. The approach consists of a universal slot tagging method and a runtime technique called *constrained decoding* that performs the decoding according a specific schema. The proposed approach, heavily enables reusing existing slot modeling data that are collected and annotated for potentially different domains and applications, for building new domains and applications. The new domains can be expressed in terms of existing semantic schema.

The rest of the paper is organized as follows. In the next section, we talk about universal slot modeling. In section 3, we present the constrained decoding technique. We describe the experimental set up, results and analysis in section 4 followed by the conclusions and future directions in section 5.

2 Universal Slot Tagging

The first step is to build a universal slot tagger, a single model that can handle all the domains an agent (e.g. Cortana) can support. In Table 1, we show a subset of the domains that are supported by Cortana for experimentation purposes.

2.1 Universal Slot Tagger Training

To train the universal slot tagger, we consider two simple and intuitive approaches: *Binary* and *All-in-one*.

Suppose we have a combined k slots across domains and also have access to the labeled data, *Binary* approach trains k binary classifier one for each slot type. For each binary slot tagger targeting a specific slot type, the labeled data is programatically mapped to create a new labeled data set, where only the target label is kept while all the other labels are mapped "other" label. *All-in-one* approach simply trains a single model by aggregating queries across all domains.

2.2 Slot Tagging Ambiguity

Universal slot tagging model has an advantage, which can share schema across all domains used for training time. In spite of the advantage, there are ambiguity problems caused by combining all domains (and the underlying data) into a single model. The problems can be grouped into two categories:

- *Imbalanced training data distribution*: The amount of training data varies across domains. Universal slot model may have bias towards predicting the slots in domains with larger training data. For example, slots with less training data (e.g. *app_name* in *MEDIACONTROL* domain could be overwhelmed by slots with large training data (e.g. *place_name* in *PLACES* domain).

- *Domain-specific schema:* In practice, the domains the system handles are not constructed at the same time. They are designed for different application back-ends, requirements and scenarios at different points in time. In other words, the semantic schema for a domain is designed without considering other domains. In *ALARM* domain, the slot indicating time is subdivided into sub-slots such as *start_time* representing starting time, *duration* representing duration for an alarm. In contrast, in *PLACES* domain, there is only a single slot indicating time (*time*).

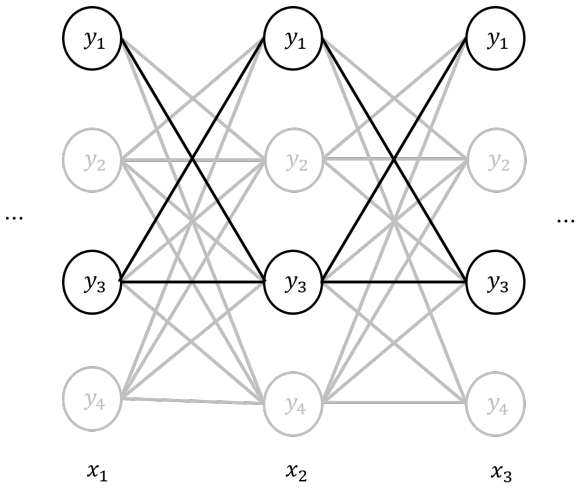

Figure 1: Constrained Lattice: Disabling nodes and transition while decoding the lattice to honor given constraints of domain schema.

3 Constrained Decoding

Slot tagging is considered as a sequence learning problem (Deoras and Sarikaya, 2013; Li et al., 2009; Xu and Sarikaya, 2014; Celikyilmaz et al., 2015; Kim et al., 2015b; Kim et al., 2015c; Kim et al., 2015a). In sequence learning, given a sample query $x_1 \dots x_n$, the decoding problem is to find the most likely slot sequence among all the possible sequences, $y_1 \dots y_n$:

$$f(x_1 \dots x_n) = \arg\max_{y_1 \dots y_n} p(x_1 \dots x_n, y_1 \dots y_n)$$

Here, we assume that output space in training is same as those in test time.

However, in our problem, output (slot) space in test time can be different from those in training time. At test time, we may observe different slot sequences than what is observed in the training time.

This is not an issue for the *Binary* approach, since we can use the output of the selected taggers needed for the new domain. We simply use general decoding approach with each of the selected taggers. Note

that a given query is run as many times as the numbers of slot types covered in a given domain.

For *All-in-One* technique, we consider two possible approaches: *Post-Filter* and *Constrained Decoding*. With *Post-Filter*, we simply provide the best hypothesis generated by the slot tagger that meets the domain schema constraints, by computing the full n-best of slots and filtering out the slot types that do not meet the target domain schema. With *Constrained Decoding*, given a schema $\tilde{y} \subset y$ for the target domain, we first define a constrained lattice *lattice* $\mathcal{Y}(x, \tilde{y}) = \mathcal{Y}(x_1, \tilde{y}) \times \ldots \times \mathcal{Y}(x_n, \tilde{y})$, as shown in Figure 1. Then, we perform the decoding in the constrained lattice:

$$f(x_1 \ldots x_n, \tilde{y}) = \arg\max_{\mathcal{Y}(x,\tilde{y})} p(x_1 \ldots x_n, y_1 \ldots y_n)$$

4 Experiments

In this section, we conducted a series of experiments to evaluate the proposed techniques on datasets obtained from real usage.

4.1 Experimental Setup

To test the effectiveness of the proposed approach, we apply it to a suite of 16 Cortana domains for slot tagging tasks. The data statistics and short descriptions are shown in Table 1. As the table indicates, the domains have different granularity and diverse semantics. Note that we keep domain-specific slots such as alarm_state, but there are enough shared labels across domains. For example, *ALARM* domain, there are 6 shared slots among 8 slots. There are 62 slots appearing more than one domain. Especially, some basic slots such as *time*, *date*,*place_name*,*person_name*,*location* and *product* appear in most domains.

4.2 Slot Taggers

In all our experiments, we trained Conditional Random Fields (CRFs)(Lafferty et al., 2001) and used n-gram features up to n = 3, regular expression, lexicon features, and Brown Clusters (Brown et al., 1992). With these features, we compare the following methods for slot tagging[1]:

- *In-domain*: Train a domain specific model using the domain specific data covering the slots supported in that domain.
- *Binary*: Train a binary classifier for each slot type, combine the slots needed for a given domain schema.
- *Post*: Train a single model with all domain data, take the one-best parse of the tagger and filter-out slots outside the domain schema.
- *Const*: Train a single model with all domain data and then perform constrained decoding using a domain specific schema.

4.3 Results

For the first scenario, we assume that test domain semantic schema is a subset of training domain schema. The results of this scenario are shown in Table 2. We consider *In-domain* as a plausible upper bound on the performance, yielding 94.16% of F1 on average. First, *Binary* has the lowest performance of 75.85%. We believe that when we train a binary classifier for each slot type, the other slots that provide valuable contextual information for the slot sequence are ignored. This leads to degradation in tagging accuracy. *Post* improves F1 scores across domains, resulting into 86.50% F1 on average. Note that this technique does not handle ambiguities and data distribution mismatch due to combining multiple domain specific data with different data sizes. Finally, *Const* lead to consistent gains across all domains, achieving 93.36%, which almost matches the *In-domain* performance. The reason why *Const* performs better than *Binary* is that *Const* constrains the best path search to the target domain schema. It does not consider the schema elements that are outside the target domain schema. By doing so, it addresses the training data distribution issue as well as overlap on various schema elements.

For the second scenario, we consider a new set of test domains not covered in the training set, as shown in Table 3. The amount of training data for the test domains are limited ($< 5K$). These domains lack training data for the *location* and *app_name* slots. When we use universal slot tagger with constrained decoding *Const* yields 94.30%. On average, *Const* increases F1-score by 1.41 percentage points,

[1]For parameter estimation, we used L-BFGS (Liu and Nocedal, 1989) with 100 as the maximum iteration count and 1.0 for the L2 regularization parameter.

	# slots	#shared slots	#train	#test	Description	Example Slots
ALARM	8	6	160K	16K	Set alarms	alarm_state,end_time,title
CALENDAR	21	17	100K	10K	Set meeting in calendar	LOC,time,title,meetingroom
COMM.	21	14	700K	70K	Make a call&send msg	contact,date,title,time
MYSTUFF	20	16	24K	2.5K	find&open a document	contact,data_type
ONDEVICE	10	8	227K	24k	Set up a phone	APP,contact,media_type
PLACES	31	22	478K	45K	Find location & info	LOC,time,route,cuisine
REMIND	17	13	153K	14K	Remind to-do list	LOC,contact,time,date,text
WEATHER	9	5	281K	26K	Ask weather	LOC,map,temperature
TRANSIT	16	16	0	2k	Ask bus schedule & info	APP,LOC,date,time
MEDIACONT.	15	15	0	10k	Set up a music player	rating,LANG,price,APP,title
ENTERTAIN.	18	12	130k	13k	Find&play movie&music	genre,title,price,date,rating
ORDERFOOD	15	15	2.5k	2k	Order food	product,LOC,cuisine,quantity
RESERVATIONS	21	19	3k	2k	Reserve restaurant	LOC,APP,number,product,rating
TAXI	17	17	0	2k	Book a cab	LOC,date,time,company
EVENTS	7	7	2k	1k	Book an event ticket	quantity,LOC,date,title,seat_type
SHOWTIMES	15	15	2k	1k	Book a movie ticket	LOC,date,title,start_time

Table 1: The overview of data we used and descriptions.

Domain	In-domain	Binary	Post	Const
ALARM	**96.24**	76.49	91.86	95.33
CALENDAR	**91.79**	75.62	80.58	90.19
COMM.	**95.06**	84.17	88.19	94.76
ENTER.	**96.05**	85.39	90.42	95.84
MYSTUFF	**88.34**	51.3	80.6	87.51
ONDEVICE	**97.65**	70.16	77.8	96.43
PLACES	**92.39**	75.27	87.63	91.36
REMIND	**91.53**	72.67	88.98	91.1
WEATHER	**98.37**	91.56	92.45	97.73
Average	**94.16**	75.85	86.50	93.36

Table 2: Performance for universal models.

Domain	In-domain	Const
ORDERFOOD	93.62	**95.63**
RESERVATIONS	93.03	**94.58**
EVENTS	92.82	**94.28**
SHOWTIMES	92.07	**92.69**
Average	92.89	**94.30**

Table 3: Performance for prototype domains.

	TAXI	TRANSIT	MEDIAC.	AVG.
Const	90.86	99.5	93.08	94.48

Table 4: Results across new domains.

corresponding a 20% decrease in relative error. We believe that universal slot tagger learns to tag these slots from data available in *PLACES* and *ENTERTAINMENT* domains.

For the last scenario shown in Table 4, we assume that we do not have training data for the test domains. The *Const* performs reasonably well, yielding 94.48% on average. Interestingly, for the *TRANSIT* domain, we can get almost perfect tagging performance of 99.5%. Note that all tags in *TRANSIT* and *TAXI* domains are fully covered by our universal models, but the *MEDIACONTROL* domain is partially covered.

4.4 Discussion

By using the proposed technique, we maximize the reuse of existing data labeled for different domains and applications. The proposed technique allows mixing and matching of slots across different domains to create new domains. For example, we can tag the slots in the *SHOWTIMES* domain, which involves finding a movie to watch by using *movie_titles, actor_names* from the *ENTERTAINMENT* domain, and the location of the theater by using *location, place_name* slots from the *PLACES* domain. If the new domain needs some new slots that are not covered by the universal tagger, then some examples queries could be annotated and added to the universal slot tagger training data to retrain the models. Instead of maintaining a separate slot tagger for each domain, one needs to maintain a single slot tagger. The new slots added can be used by future domains and applications.

5 Conclusions

We proposed a solution for scaling domains and experiences potentially to a large number of use cases by reusing existing data labeled for different domains and applications. The universal slot tagging coupled with constrained decoding achieves almost as good a performance as those slot taggers built in a domain specific fashion. This approach enables creation of virtual domains through any combination of slot types covered in the universal slot tagger schema, reducing the need to collect and annotate the same slot types multiple times for different domains. One of the future directions of research is to extend the same idea to the intent modeling, where we can re-use intent data built for different applications and domains for a new domain. Also, we plan to extend the constrained decoding idea to slot tagging with neural networks (Kim et al., 2016), which achieved gains over CRFs.

References

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Asli Celikyilmaz, Dilek Hakkani-Tur, Panupong Pasupat, and Ruhi Sarikaya. 2015. Enriching word embeddings using knowledge graph for semantic tagging in conversational dialog systems. AAAI - Association for the Advancement of Artificial Intelligence, January.

Anoop Deoras and Ruhi Sarikaya. 2013. Deep belief network markov model sequence classification spoken language understanding. In *Interspeech*.

Ali El-Kahky, Xiaohu Liu, Ruhi Sarikaya, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2014. Extending domain coverage of language understanding systems via intent transfer between domains using knowledge graphs and search query click logs. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4067–4071. IEEE.

Narendra Gupta, Gokhan Tur, Dilek Hakkani-Tur, Srinivas Bangalore, Giuseppe Riccardi, and Mazin Gilbert. 2006. The at&t spoken language understanding system. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):213–222.

Minwoo Jeong and Gary Geunbae Lee. 2009. Multi-domain spoken language understanding with transfer learning. *Speech Communication*, 51(5):412–424.

Young-Bum Kim, Minwoo Jeong, Karl Stratos, and Ruhi Sarikaya. 2015a. Weakly supervised slot tagging with partially labeled sequences from web search click logs. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 84–92. Association for Computational Linguistics.

Young-Bum Kim, Karl Stratos, Xiaohu Liu, and Ruhi Sarikaya. 2015b. Compact lexicon selection with spectral methods. In *Proceedings of Association for Computational Linguistics (ACL)*, pages 806–811. Association for Computational Linguistics.

Young-Bum Kim, Karl Stratos, and Ruhi Sarikaya. 2015c. Pre-training of hidden-unit crfs. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 192–198. Association for Computational Linguistics.

Young-Bum Kim, Karl Stratos, Ruhi Sarikaya, and Minwoo Jeong. 2015d. New transfer learning techniques for disparate label sets. In *ACL*. Association for Computational Linguistics.

Young-Bum Kim, Karl Stratos, Minjoon Seo, and Ruhi Sarikaya. 2016. Domainless adaptation by constrained decoding on a schema lattice. In *Proceedings of the International Conference on Computational Linguistics (Coling)*. Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289.

Xiao Li, Ye-Yi Wang, and Alex Acero. 2009. Extracting structured information from user queries with semi-supervised conditional random fields. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*.

Dong C Liu and Jorge Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528.

Ruhi Sarikaya, Paul Crook, Alex Marin, Minwoo Jeong, Jean-Philippe Robichaud, Asli Celikyilmaz, Young-Bum Kim, Alexandre Rochette, Omar Zia Khan, Xiuahu Liu, Daniel Boies, Tasos Anastasakos, Zhalleh Feizollahi, Nikhil Ramesh, Hisami Suzuki, Roman Holenstein, Elizabeth Krawczyk, and Vasiliy Radoste. 2016. An overview of end-to-end language understanding and dialog management for personal digital assistants. In *IEEE Workshop on Spoken Language Technology*.

Ruhi Sarikaya. 2015. *The technology powering personal digital assistants*. Keynote at Interspeech, Dresden, Germany.

Gokhan Tur and Renato De Mori. 2011. *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Gokhan Tur. 2006. Multitask learning for spoken language understanding. In *In Proceedings of the ICASSP*, Toulouse, France.

Puyang Xu and Ruhi Sarikaya. 2014. Targeted feature dropout for robust slot filling in natural language understanding. In *ISCA - International Speech Communication Association*, September.