Bag of Experts Architectures for Model Reuse in Conversational Language Understanding

Rahul Jha, Alex Marin, Suvamsh Shivaprasad, and Imed Zitouni
Microsoft Corporation, Redmond, WA
{rajh,alemari,sushivap,izitouni}@microsoft.com

Abstract

Slot tagging, the task of detecting entities in input user utterances, is a key component of natural language understanding systems for personal digital assistants. Since each new domain requires a different set of slots, the annotation costs for labeling data for training slot tagging models increases rapidly as the number of domains grow. To tackle this, we describe Bag of Experts (BoE) architectures for model reuse for both LSTM and CRF based models. Extensive experimentation over a dataset of 10 domains drawn from data relevant to our commercial personal digital assistant shows that our BoE models outperform the baseline models with a statistically significant average margin of 5.06% in absolute F1-score when training with 2000 instances per domain, and achieve an even higher improvement of 12.16% when only 25% of the training data is used.

1 Introduction

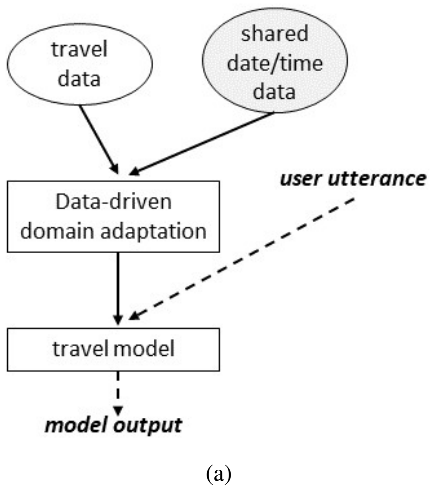
Natural language understanding (NLU) is a key component of dialog systems for commercial personal digital assistants (PDAs) such as Amazon Alexa, Google Home, Microsoft Cortana and Apple Siri. The task of the NLU component is to map input user utterances into a semantic frame consisting of domain, intent and slots (Kurata et al., 2016). The semantic frame is used by the dialog manager for state tracking and action selection.

Slot tagging can be formulated as a sequence classification task where each input word in the user utterance must be classified as belonging to one of the slot types in a predefined schema (Sarikaya et al., 2016). In a standard NLU architecture, each new domain defines a new domain-specific schema for its slots. Figure 1 shows examples of annotated queries from three different domains relevant to a typical commercial digital assistant. Since the schemas for different domains can vary, the usual strategy is to train a separate slot tagging model for each new domain. However, the number of domains increases rapidly as the PDAs are required to support new scenarios and training a separate slot tagging model for each new domain becomes prohibitively expensive in terms of annotation costs.

Travel
What are the $[best]_{rating}$ $[hotels]_{service}$ in $[Austin]_{location}$
I need a room from $[March\ 23rd]_{start\_date}$ to $[April\ 7th]_{end\_date}$
Flight Status
Check flight heading for $[New\ York]_{location}$ on $[October\ 16]_{start\_date}$ at $[3\ pm]_{start\_time}$
What time will $[Lufthansa]_{airline}$ flight $[182]_{flight}$ from $[Denver]_{location}$ land?
Real Estate
See $[houses]_{property\_type}$ $[for\ rent]_{listing\_type}$ in $[Houston]_{location}$
Find out if $[123\ main\ street]_{location}$ is on the market

Figure 1: Example utterances with the output slot tags for three different domains.

Even though different domains have different slot tagging schemas, some classes of slots appear across a number of domains, as suggested by the examples in Figure 1. Both **travel** and **flight status** have *date* and *time* related slots, and all three domains have the *location* slot. Reusing annotated data for these common slots would allow us to train models with better accuracy using less data. However, since both the input distribution and the label distribution are different across domains, we must use domain adaptation methods to train on the joint data (Daume, 2007; Kim et al.,



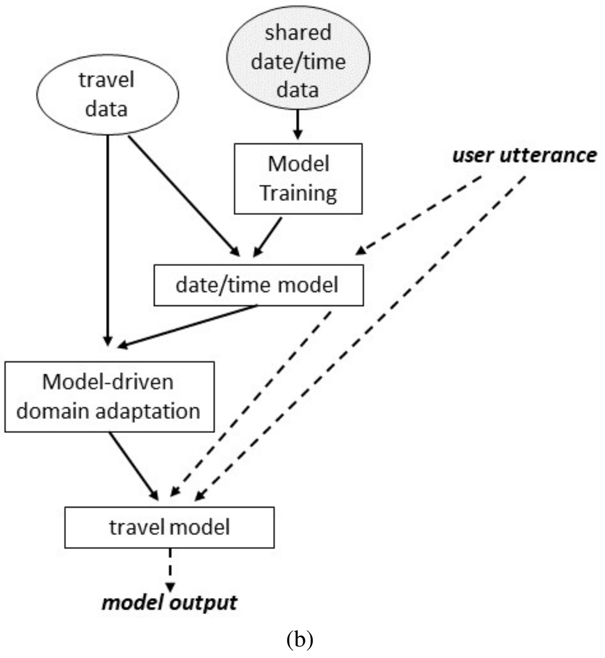


Figure 2: Examples for two different strategies for reusing annotated data from reusable slots. Figure (a) shows data-driven adaptation, while Figure (b) shows model-driven adaptation. Solid lines show the flow at training time, while the dashed lines show the flow at run-time for the deployed travel model. The model output at run-time is a slot-tagged user utterance.

2016c; Blitzer et al., 2006).

In this data-driven adaptation approach, we build a repository of annotated data containing date, time, location and other reusable slots. We then combine relevant data from the reusable repository with the domain specific data during model training. Figure 2(a) shows an example of this architecture where reusable date/time data is used for training travel domain.

A drawback of the data-driven adaptation approach is that as the repository of data for reusable slots grows, the training time for new domains increases. The training data for a new domain might be in the hundreds of samples, while the training data for the reusable slots might contain hundreds of thousands of samples. This increase in training time makes iterative refinement difficult in the initial design of new domains, which is when the ability to deploy new models quickly is crucial.

An alternative strategy is to use model-driven adaptation approaches (Kim et al., 2017b) as shown in Figure 2(b). Here, instead of retraining on the data for the reusable slots, we train "expert" models for these slots, and use the output of these models directly when training new domains. Using model-driven adaptation ensures that model training time is proportional to the data size of new target domains, as opposed to the large data size for reusable slots, allowing for faster training.

In this paper, we present a model-driven adaptation approach for slot tagging called Bag of Experts (BoE). In Section 2, we first describe how this approach can be applied to two popular machine learning methods used for slot tagging: Long Short Term Memory (LSTM) and Conditional Random Fields (CRF) models. We then describe a dataset of 10 target domains and 2 reusable domains that we've collected for use in a commercial digital assistant, in Section 3. Using this data, we conduct experiments comparing the BoE models with their non-expert counterparts, and show that BoE models can lead to significant F1-score improvements. The experimental setup is described in Section 4.1 and the results are discussed in Section 4.3. This is followed by a survey of related work in Section 5 and the conclusion in Section 6.

2 Approaches

We first describe our LSTM and CRF models for slot tagging, followed by their BoE variants: LSTM-BoE and CRF-BoE. Tensorflow (Abadi et al., 2015) was used for implementing the LSTM models, while a custom C++ implementation was

Domain	#Train	#Test	#Dev	#Slots	Example Utterance
Fashion	5273	701	696	8	Show me the $[turtleneck]_{item}$ I wore $[last\ Tuesday]_{date}$
Flight Stat.	9481	553	492	9	Is flight $[283]_{flight\_number}$ at $[Kennedy\ airport]_{location}$ on time $[today]_{start\_date}$?
Deals	25598	1271	2036	5	Find $[mexican]_{category}$ deals in $[seattle]_{location}$
Purchase	5033	397	402	18	Buy the $[shirt]_{item}$ I was looking at $[yesterday]_{date}$
Real Estate	5633	525	498	7	Show me the $[rental]_{property\_type}$ at $[13\ Holt\ Street]_{location}$
Shopping	19725	1106	892	16	Find the $[Nov\ 2017]_{date}$ $[iphone]_{brand\_name}$ model
Soc. Net.	38450	432	441	21	Display $[Mike]_{username}$'s $[tweets]_{media\_type}$ from $[yesterday]_{date}$
Sports	20341	1048	1048	21	Display games $[this\ week]_{date}$ for $[texas\ tech]_{team\_name}$
Transport.	162951	19706	19724	17	$[Driving]_{transport\_type}$ directions to $[union\ station]_{location}$
Travel	49300	2027	1990	27	How much for $[2]_{number\_rooms}$ rooms at the $[Hilton]_{place\_name}$ in $[SF]_{location}$?

Table 1: List of target domains used for our experiments, along with some statistics and example utterances. The test and development data sets are sampled at 10% of the total annotated data. "Flight Stat." stands for "Flight Status", "Soc. Net." stands for "Social Network", and "Transport." stands for "Transportation".

used for the CRF models.

2.1 LSTM

For our LSTM model, we follow a standard bidirectional LSTM architecture (Huang et al., 2015; Ma and Hovy, 2016; Lample et al., 2016). Let $w_1...w_n$ denote the input word sequence. For every input word w_i, let f_i^C and b_i^C be the outputs of the forward and backward character level LSTMs respectively, and let m_i be the word embedding (initialized either randomly or with pre-trained embeddings). The input to the word level LSTMs, g_i, is the concatenation of these three vectors:

$$g_i = [f_i^C; b_i^C; m_i]$$

where both $f_i^C, b_i^C \in R^{25}$ and m_i has the same dimensions as the pre-trained embeddings. The forward and backward word level LSTMs take g_i as input and produce f_i^W and b_i^W, which are then concatenated to produce h_i:

$$h_i = [f_i^W, b_i^W]$$

where $f_i^W, b_i^W \in R^{100}$, making $h_i \in R^{200}$. h_i is then input to a dense feed forward layer with a softmax activation to predict the label probabilities for each word. We train using stochastic gradient descent with Adam (Kingma and Ba, 2015). To avoid overfitting, we also use dropout on top of m_i and h_i layers, with a default dropout keep probability of 0.8. We experiment with some variations of this default LSTM architecture, the results are described in Section 4.2.

2.2 LSTM-BoE

We now describe the LSTM Bag of Experts (LSTM-BoE) architecture. Let $e_1...e_k \in E$ be the set of reusable expert domains. For each expert e_j, we train a separate LSTM with the architecture described in Section 2.1. Let $h_i^{e_j}$ be the bi-directional word LSTM output for expert e_j on word w_i.

When training on a target domain, for each word w_i, we first compute the character level LSTMs f_i^C, b_i^C similarly to Section 2.1. We then compute a BoE representation for this word as:

$$h^E = \sum_{e_i \in E} h_i^{e_j}$$

The input to the word level LSTM for word w_i in the target domain is now a concatenation of the character level LSTM outputs (f_i^C, b_i^C), the word embedding m_i, and h^E:

$$g_i = [f_i^C; b_i^C; m_i; h^E]$$

g_i is then input to the word level LSTM for the target domain to produce h_i in the same way as Section 2.1. This architecture is similar to the one presented in (Kim et al., 2017b), with the exception that in their architecture, h^E is concatenated with the word level LSTM output h_i for the target domain. In our architecture, we add h^E before the word-level LSTM in order to capture long-range dependencies of label prediction for a word on expert predictions for context words.

2.3 CRF

Conditional Random Fields (CRF) are a popular family of models that have been proven to work well in a variety of sequence tagging NLP applications (Lafferty et al., 2001). For our experiments, we use a standard linear-chain CRF architecture with n-gram and context features.

In particular, for each token, we use unigram, bigram and trigram features, along with previous and next unigrams, bigrams, and trigrams for context length of up to 3 words. We also use a skip bigram feature created by concatenating the current unigram and skip-one unigram.

We train our CRF using stochastic gradient descent with L1 regularization to prevent overfitting. The L1 coefficient was set to 0.1 and we use a learning rate of 0.1 with exponential decay for learning rate scheduling (Tsuruoka et al., 2009).

2.4 CRF-BoE

Similar to the LSTM-BoE model, we first train a CRF model c_j for each of the reusable expert domains $e_j \in E$. When training on a target domain, for every query word w_i, a one-hot label vector l_i^j is emitted by each expert CRF model c_j.

The length of the label vector l_i^j is the number of labels in the expert domain, with the value corresponding to the label predicted by c_j for word w_i set to 1, and values for all other labels set to 0. For each word, the label vectors for all the expert CRF models are concatenated and provided as features for the target domain CRF training, along with the n-gram features.

3 Data

3.1 Target Domains

We built a dataset of 10 target domains for experimentation. Table 1 shows the list of domains as well as some statistics and example utterances. We treated these as new domains - that is, we do not have real interaction data with users for these domains. The annotated data is therefore prepared in two steps.

First, utterances are obtained using crowdsourcing, where workers are provided with prompts for different intents of a domain and asked to generate natural language utterances corresponding to those intents. Next, the generated utterances are annotated by a different set of crowd workers, using the slot schema for each domain. Inter-annotator agreement as well as manual inspection are used to ensure data quality in both stages.

Variation	Average Diff	P-val
Embeddings		
Glove (100)	$+1.61 \pm 0.71^*$	0.048
Glove (200)	$+2.01 \pm 0.64^*$	0.012
Glove (300)	$+1.92 \pm 0.94$	0.073
PDA Logs (500)	$+2.60 \pm 0.61^*$	0.002
Output Layer		
CRF	$+0.67 \pm 0.30$	0.054
Dropout (default keep probability 0.8)		
keep prob. = 0.5	$-2.53 \pm 0.62^*$	0.003
keep prob. = 0.6	$-1.60 \pm 0.31^*$	0.001
keep prob. = 0.7	-0.29 ± 0.28	0.330
keep prob. = 0.9	$+0.36 \pm 0.25$	0.176
keep prob. = 1.0	$+0.63 \pm 0.30$	0.065

Table 2: Average absolute F1-score improvement on the dev data for different LSTM variations. $*$ indicates the improvement is statistically significant with p-value < 0.05.

The amount of data collected varies for each domain based on its complexity and business priority. Dataset size statistics for the data used in our experiments are presented in section 4.1. Test and dev data are sampled at 10% of the total annotated data, with stratified sampling used in order to preserve the distribution of the intents.

3.2 Reusable Domains

We experiment with two domains containing reusable slots: timex and location. The timex domain consists of utterances containing the slots $date$, $time$ and $duration$. The location domain consists of utterances containing $location$, $location\_type$ and $place\_name$ slots. Both of these types of slots appear in more than 20 of a set of 40 domains developed for use in our commercial personal assistant, making them ideal candidates for reuse.[1]

[1]Several other candidate reusable domains exist, including: the **name** domain containing the slot $contact\_name$; the **number** domain containing the slots $rating$, $quantity$ and $price$; and the **reference** domain containing the slots $ordinal$ (whose values include "first", "second" or "third") and $order\_ref$ (with values such as "before" or "after"). All of these slots appear in more than 25% of the available domains.

Data for these domains was sampled from the input utterances from our commercial digital assistant. Each reusable domain contains about a million utterances. There is no overlap between utterances in the target domains used for our experiments and utterances in the reusable domains. The data for the reusable domains is sampled from other domains available to the digital assistant, not including our target domains.

Grouping the reusable slots into domains in this way provides additional opportunities for a commercial system: the trained reusable domain models can be used in other related products which need to identify time and location related entities. Models trained on the timex and location data have F1-scores of 96% and 89% respectively on test data from their respective domains.

4 Experiments

4.1 Experimental Setup

We want to verify if BoE models can improve slot tagging performance by using the information from reusable domains. To simulate the low data scenario for the initial model training, we create three training datasets by sampling 2000, 1000 and 500 training examples from every domain. We use stratified sampling to maintain the input distribution of the intents across the three training datasets.

For each training dataset, we train the four models as described in Section 2 and compute the precision, recall and F1-score on the test data. Fixed seeds are used when training all models to make the results reproducible. Table 3 summarizes these results, with only F1-scores reported to save space. We describe these results in Section 4.3.

4.2 LSTM architecture variants

Using the dev data set for the 10 domains, we experimented with using different pretrained embeddings, dropout probabilities and a CRF output layer in our LSTM architecture. The results are summarized in Table 2. For each of the 10 domains, we trained using each variant with 10 different seeds, and computed the mean F1-score for each domain. For comparing two variants, we computed the mean difference in the F1-scores over the 10 domains and its p-value.

We tried word level Glove embeddings of 100, 200 and 300 dimensions as well as 500-dimensional word embeddings trained over the utterances from our commercial PDA logs. Both 100 and 200 dimensional Glove embeddings led to statistically significant improvements, but the word embeddings trained over our logs led to the biggest improvement. We also tried using a CRF output layer (Lample et al., 2016) and different values of dropout keep probability, but none of them gave statistically significant improvements over the default model. Based on this, we used PDA trained 500-dimensional word embeddings for our final experiments on test data.

4.3 Results and Discussion

Table 3(a) shows the F1-scores obtained by the different methods for the training data set of 2000 training instances for each of the 10 domains. LSTM based models in general perform better than the CRF based models. The LSTM models have a statistically significant average improvement of 3.14 absolute F1-score over the CRF models. The better performance of LSTM over CRF can be explained by the LSTM being able to use information over longer contexts to make predictions, while the CRF model is limited to at most the previous and next 3 words.

The results in Table 3(a) also show that both the CRF-BoE and LSTM-BoE outperform the basic CRF and LSTM models. LSTM-BoE has a statistically significant mean improvement of 1.92 points over LSTM. CRF-BoE also shows an average improvement of 2.19 points over the CRF model, but the results are not statistically significant. Looking at results for individual domains, the highest improvement for BoE models are seen for **transportation** and **travel**. This can be explained by these domains having a high frequency of $timex$ and $location$ slots, as shown in Table 4.

The **shopping** model shows a regression for BoE models, and a reason could be the low frequency of expert slots (Table 4). However, low frequency of expert slots does not always mean that BoE methods can't help, as shown by the improvement in the **purchase** domain. Finally, for **sports**, **social network** and **deals** domains, the LSTM-BoE improves over LSTM, while CRF-BoE does not improve over CRF. Our hypothesis is that given the query patterns for these domains, the dense vector output used by LSTM-BoE is able to transfer some information, while the categorical label output used by CRF-BoE is not.

Table 3(b) shows the results with 500 and 1000

Train size	**2000**			
Domain	**CRF**	**LSTM**	**CRF-BoE**	**LSTM-BoE**
Fashion	79.54	82.18	80.87	**83.21**
Purchase	66.24	77.56	70.09	**79.72**
Flight Status	87.60	89.86	89.30	**91.51**
Deals	83.74	85.69	83.59	**87.31**
Travel	66.39	71.02	72.81	**75.52**
Transportation	79.18	80.93	**89.65**	85.95
Sports	75.70	77.82	75.08	**79.43**
Social Network	81.71	81.02	81.65	**83.74**
Shopping	77.16	**81.67**	76.07	80.65
Real Estate	96.16	**97.07**	96.18	97.01
Average improvement		+3.14*	+2.19	+5.06*

(a)

Train size	**500**				**1000**			
Domain	**CRF**	**LSTM**	**CRF-BoE**	**LSTM-BoE**	**CRF**	**LSTM**	**CRF-BoE**	**LSTM-BoE**
Fashion	69.05	75.31	71.52	**76.85**	73.81	79.63	75.49	**80.07**
Purchase	53.12	63.58	54.52	**70.66**	61.04	**69.39**	62.23	64.46
Flight Status	78.03	84.33	82.59	**88.29**	84.14	88.12	86.17	**89.85**
Deals	69.82	78.31	72.66	**81.11**	78.60	81.58	78.01	**82.95**
Travel	47.66	58.71	64.28	**70.00**	57.37	65.77	67.53	**73.91**
Transportation	70.37	75.02	**87.12**	85.59	75.03	76.53	**88.21**	86.68
Sports	55.93	65.71	56.88	**68.94**	66.92	71.78	66.52	**71.96**
Social Network	69.73	78.08	66.59	**79.91**	78.45	**80.31**	75.78	79.27
Shopping	59.26	66.55	57.11	**71.10**	70.01	76.32	69.26	**77.04**
Real Estate	91.04	**93.66**	92.59	93.15	93.76	95.07	94.29	**95.34**
Average improvement		+7.52*	+4.19	+12.16*		+4.54*	+2.44	+6.24*

(b)

Table 3: F1-scores obtained by each of the four models for the 10 domains, with the highest score in each row marked as bold. Table (a) reports the results for 2000 training instances, and Table (b) reports the results for 500 and 1000 training instances. The average improvement is computed over the CRF model, with the ones marked * being statistically significant with p-value < 0.05. The average improvement of LSTM-BoE over LSTM is +1.92*, +1.70 and +4.63* for 2000, 1000, and 500 training instances respectively.

training data instances. Note that the improvements are even higher for the experiments with smaller training data. In particular, LSTM-BoE shows an improvement of 4.63 in absolute F1-score over LSTM when training with 500 instances. Thus, as we reduce the amount of training data in the target domain, the performance improvement from BoE models is even higher.

As an example, in the **purchase** domain, the LSTM-BoE model achieves an F1-score of 70.66% with only 500 training instances, while even with 2000 training instances the CRF model achieves an F1-score of only 66.24%. Thus the LSTM-BoE model achieves better F1-score with only one-fourth the training data. Similarly, for **flight status**, **travel**, and **transportation** domains, the LSTM-BoE model gets better performance with 500 training instances, compared to a CRF model with 2000 training instances. The LSTM-BoE architecture, therefore, allows us to reuse the domain experts to produce better performing models with much lower data annotation costs. As the target domain training data increases, the contribution due to domain experts goes down, but more experimentation is needed to establish the threshold at which it is no longer useful to add experts.

5 Related Work

Early methods for slot-tagging used rule-based approaches (Ward and Issar, 1994). Much of the later work on supervised learning focused on CRFs, for example (Sarikaya et al., 2016), or neural networks (Deoras and Sarikaya, 2013; Yao et al., 2013; Liu et al., 2015; Celikyilmaz and Hakkani-Tur, 2015). Unsupervised (or weakly-supervised) methods also were used for NLU tasks, primarily leveraging search query click logs (Hakkani-Tur et al., 2011a,b, 2013) and knowledge graphs (Tur et al., 2012; Heck and Hakkani-Tur, 2012; Heck et al., 2013); hybrid methods, for example as described in (Kim et al., 2015a; Celikyilmaz et al., 2015; Chen et al., 2016), also exist. Our approach

Domain	%Timex	%Location
Fashion	16.08	0.00
Purchase	3.83	0.00
Flight Status	23.42	33.01
Deals	0.00	21.07
Travel	4.79	32.91
Transportation	2.08	85.87
Sports	23.30	3.29
Social Network	5.63	16.83
Shopping	1.84	0.00
Real Estate	0.00	85.74

Table 4: Percentange of queries with timex and location slots in each of our target domains.

in this paper is a purely supervised one.

Transfer learning is a vast area of research, with too many publications for an exhaustive list. We discuss some of the recent work most relevant to our methods. In (Kim et al., 2015b), the slot labels from across different domains are mapped into a shared space using Canonical Correlation Analysis (CCA) and automatically-induced embeddings over the label space. These label representations allow mapping of label types between different domains, which makes it possible to apply standard data-driven domain adaptation approaches (Daume, 2007). They also introduce a model-driven adaptation technique based on training a hidden unit CRF (HUCRF) on the source domain, which is then used to initialize the training for the target domain. The limitation of this approach is that only one source domain can be used, while multiple experts can be used in the proposed BoE approach.

(Kim et al., 2016a) build a single, universal slot tagging model, and constrain the decoding process to subsets of slots for various domains; this process assumes that a mapping of slot tags in the new domain to the ones in the universal slot model has already been generated. A related work by (Kim et al., 2016b) directly predicts the required schema prior to performing the constrained decoding. These approaches are attractive because only one universal model needs to be trained, but do not work in cases when a new domain contains a mixture of new and existing slots. Our approach allows transfer of partial knowledge in such cases.

(Kim et al., 2016c) uses a neural version of the approach first described in (Daume, 2007), by using existing annotated data in a variety of domains to adapt the slot tag models of new domains where the tag space is partly shared. The drawback of such data-driven domain adaptation is the increase in training time as more experts are added.

An expert-based adaptation, similar to the techniques applied in this paper, was first described in (Kim et al., 2017b). (Jaech et al., 2016) use multitask learning, training a bidirectional LSTM with character-level embeddings, trained jointly to produce slot tags for a number of travel-related domains. Finally, (Kim et al., 2017a) frame the problem of temporal shift in data of a single domain (and the related problem of bootstrapping a new domain with imperfectly-matched synthetic data) as one of domain adaptation, applying adversarial training approaches.

A number of researchers also investigated bootstrapping NLU systems using zero-shot learning. (Dauphin et al., 2014; Kumar et al., 2017) both investigated domain classification; most relevant to us is the work by (Bapna et al., 2017), who studied full semantic frame tagging using zero-shot learning, by projecting the tags into a shared embedding space, similar to work done by (Kim et al., 2015b).

6 Conclusion

We experimented with Bag of Experts (BoE) architectures for CRF and LSTM based slot tagging models. Our experimental results over a set of 10 domains show that BoE architectures are able to use the information from reusable expert models to perform significantly better than their non-expert counterparts. In particular, the LSTM-BoE model shows a statistically significant improvement of 1.92% over the LSTM model on average when training with 2000 instances. When training with 500 instances, the improvement of LSTM-BoE model over LSTM is even higher at 4.63%. For multiple domains, an LSTM-BoE model trained on only 500 instances is able to outperform a baseline CRF model trained over 4 times the data. Thus, the BoE approach produces high performing models for slot tagging at much lower annotation costs.

Acknowledgments

We would like to thank Ahmed El Kholy for his comments and feedback on an earlier version of this paper. Also, thanks to Kyle Williams and Zhaleh Feizollahi for their help with code and data collection.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Ankur Bapna, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2017. Toward zero-shot frame semantic parsing for domain scaling. In *Proc. Interspeech*.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 120–128, Stroudsburg, PA, USA. Association for Computational Linguistics.

Asli Celikyilmaz and Dilek Hakkani-Tur. 2015. Convolutional neural network based semantic tagging with entity embeddings. In *Proc. NIPS Workshop on Machine Learning for SLU and Interaction*.

Asli Celikyilmaz, Dilek Hakkani-Tur, Panupong Pasupat, and Ruhi Sarikaya. 2015. Enriching word embeddings using knowledge graph for semantic tagging in conversational dialog systems. In *In Proc. AAAI*.

Yun-Nung (Vivian) Chen, Dilek Hakkani-Tur, Gokhan Tur, Asli Celikyilmaz, Jianfeng Gao, and Li Deng. 2016. Syntax or semantics? knowledge-guided joint semantic frame parsing. In *IEEE Workshop on Spoken Language Technology*.

Hal Daume, III. 2007. Frustratingly Easy Domain Adaptation. In *Proc. ACL*, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

Yann Dauphin, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2014. Zero-shot learning and clustering for semantic utterance classification. In *International Conference on Learning Representations*.

Anoop Deoras and Ruhi Sarikaya. 2013. Deep belief network based semantic taggers for spoken language understanding. In *Proc. Interspeech*.

D. Hakkani-Tur, G. Tur, L. Heck, A. Celikyilmaz, A. Fidler, D. Hillard, R. Iyer, and S. Parthasarathy. 2011a. Employing web search query click logs for multi-domain spoken language understanding. In *Proc. ASRU*.

D. Hakkani-Tur, G. Tur, L. Heck, and E. Shriberg. 2011b. Bootstrapping domain detection using query click logs for new domains. In *Proc. Interspeech*.

Dilek Hakkani-Tur, Asli Celikyilmaz, Larry Heck, and Gokhan Tur. 2013. A weakly supervised approach for discovering new user intents from search query logs. In *Proc. Interspeech*.

Larry Heck and Dilek Hakkani-Tur. 2012. Exploiting the semantic web for unsupervised spoken language understanding. In *Proc. SLT*.

Larry Heck, Dilek Hakkani-Tur, and Gokhan Tur. 2013. Leveraging knowledge graphs for web-scale unsupervised semantic parsing. In *Proc. Interspeech*.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Aaron Jaech, Larry Heck, and Mari Ostendorf. 2016. Domain adaptation of recurrent neural networks for natural language understanding. In *Proc. Interspeech*.

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 *Proc. NAACL*.

Young-Bum Kim, Alexandre Rochette, and Ruhi Sarikaya. 2016a. Natural language model reusability for scaling to different domains. In *Proc. EMNLP*.

Young-Bum Kim, Karl Stratos, and Dongchan Kim. 2017a. Adversarial adaptation of synthetic or stale data. In *Proc. ACL*.

Young-Bum Kim, Karl Stratos, and Dongchan Kim. 2017b. Domain attention with an ensemble of experts. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 643–653.

Young-Bum Kim, Karl Stratos, and Ruhi Sarikaya. 2016b. Domainless adaptation by constrained decoding on a schema lattice. In *Proc. COLING*.

Young-Bum Kim, Karl Stratos, and Ruhi Sarikaya. 2016c. Frustratingly easy neural domain adaptation. In *COLING*, pages 387–396.

Young-Bum Kim, Karl Stratos, Ruhi Sarikaya, and Minwoo Jeong. 2015b. New transfer learning techniques for disaparate label sets. In *Proc. ACL*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *ICLR*.

A. Kumar, P. R. Muddireddy, M. Dreyer, and B. Hoffmeister. 2017. Zero-shot learning across heterogeneous overlapping domains. In *Proc. Interspeech*.

Gakuto Kurata, Bing Xiang, Bowen Zhou, and Mo Yu. 2016. Leveraging sentence-level information with encoder lstm for semantic slot filling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2077–2083, Austin, Texas. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *NAACL-HLT*.

C. Liu, P. Xu, and R. Sarikaya. 2015. Deep contextual language understanding in spoken dialog systems. In *Proc. Interspeech*.

Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, abs/1603.01354.

Ruhi Sarikaya, Paul A. Crook, Alex Marin, Minwoo Jeong, Jean-Philippe Robichaud, Asli Celikyilmaz, Young-Bum Kim, Alexandre Rochette, Omar Z. Khan, Derek (Xiaohu) Liu, Daniel Boies, Tasos Anastasakos, Zhaleh Feizollahi, Nikhil Ramesh, Hisami Suzuki, Roman Holenstein, Elizabeth Krawczyk, and Vasiliy Radostev. 2016. An overview of end-to-end language understanding and dialog management for personal digital assistants. IEEE.

Yoshimasa Tsuruoka, Jun'ichi Tsujii, and Sophia Ananiadou. 2009. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pages 477–485, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gokhan Tur, Minwoo Jeong, Ye-Yi Wang, Dilek Hakkani-Tur, and Larry Heck. 2012. Exploiting the semantic web for unsupervised natural language semantic parsing. In *Proc. Interspeech*.

W. Ward and S. Issar. 1994. Recent improvements in the CMU spoken language understanding system. In *Proc. ACL*.

K. Yao, J. Zweig, M. Hwang, Y. Shi, and D. Yu. 2013. Recurrent neural networks for language understanding. In *Proc. Interspeech*.