Zero-Shot Adaptive Transfer for Conversational Language Understanding

Abstract

Conversational agents such as Alexa and Google Assistant constantly need to increase their language understanding capabilities by adding new domains. A massive amount of labeled data is required for training each new domain. While domain adaptation approaches alleviate the annotation cost, prior approaches suffer from increased training time and suboptimal concept alignments. To tackle this, we introduce a novel Zero-Shot Adaptive Transfer method for slot tagging that utilizes the slot description for transferring reusable concepts across domains, and enjoys efficient training without any explicit concept alignments. Extensive experimentation over a dataset of 10 domains relevant to our commercial personal digital assistant shows that our model outperforms previous state-of-the-art systems by a large margin, and achieves an even higher improvement in the low data regime.

Introduction

Recently, there is a surge of excitement in adding numerous new domains to conversational agents such as Alexa, Google Assistant, Cortana and Siri to support a myriad of use cases. However, building a slot tagger, which is a key component for natural language understanding (NLU) BIBREF0 , for a new domain requires massive amounts of labeled data, hindering rapid development of new skills. To address the data-intensiveness problem, domain adaptation approaches have been successfully applied. Previous approaches are roughly categorized into two groups: data-driven approaches BIBREF1 , BIBREF2 and model-driven approaches BIBREF3 , BIBREF4 .

In the data-driven approach, new target models are trained by combining target domain data with relevant data from a repository of arbitrary labeled datasets using domain adaptation approaches such as feature augmentation BIBREF1 . A disadvantage of this approach is the increase in training time as the amount of reusable data grows. The reusable data might contain hundreds of thousands of samples, making iterative refinement prohibitive. In contrast, the model-driven approach utilizes “expert" models for summarizing the data for reusable slots BIBREF3 , BIBREF4 . The outputs of the expert models are directly used when training new domains, allowing for faster training. A drawback of this approach is that it requires explicit concept alignments which itself is not a trivial task, potentially missing lots of reusable concepts. Additionally, it's not easy to generalize these models to new, unseen slots.

In this paper, we present a new domain adaptation technique for slot tagging inspired by recent advances in zero-shot learning. Traditionally, slot tagging is formulated as a sequence labeling task using the BIO representation (Figure 1). Our approach formulates this problem as detecting spans that contain values for each slot as shown in Figure 1 . For implicit transfer of reusable concepts across domains, we represent slots in a shared latent semantic space by embedding the slot description. With the shared latent space, domain adaptation can simply be done by fine-tuning a base model, which is trained on massive data, with a handful of target domain data without any explicit concept alignments. A similar idea of utilizing zero-shot learning for slot tagging has been proven to work in semi-supervised settings BIBREF5 . Our zero-shot model architecture differs from this by adding: 1) an attention layer to produce the slot-aware representations of input words, 2) a CRF layer to better satisfy global consistency constraints, 3) character-level embeddings to incorporate morphological information. Despite its simplicity, we show that our model outperforms all existing methods including the previous zero-shot learning approach in domain adaptation settings.

We first describe our approach called Zero-Shot Adaptive Transfer model (ZAT) in detail. We then describe the dataset we used for our experiments. Using this data, we conduct experiments comparing our ZAT model with a set of state-of-the-art models: Bag-of-Expert (BoE) models and their non-expert counterparts BIBREF4 , and the Concept Tagger model BIBREF5 , showing that our model can lead to significant F1-score improvements. This is followed by an in-depth analysis of the results. We then provide a survey of related work and concluding remarks.

Adaptive Transfer

Our Zero-Shot Adaptive Transfer model for slot tagging is a hierarchical model with six layers (Figure 2).

Data

For our experiments, we collected data from a set of ten diverse domains. Table 1 shows the domains along with some statistics and sample utterances. Since these are new domains for our digital assistant, we did not have enough data for these domains in our historical logs. Therefore, the data was collected using crowdsourcing from human judges. For each domain, several prompts were created to crowdsource utterances for a variety of intents. These utterances were then annotated through our standard data annotation pipeline after several iterations of measuring interannotator agreement and refining the annotation guidelines. We collected at least 5000 instances for each domain, with more data collected for some domains based on business priority.

For each of the domains, we sampled 80% of the data as training and 10% each as dev and test sets. Further samples of 2000, 1000, and 500 training samples were taken to compare our approach with previous methods. All samples were obtained by stratified sampling based on the annotated intents of the utterances.

Baseline Systems

In order to compare our method against the state-of-the-art models, we compare against the models presented in BIBREF4 , including the BoE models and their non-BoE variants. We also compare our method with another zero-shot model for slot tagging BIBREF5 in domain adaptation settings.

Following BIBREF4 , we concatenate the output of 25 dimensional character-level bidirectional LSTMs with pre-trained word embeddings to obtain morphology-sensitive embeddings. We then use a 100 dimensional word-level bidirectional LSTM layer to obtain contextualized word representations. Finally, the output of this layer is passed on 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 BIBREF11 . To avoid overfitting, we also apply dropout to the output of each layer, with a default dropout keep probability of 0.8.

The LSTM-BoE architecture is similar to the LSTM model with the exception that we use the output vectors of the word-level bidirectional LSTM layer of each expert model to obtain enriched word embeddings. Specifically, let $e_1 ... e_k \in E$ be the set of reusable expert domains. For each expert e_j , we train a separate LSTM model. Let $h^{e_j}_i$ be the word-level bi-directional LSTM output for expert e_j on word w_i . When training on a target domain, for each word w_i , we first compute a BoE representation for this word as $h^E = \sum _{e_i \in E} h^{e_j}_i$. 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, the pre-trained word embedding, and the BoE representation.

Following BIBREF4 , We use two expert 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. 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. Models trained on the timex and location data have F1-scores of 96% and 89% respectively on test data from their respective domains.

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 BIBREF12 .

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^j_i is emitted by each expert CRF model c_j . The length of the label vector l^j_i 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.

For comparison with a state-of-the-art zero-shot model, we implement the concept tagger (CT) BIBREF5 . The CT model consists of a single 256 dimensional bidirectional LSTM layer that takes pre-trained word embeddings as input to produce contextual word representations. This is followed by a feed forward layer where the contextual word representations are combined with a slot encoding to produce vectors of 128 dimensions. The slot encoding is the average vector of the word embeddings for the slot description. This feeds into another 128 dimensional bi-directional LSTM layer followed by a softmax layer that outputs the prediction for that slot.

Domain Adaptation using Zero-Shot Model

For domain adaptation with zero-shot models, we first construct a joint training dataset by combining the training datasets of size 2000 from all domains except for a target domain. We then train a base model on the joint dataset. We sample input examples during training and evaluation for each slot to include both positive examples (which have the slot) and negative examples (which do not have the slot) with a ratio of 1 to 3. After the base model is trained, domain adaptation is simply done by further training the base model on varying amounts of the training data of the target domain. Note that the size of the joint dataset for each target domain is 18,000, which is dramatically smaller than millions of examples used for training expert models in the BoE approach. Furthermore, there are a lot of utterances in the joint dataset where no slots from the target domain is present.

Comparative Results

Table 2 shows the F1-scores obtained by the different methods for each of the 10 domains. LSTM based models in general perform better than the CRF based models. Both the CRF-BoE and LSTM-BoE outperform the basic CRF and LSTM models. Both zero-shot models, CT and ZAT, again surpass the BoE models. ZAT has a statistically significant mean improvement of \$4.04\$, \$5.37\$ and \$3.27\$ points over LSTM-BoE with training size 500, 1000 and 2000, respectively. ZAT also shows a statistically significant average improvement of \$2.58\$, \$2.44\$ and \$2.5\$ points over CT, another zero-shot model with training size 500, 1000 and 2000, respectively. 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. But BoE models show a regression in the shopping domain, and a reason could be the low frequency of expert slots. In contrast, ZAT consistently outperforms non-adapted models (CRF and LSTM) by a large margin. This is because ZAT can benefit from other reusable slots than $timex$ and $location$. Though not as popular as $5.37$0 and $5.37$1 , slots such as $5.37$2 , $5.37$3 , $5.37$4 , and $5.37$5 appear across many domains.

We plot the averaged performances on varying amounts of training data for each target domain in Figure 3 . Note that the improvements are even higher for the experiments with smaller training data. In particular, ZAT shows an improvement of 14.67 in absolute F1-score over CRF when training with 500 instances. ZAT achieves an F1-score of 76.04% with only 500 training instances, while even with 2000 training instances the CRF model achieves an F1-score of only 75%. Thus the ZAT model achieves better F1-score with only one-fourth the training data.

Table 3 shows the performances of CT and ZAT when no target domain data is available. Both models are able to achieve reasonable zero-shot performance for most domains, and ZAT shows an average improvement of 5.07 over CT.

Model Variants

In Table 4 , we ablate our full model by removing the CRF layer ($-CRF$) and character-level word embeddings ($-CHAR$). Without CRF, the model suffers a loss of 1%-1.8% points. The character-level word embeddings are also important: without this, the performance is down by 0.5%-2.7%. We study the impact of fine-tuning the pre-trained word embeddings ($+WEFT$). When there is no target domain data available, fine-tuning hurts performance. But, with a moderate amount of target domain data, fine-tuning improves performance.

Analysis

To better understand our model, in Figure 7 , we visualize the attention weights for the input sentence "Can I wear jeans to a casual dinner?" with different slots: (a) category, (b) item, and (c) time. From (a) and (b), it is clear that the attention is concentrated on the relevant words of the input and slot description. In contrast, there is no salient attention when the slot is not present in the input sentence.

To analyze the impact of context, we compute the error rate with respect to span start position in the input sentence. Figure 4 shows that error rate tends to degrade for span start positions further from the beginning. This highlights opportunities to reduce a significant amount of errors by considering previous context.

As shown in Figure 5 , our model makes more errors for longer spans. This can be improved by consulting spans detected by parsers or other span-based models such as coreference resolution systems BIBREF13 .

Finally, we compute the percentage of POS tags that are tied to labeling errors. Figure 6 shows POS tags which occurs more than 10,000 times and contributes to more than 10% of errors. It is not surprising that there are many errors for ADJ, ADV and NOUN. Our system suffers in handling conjunctive structures, for instance “Help me find my $[black\text{ }and\text{ }tan]_{described\_as}$ $[jacket]_{item}$ ”, and parsing information can be helpful at enforcing structural consistencies. The NUM category is associated with a variety of concepts and diverse surface forms. Thus it is a probably good idea to have an expert model focusing on the NUM category.

Related Work

A number of deep learning approaches have been applied to the problem of language understanding in recent years BIBREF14 , BIBREF15 , BIBREF16 . For a thorough overview of deep learning methods in conversational language understanding, we refer the readers to BIBREF17 .

As the digital assistants increase in sophistication, an increasing number of slot models have to be trained, making scalability of these models a concern. Researchers have explored several directions for data efficient training of new models. One of the directions has been multi-task learning, where a joint model across multiple tasks and domains might be learned BIBREF18 , BIBREF19 , BIBREF20 . As a recent example, BIBREF21 presented an approach for multi-task learning across the tasks of language understanding and dialog state tracking. BIBREF22 presented a multi-task learning approach for language understanding that consists of training a shared representation over multiple domains, with additional fine-tuning applied for new target domains by replacing the affine transform and softmax layers.

Another direction has been domain adaptation and transfer learning methods. Early focus was on data driven adaptation techniques where data from multiple source domains was combined BIBREF1 . Such data-driven approaches offer model improvements at the cost of increased training time. More recently, model-driven approaches have shown success BIBREF3 , BIBREF4 . These approaches follow the strategy of first training expert models on the source data, and then using the output of these models when training new target models. A benefit of these approaches over data-driven adaptation techniques is the improved training time that scales well as the number of source domains increase.

However, both these transfer learning approaches require concept alignment to map the new labels to existing ones, and cannot generalize to unseen labels. This has led researchers to investigate zero-shot learning techniques, where a model is learned against label representations as opposed to a fixed set of labels.

Several researchers have explored zero-shot models for domain and intent classification. BIBREF23 described a zero-shot model for domain classification of input utterances by using query click logs to learn domain label representations. BIBREF24 also learn a zero-shot model for domain classification. BIBREF25 learn a zero-shot model for intent classification using a DSSM style model for learning semantic representations for intents.

Slot tagging using zero-shot models has also been explored. BIBREF26 presented a zero-shot approach for slot tagging based on a knowledge base and word representations learned from unlabeled data. BIBREF5 also applied zero-shot learning to slot-filling by implicitly linking slot representations across domains by using the label descriptions of the slots. Our method is similar to their approach, but we use an additional attention layer to produce the slot-aware representations of input words, leading to better performance as demonstrated by our empirical results.

More recently, zero-shot learning has also been applied to other tasks. For example, BIBREF27 apply zero-shot learning for training language understanding models for multiple languages and show good results. BIBREF28 presented a zero-shot model for question generation from knowledge graphs, and BIBREF29 presented a model for zero-shot transfer learning for event extraction.

Conclusion

In this paper, we introduce a novel Zero-Shot Adaptive Transfer method for slot tagging that utilizes the slot description for transferring reusable concepts across domains to avoid some drawbacks of prior approaches such as increased training time and suboptimal concept alignments. Experiment results show that our model performs significantly better than state-of-the-art systems by a large margin of 7.24% in absolute F1-score when training with 2000 instances per domain, and achieves an even higher improvement of 14.57% when only 500 training instances are used. We provide extensive analysis of the results to shed light on future work. We plan to extend our model to consider more context and utilize exogenous resources like parsing information.