Open Intent Extraction from Natural Language Interactions

Nikhita Vedula
Ohio State University
vedula.5@osu.edu

Nedim Lipka
Adobe Research
lipka@adobe.com

Pranav Maneriker
Ohio State University
maneriker.1@osu.edu

Srinivasan Parthasarathy
Ohio State University
srini@cse.ohio-state.edu

ABSTRACT

Accurately discovering user intents from their written or spoken language plays a critical role in natural language understanding and automated dialog response. Most existing research models this as a classification task with a single intent label per utterance, grouping user utterances into a single intent type from a set of categories known beforehand. Going beyond this formulation, we define and investigate a new problem of *open intent* discovery. It involves discovering one or more generic intent types from text utterances, that may not have been encountered during training. We propose a novel domain-agnostic approach, *OPINE*, which formulates the problem as a sequence tagging task under an open-world setting. It employs a CRF on top of a bidirectional LSTM to extract intents in a consistent format, subject to constraints among intent tag labels. We apply a multi-head self-attention mechanism to effectively learn dependencies between distant words. We further use adversarial training to improve performance and robustly adapt our model across varying domains. Finally, we curate and plan to release an open intent annotated dataset of 25K real-life utterances spanning diverse domains. Extensive experiments show that our approach outperforms state-of-the-art baselines by 5-15% F1 score points. We also demonstrate the efficacy of OPINE in recognizing multiple, diverse domain intents with limited (can also be zero) training examples per unique domain.

ACM Reference Format:
Nikhita Vedula, Nedim Lipka, Pranav Maneriker, and Srinivasan Parthasarathy. 2020. Open Intent Extraction from Natural Language Interactions. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3366423.3380268

1 INTRODUCTION

Recent advances in natural language understanding (NLU) and speech recognition technologies have triggered the advent of a wealth of conversational agents such as Apple's Siri, Microsoft's Cortana and Amazon's Alexa. To effectively interact with people and answer their diverse questions, such agents need to parse and interpret human language utterances, especially people's intentions or *intents*, and respond accordingly. Progress in the field of deep learning has led to the emergence of numerous user intent detection models [3, 9, 10, 17, 20, 30, 32, 39, 67, 70, 71, 74]. Most existing research including commercial NLU engines detect user intents via multi-class classification, by categorizing input utterances into pre-defined intent classes for which sufficient labeled data is available during model training. Such works cannot address new or previously unseen intent categories, i.e., they work with a *closed world* assumption [9, 10, 17, 30, 39]. They further assume that an input text expresses only a single intent [10, 17, 30, 39, 70]. This is unlike real-world scenarios where users can express multiple, distinct intentions in a single utterance.

In this work, we propose a framework called OPINE (OPen INtent Extraction) that automatically discovers user intents in natural language, *without* prior knowledge of a comprehensive list of intent classes that the text may contain. In other words, OPINE is not restricted to a pre-defined set of intent categories. It can recognize instances of novel or newly emerging intent types that it has never seen before. Tackling such an *open world* case is much more challenging than the closed-world classification setting predominantly found in literature. We name this novel task of identifying and extracting explicit user intentions from text utterances, without any information about the potential intent schema, as *Open Intent Discovery.* Recognizing open intents from users' text or speech inputs has several downstream applications, especially when it is unfeasible, expensive and restrictive to enumerate or possess prior knowledge about all possible intents during model development and training. Open intent discovery can help summarize the common or frequent user objectives and functions associated with a business or a product. It can highlight and help prioritize common bugs and issues reported to customer support or public forums, and spot action items in emails or meeting transcripts. It can also aid the discovery of novel or newly emerging characteristics, skills or functionalities. To illustrate, the text *"Please make a 10:30 sharp appointment for a haircut"* contains a single user intent of making a haircut appointment; whereas the text *"I would like to reserve a seat and also if possible, request a special meal on my flight"* contains multiple intents – a seat reservation and a meal request. Contrarily, the sentence *"Anyone knows the battery life of iPhone?"* merely requests information on a particular topic and does not contain an explicit intent action, such as buying an iPhone. We do not consider such ambiguous or questionable utterances in this work.

Recent efforts [29, 37, 70] have a similar objective as ours, i.e., recognizing intents outside of the labeled training data. Xia et al [70] treat this as a zero-shot classification problem, under the assumption that a list of new or unseen (during training) intent classes is available at test time along with some prior knowledge about them, and classifies the input text into one of these classes.

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380268

Other techniques [29, 37] do not have this limitation. But they can only identify if an input utterance is likely to contain a new intent or domain. They do not 'discover' or specify what the new intents are. Further, all above mentioned approaches can only handle the basic case of an input utterance containing a *single* intent. Our work does not have any of the above restrictions, and to the best of our knowledge, is the *first* work to address the aforementioned limitations. It gives a fine-grained picture of the diverse intents in user utterances, rather than merely recognizing the presence of novel intents or classifying new intents into high level categories.

Unlike the prior literature, our proposed approach, OPINE, models the open intent discovery problem as a *sequence tagging* task (Section 2). We develop a neural model consisting of a Conditional Random Field (CRF) on top of a bidirectional LSTM with a multi-head self-attention mechanism. A crucial challenge associated with developing a generic technique for open intent discovery is ensuring its effectiveness across several task domains or fields. For this purpose, OPINE represents all kinds of user intents extracted from the textual input in a consistent and generalizable format, independent of their domain. We further employ adversarial training at the lower layers of our model, and unsupervised pre-training in the target domain under consideration. These strategies empower our model for cross-domain adaptation even in the absence of sufficient labeled training data, as we show empirically in Section 4.

Commonly used labeled datasets in the intent detection literature such as SNIPS [10] or ATIS [11, 24] largely have concise, coherent and single-sentence texts. They are not very representative of complex, real-world dialog scenarios (e.g. customer support conversations) which could be verbose and ungrammatical, with intents scattered throughout their content. Thus, we develop and plan to make available a large dataset with 25K real-world utterances from the online Stack Exchange[1] forum. They span several genres and have been annotated with intents by crowd workers. To summarize, the key contributions of our work are:

- We formulate and solve a novel problem of *open intent discovery* in text. Our proposed technique OPINE is flexible, generalizable, and agnostic of the domain of the target text.
- OPINE can discover both previously seen as well as *unseen* (during training) user intents in diverse real-world scenarios. It can identify *multiple* user intents per utterance, without any restriction on the number or types of intents possible.
- We curate and present a large intent-annotated dataset of 25K text instances from various real-world task domains.

2 PROBLEM FORMULATION

This work introduces and addresses the novel problem of *Open Intent Discovery* in asynchronous text conversations. The objective of this problem is to identify all possible *actionable intents* from text utterances. These may be underlying goals, activities or tasks that a user wants to perform or have performed. We define an *intent* as consisting of two parts: (i) an *action*, which is a word or phrase representing a tangible purpose, task or activity which is to be requested or performed, and (ii) an *object*, which represents those entity words or phrases that the action is going to act or operate upon. A similar definition has been used to define intention posts in social media and discussion forums [9, 67]. For instance, the intent of the text *"Please make a 10:30 sharp appointment for a haircut"* is to make or schedule a haircut appointment. It consists of an action *"make"* and an object *"appointment"*, *"appointment for haircut"*, or *"haircut appointment"*. Similarly, for the utterance *"I would like to reserve a seat and request a special meal on my flight"*, the actions are *"reserve"* and *"request"* and the objects are *seat* and *special meal*, for the respective intents of seat booking and meal request.

We concede that there may be user utterances that indicate an intent by implying an object, without explicitly mentioning it. An example of this case is the statement *"I want to arrive by 8:30"*. We clarify that such utterances are outside the scope of this work. We focus primarily on actionable intents that explicitly contain the presence of one or more action phrases as well as object phrases, since we believe that both of these are essential to holistically and unambiguously understand a user's intent. We choose such a definition for user intents to address commonly available user interactions within help or customer support forums and with smart speaker devices (e.g. Amazon Alexa, Apple Siri), which often contain user requests for assistance on a particular task.

Following our two-part definition of an intent, we formulate the open intent discovery problem as a sequence tagging task over three tags: Action, Object, and None (the remaining words that are neither an Action nor an Object). A user intent then consists of a matching pair of an Action phrase and an Object phrase. As previous illustrations show, the Action component of an intent is likely to consist of a verb or infinitive phrase that follows a noun or a subject. The Object component often comprises of a noun or compound noun (i.e., an expression with multiple nouns) phrase, possibly qualified by adverbs or adjectives. However, we cannot simply use a part-of-speech (POS) tagger, or a semantic parser to identify Action-Object tags due to the following reasons. First, a POS tagger or a parser cannot distinguish between the Action-Object pairs associated with intents, and those that are merely part of the descriptive text. They will hence suffer from a low precision problem (Table 2). Second, corresponding Action and Object phrases may be spatially distant from each other in the input text and may even span multiple sentences (Table 5). Having said that, we do notice the efficacy of initially pre-training the model weights of OPINE with the verb-object tags obtained from a dependency parser (Table 3). It helps our model learn generic indicators for various kinds of intents independent of the input domain, especially if there is insufficient annotated training data. We then fine-tune our model with labeled data specific to our problem.

An extension to the intent discovery task involves *slot filling*, that identifies entities semantically relevant to the identified intents to fill embedded 'slots' in a semantic frame. The frame corresponds to a specific task or goal. Actions can be programmed based on each predefined semantic frame. In this work, we only focus on identifying intents from text utterances. Further analysis of frequently occurring intents or relationships between the discovered intents can be used to speed up curation and creation of novel frames for further application-specific downstream analysis. We provide examples of such analysis in Sections 4.3 and 5).

The task that we describe in this work, and our approach to solve it is also different from the Open Information Extraction (OpenIE) tasks (e.g. [2]) and Semantic Role Labeling (SRL) tasks (e.g. [61])

[1] www.stackexchange.com

in the following ways: (i) OpenIE is used to extract relation triples from text, with the constituents occurring in the input sentence, whereas we define intent in the form of Action-Object pairs. (ii) SRL aims to label and relate constituents in input sentences with their semantic meanings. Not all such constituents pertain to expressed user intent; we focus on intent relations only. (iii) Typical OpenIE and SRL tasks use individual sentences as inputs in their frameworks. Our approach does not have such a restriction, and can distinguish sentences that contain extraneous information and do not express users' intent. Therefore, the algorithms proposed for the OpenIE or SRL tasks are not directly applicable to the Open Intent Discovery task. However for the purposes of evaluation, we compare OPINE with an SRL baseline in Table 2.

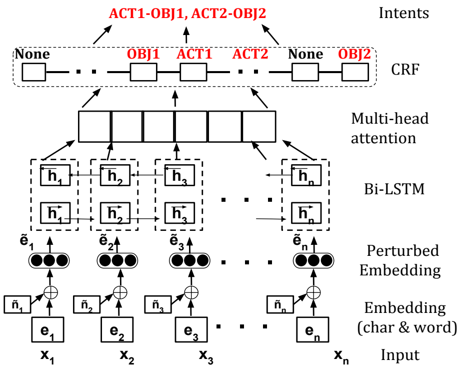


Figure 1: Our OPINE open intent extraction model

3 THE OPINE FRAMEWORK

Figure 1 displays the architecture of OPINE. Given an input text x consisting of a sequence of words $[x_1, x_2, ..., x_n]$, we first transform it into a feature sequence by constructing the character level representation of each word x_i. This is because incorporating character level representations of words can boost the effectiveness of sentence representations by capturing morphological information present in the language [42, 75]. For this purpose, we build a CNN consisting of convolutional and max pooling layers with dropout [56], similar to [26]. We also obtain pre-trained word level embeddings for each token. Such low-dimensional and dense embeddings are highly effective in capturing both syntactic and semantic information. Character-level information can often be overshadowed by word-level embeddings if both are simply concatenated to produce a combined representation for each word. We thus adopt a *highway network* [57] to combine both the character level and word level embeddings in a balanced manner and retain the impact of the both kinds of embeddings. Let e_i^{cw} be the concatenation of the character and word level representations e_i^c and e_i^w of word x_i. The combined embedding e is given by:

$$e = r \odot \tanh(W_H e^{cw} + b_H) + (1 - r) \odot e^{cw}$$
$$r = \sigma(W_R e^{cw} + b_R)$$

where tanh is the hyperbolic tangent function, $\odot$ denotes element-wise multiplication, W_R and W_H are weight matrices, and b_R and b_H are bias vectors. r (transform gate) and $1 - r$ (carry gate) are non-linear transformations indicating the proportion of output produced by transforming the input, and carrying it. Every word x_i is thus transformed into an embedding e_i, which is input to the next layer, namely a bidirectional LSTM layer [19, 25]. This layer generates a sequence of word-level representations $[h_1, h_2, ..., h_n]$ from forward $(\overrightarrow{h_t})$ and backward $(\overleftarrow{h_t})$ sequence contexts, based on the recurrences of an LSTM cell [25].

Adversarial Training: We employ adversarial training to regularize our model [18, 46], improve its robustness to slight input perturbations, and discover features and structures common across multiple domains [14, 33, 40]. We generate *adversarial* input examples that are very close to the original inputs and should yield the same labels, yet are likely to be mispredicted by the current model. These examples are created by adding small worst case perturbations or noise to the inputs in the direction that significantly increases the model's loss function. OPINE is then trained on the mix of original and adversarial examples to improve its stability to input perturbations. Since adversarial training considers continuous perturbations to inputs, we add adversarial noise at the embedding layer [46]. Let input text $x = [x_1, \ldots, x_n]$ be represented by embedding e. We generate its worst case perturbation η of a small bounded norm ϵ, which is a tunable hyperparameter. It maximizes the loss function $\mathcal{L}$ of the current model with parameters θ as:

$$\widetilde{\eta} = \underset{||\eta|| \le \epsilon}{\arg\max}\, \mathcal{L}(e + \eta; \theta)$$

Since the exact computation of $\widetilde{\eta}$ is intractable in complex neural networks, we use the first order approximation via the fast gradient method [18] to obtain an approximate worst case perturbation of norm ϵ. We also normalize the word and character embeddings, so that the model does not trivially learn the embeddings of large norms and make the perturbations insignificant [46].

$$\widetilde{\eta} = \epsilon \frac{g}{||g||}; \text{ where } g = \nabla_e(\mathcal{L}(e; \theta))$$
$$\widetilde{e} = e + \widetilde{\eta}$$
$$\mathcal{L}' = \alpha \mathcal{L}(e; \theta') + (1 - \alpha)\mathcal{L}(\widetilde{e}; \theta')$$

where $\widetilde{e}$ represents the perturbed embedding of an adversarial example generated from embedding e and ∇_e denotes the gradient operator. $\mathcal{L}(e; \theta')$ and $\mathcal{L}(\widetilde{e}; \theta')$ represent the loss functions from the original training instance and its adversarial transformation respectively. α is a weighting parameter. The new loss function $\mathcal{L}'$ can be optimized in the same way as the original loss $\mathcal{L}$. While generating adversarial examples, we measure the semantic (cosine) similarity between the original and adversarial embeddings, and only choose those adversarial examples where the similarity is greater than a threshold. Adversarial training ensures that the meaning of a sentence cannot be inverted via a small change. So, words with similar grammatical role but different meanings are still separable.

Attention Mechanism: We employ attention to select and focus on the important and essential hidden states of the Bi-LSTM layer. In particular, we use a multi-head self-attention mechanism [38, 60, 62] that jointly attends to information at different positions of the input sequence, with multiple individual attention functions and separately normalized parameters called *attention heads*. This enables it to capture different contexts in a fine-grained manner and learn long-range dependencies effectively. Each attention head computes a sequence z from the output $h = [h_1, h_2, ..., h_n]$ of the Bi-LSTM layer by projecting it to a key k, a value v, and a query q via distinct affine transformations with ReLU activations [15]. Here k, v and q each belong to the space $\mathbb{R}^{d/P}$, where P is the total number of attention heads. The attention weights a_{ijp} for attention head p between word tokens i and j are computed as:

$$a_{ijp} = \text{softmax}(\frac{q_{ip}^T k_{jp}}{\sqrt(d)})$$
$$z_{ip} = \textstyle\sum_j v_{jp} \odot a_{ijp}$$
$$z_i = \oplus z_{ip}; \forall p$$

Here $\odot$ denotes an element-wise product and *softmax* indicates the softmax function along the j-th dimension. The individual attention head outputs z_{ip} are concatenated into z_i for token i. The scaled dot product above enhances the optimization process by better distributing the gradients and flattening the softmax function [62].

3.1 Sequence Tagging via CRFs

The output of the attention layer serves as input to the next layer of OPINE's intent extraction model, namely a CRF [36]. CRFs effectively utilize the correlations between labels in a sequence neighborhood to predict the best label sequence for a given input. As mentioned earlier, the task of the CRF layer is to predict one of three tags for each word of the input sequence: Action, Object, or None. The input to the CRF layer is the sequence $z = [z_1, z_2, ..., z_n]$ from the attention layer, where z_i represents the i-th word token. y represents a certain output label sequence for z, and $Y'(z)$ denotes the possible set of label sequences. The conditional probability function for the CRF, $P(y|z; W, b)$, over all possible label sequences y given input sequence z is given by:

$$P(y|z; W, b) = \frac{\prod\limits_{i=1}^{n} \Psi_i(y_{i-1}, y_i, z)}{\sum\limits_{y' \in Y'(z)} \prod\limits_{i=1}^{n} \Psi_i(y'_{i-1}, y'_i, z)}$$

where $\Psi_i(y'_{i-1}, y'_i, z) = exp(W^T_{y',y} z_i + b_{y',y})$ are potential functions to be learned. $W^T_{y',y}$ and $b_{y',y}$ are weight and bias matrices corresponding to the label pair (y', y) respectively. We use linear chain CRFs with maximum conditional log-likelihood estimation.

Constraint-enhanced CRFs: The Viterbi algorithm [13] used for decoding the CRF layer only considers interactions between sequentially adjacent tag labels. However, we encounter additional constraints in our problem. First, we want to ensure that the CRF never predicts only an Action tag or only an Object tag, since our definition mandates the occurrence of both an action and the object it acts upon to constitute a valid intent. Next, it is often useful to identify *intent indicator* phrases that suggest the presence of an intent in the corresponding text, or are characteristic of an action following them. Since it is challenging to construct a comprehensive list of all such intent indicators, we pick a small number of highly indicative cues [20, 67]. These include: (i) presence of a first-person pronoun (e.g. *i, we*) within a three-word window of an infinitive verb phrase ('to' followed by a verb) in the utterance; and (ii) phrases denoting an 'action plan' (e.g. *plan to, want to* etc). For each such phrase, we selectively choose candidates having labelled intent tags in a small contextual neighbourhood (up to five words) following the intent indicator. These constraints operate at the level of the fully inferred sequence, and cannot be easily integrated into the Viterbi decoding algorithm by straightforward techniques like modifying its transition matrix [35, 52]. We circumvent this in two ways during the tag inference phrase of the CRF (Table 2). The first is using a *beam search* that penalizes sequences in the beam not satisfying the aforementioned constraints, and falls back to using the next most probable tag predictions.

Second, we use the fact that the solution output by the Viterbi algorithm is in fact the shortest path in a graph constructed among the sequence tokens and the possible tag values each token can take [52]. A sequence of length n with m possible tag labels is mapped to a graph with $nm + 2$ nodes and $(n-1)m^2 + 2m$ edges. We reduce this shortest path problem to an Integer Linear Programming problem, with added tag-specific constraints to it as inequalities between the graph node variables. These ensure that action and object tags do not occur in isolation, and certain indicator words increase the likelihood of a sequence being tagged with an action tag. This modified algorithm is then used to decode the CRF.

3.2 Generating Intents from Tag Sequences

Once the CRF predicts Action, Object and None tags for each input word, our final step is to match appropriate Action and Object tag phrases to generate meaningful intents. As specified earlier, we define an intent as a combination of Action tagged phrases followed by Object tagged phrases. We develop two techniques for this. First, we employ the simple but effective technique of linking Action and Object tagged phrases with respect to their word-based proximity in the input text. This distance-based heuristic assumes that related action-object phrases are likely to occur spatially close to each other. For instance, in the statement *"I would like to reserve a seat and also if possible, request a special meal on my flight"*, the action '*reserve*' is more likely to match with the nearer object '*seat*', than with the farther object '*special meal*'. But, this assumption may not hold depending on the way the text is worded.

Our second technique of matching Action-Object tagged phrases is by learning a multi-layer perceptron (MLP) classifier. The input features for the MLP consist of the sum of the pre-trained GloVe embeddings [50] of the words in the potential Action-Object intent phrase, concatenated with the normalized word distance value between the Action and Object phrases in the original input text. These features account for the word proximity of the intent terms, and their semantic likelihood of co-occurring in a single phrase. The input to the MLP is thus the feature representation of all possible paired combinations of the predicted Action and Object tagged phrases. The MLP contains two fully connected layers of ReLU units, followed by a fully connected layer of size one. It outputs a score y_{mlp} for each potential Action-Object pair under consideration, showing the probability of combining them to produce an intent. The MLP is trained with a margin-based hinge loss function, maximizing the separation between the true and the highest scoring incorrect Object option for the current Action phrase. We present the performance of both the above techniques in Table 3.

Our OPINE open intent extraction framework thus makes use of semantic information from the previous and future time steps, and dependency constraints learned and enforced by the CRF; to

Table 1: Statistics of our curated Stack Exchange dataset

Name of Genre	No. of utterances	Avg utterance length	Vocab size per genre
Data science	8184	60	11561
Software engineering	7114	60	23417
Web apps	7048	50	28906
Webmasters	7524	56	18688
Sharepoint	9366	60	40094
Productivity	8968	60	9529
Development ops	1660	60	1871
Open data	2166	60	7952
Server fault	7772	53	16047
Life hacks	1836	50	7837
DIY	2378	35	4140
CRM software[2]	11723	60	47219

predict open intents for an input text utterance. Multi-head self-attention enables it to learn dependencies between distant words (possibly across sentences) effectively. Adversarial training acts as a powerful regularizer for our model, contributing to its robustness and resilience to user intents from diverse domains.

4 EVALUATION

4.1 Data Collection

We collected about 75K questions with their top correct answer on various topical categories, from www.stackexchange.com. We formulated an Amazon Mechanical Turk experiment to annotate 25K of these with up to three intents that the crowd workers felt were most important or relevant. We observed an inter-annotator agreement of 0.73. We used the remaining 50K unlabeled questions for unsupervised pre-training, by generating *verb-object* parse tags for these texts via the Stanford CoreNLP dependency parser [44]. We employed words tagged as verbs and objects as proxies for the Action and Object tagged phrases that compose an intent. We then fine-tuned our model on 80% of the intent-labeled data tagged with Action and Object phrases by the annotators, and tested it on the remaining 20%. Our curated Stack Exchange dataset consists of 12 diverse genres with hundreds of unique intent types. (Table 1),

Most (if not all) intent detection benchmark datasets (e.g. SNIPS, ATIS) are typical of automated voice agents with short, concise text utterances and a single intent per utterance. These are quite distinct from help forum or user support style conversations, containing longer utterances with descriptive background context. We show empirically that a strength of OPINE is being able to handle both short utterances with limited to no context (e.g. SNIPS and ATIS data in Section 5), as well as longer conversational utterances (e.g. Stack Exchange in Table 2). We choose Stack Exchange as our data source due to its long and verbose text with background details, linguistic complexity and diversity, and multiple intents scattered throughout the text. We hope that such an intent-annotated dataset would be a novel contribution to the literature.

Implementation: We use the 300-dimensional GloVe embeddings [50] pre-trained on the Common Crawl dataset[3], and character embeddings as per Ma et al [42]. We use 400 LSTM units with L2 regularization and dropout [56] at the Bi-LSTM layer with a probability value of 0.5, to avoid overfitting and co-adaptation of the hidden units. Parameter optimization is performed via the Adam [34] optimizer with gradient clipping and early stopping based on the validation set. We set the initial learning rate to 0.001 with a decay of 0.05.

4.2 Results

Employing a consistent, domain agnostic representation for intents that contains an Action and an Object that it acts upon enables OPINE to identify and extract all possible intents which fit in this format, irrespective of their target domain. These include previously unseen intent types that were not encountered while training, unlike a classifier that can only address a pre-defined set of intent categories. This formulation also helps OPINE discover *multiple* possible intents for a single utterance and not just a single intent, unlike most of the current literature. This is crucial since user queries often consist of multiple tasks to be accomplished together (e.g. *reserve seat* and *request special meal* in the text *"I would like to reserve a seat and also if possible, request a special meal on my flight"*), or a single principal intent accompanying other interlinked intents. As specified in Sections 2 and 3, we do not consider intents containing only an action (e.g. *play, search*) without a qualifying object, since we believe that knowing the object entity of the action is equally important to holistically and unambiguously understand the intent of the user from the corresponding utterance.

Comparative Analysis on Stack Exchange Data: Table 2 shows the performance of various baseline approaches for open intent extraction on our curated Stack Exchange dataset. The first baseline levers a cue-based intent detection strategy [20, 67] that essentially returns as intents the phrases following the occurrence of 'intent-indicator' cue words or phrases (described in Section 3.1). The second baseline levers the verb-object tags learned by the Stanford dependency parser, used as proxies for Action and Object tags respectively. The third approach is a state-of-the-art deep semantic role labeling model with self attention [61]. Semantic role labeling is a shallow semantic parsing task that extracts various 'semantic roles', i.e. event properties and relations among relevant entities from an input utterance. In this work, we only focus on the two roles of verb and the object or entity acted upon by the verb as contributors to user intent. The second column of Table 2 reports the precision, recall and F1-score of the Action tags for each word of the input utterance, whereas the third column only assesses the Object tags. The fourth column displays the results considering the combination of both tag types to create an intent. The last column of semantic similarity computes the mean of the cosine similarities between the embeddings of the predicted and actual (annotated) intents. Each intent phrase's embedding is the average of the pre-trained GloVe [50] embeddings of its constituent words. We ignore the words whose embeddings do not exist. '*beam-CRF*' and '*constr-CRF*' in the last two rows refer to the two CRF enhancements from Section 3.1 of (i) considering a beam of probable tag sequences, and (ii) adding additional constraints to the decoding algorithm.

[2] A commercial Customer Relationship Management (CRM) software.

[3] nlp.stanford.edu/projects/glove/

Table 2: OPINE vs. State-of-the-art: precision(P), recall(R), F1-score and semantic similarity on Stack Exchange data

Approach	ACTION P/R/F1	OBJECT P/R/F1	Intent P/R/F1	Semantic Similarity
Cue-based Intent Detector [20, 67]	0.65/0.59/0.62	0.6/0.54/0.57	0.63/0.56/0.59	0.67
Stanford CoreNLP dependency parser (SC) [44]	0.56/0.49/0.52	0.51/0.43/0.47	0.53/0.45/0.49	0.59
Deep Semantic Role Labeling (SRL) [61]	0.79/0.63/0.7	0.69/0.62/0.65	0.7/0.62/0.66	0.75
OPINE (beam-CRF)	0.84/0.72/0.77	**0.81/0.69/0.75**	**0.82/0.70/0.76**	**0.86**
OPINE (constr-CRF)	**0.84/0.73/0.78**	0.81/0.68/0.74	**0.82/0.70/0.76**	**0.86**

We observe a significant improvement of OPINE of over 15% in terms of F1-score and semantic similarity, compared to the simple intent-indicator based model and the Stanford parser (first two rows of Table 2). The best performing variant of our proposed approach also significantly outperforms the SRL model (third row of Table 2) by about 9% F1-score points. These results show that OPINE can successfully filter out all the additional "non-intent" background information present in the input utterance, and only focus on the text needed to extract the user intent.

OPINE Design Choice Analysis: Table 3 describes the performance of different variations (design choices) of OPINE for open intent extraction on our curated Stack Exchange dataset. Except for the first row of Table 3, all other variants of OPINE are first pre-trained on the verb-object tags learned by a dependency parser (SC), followed by fine-tuning on the intent annotated utterances. '*train on MTurk*' denotes OPINE being trained only on the human annotated intent data. '*att*' and '*adv*' denote the presence of attention and adversarial training respectively in the model. *'w-dist'* and '*MLP*' denote the two methods of matching appropriate Action-Object phrases to create a holistic intent, from Section 3.2 based on (i) word proximity in the input text, and (ii) the score learned by the MLP classifier. Utilizing the dependency parser data as a pre-training step for the weights of our model, followed by fine-tuning on the actual intent-labeled data improves the F1-score by at least 6%. Enhancing the CRF decoding algorithm with added constraints (*beam-CRF* and *constr-CRF*) benefits the F1-score further by 2-5%. We find a performance difference of ≤ 3% between using the word proximity heuristic (*w-dist*), and the MLP classifier for matching Action and Object phrases. In general, the unsupervised word proximity heuristic is more efficient than the MLP classifier because it does not incur an additional training cost. Overall, OPINE trained with attention, adversarial training and CRF enhancements outperforms the alternative variations (Table 3) and state-of-the-art baselines (Table 2), with an intent F1 score of 76%, and a semantic similarity of 86% between the true and predicted intents.

Domain Adaptation Capability: Encountering newly emergent domains with little to no labeled intent categories available is a common real-world scenario. It is time-consuming and labor-intensive to obtain sufficient domain-specific annotated training data. It is thus desirable to adapt and generalize an existing trained model with minimum re-training effort, each time a new domain with potentially new intents is added. We investigate in Table 4 the capability of OPINE in adapting and transferring knowledge across distinct conversational domains. We consider several different test domains in the first column. The average overlap in the text vocabularies across pairwise domain combinations is 43%. We evaluate OPINE trained on utterances from the remaining domains other than the test domain in the second and fourth columns. The third and fifth columns indicate the respective F1-score and semantic similarity achievable for the test domain, when OPINE is trained using labeled data from the test domain as well. The F1-score and semantic similarity metrics are computed in a similar manner as for Tables 2 and 3. The difference in both metrics with and without using training data from the test domain is ≤ 5%, for most domains. Only the *Life Hacks* domain suffers a loss of 6.5% in terms of F1-score when we eliminate the data from this domain while training. Interestingly for the *DIY* domain, its training data is dominated by other semantically distinct domains. However, OPINE still attains a good F1-score of 72%, only 4% lesser than what is possible if *DIY* domain data is used for training. These results show that OPINE can effectively detect novel actionable intents in newly emerging low-resource domains with minimal manual effort.

Role of Attention: Table 3 indicates that the presence of attention lends OPINE an F1 score gain of at least 4%. We further explore OPINE's capability of identifying relevant and meaningful semantic features from its input utterances, which contribute in discovering open intents. We examine and visualize in Table 5 the self-attention values for specific utterances from our Stack Exchange dataset. For the sake of brevity we display truncated versions of the text inputs in the first column, and their associated intents in the second column. A darker colored highlight on a specific utterance word indicates that it receives higher attention, and plays a greater role during intent discovery. Input utterance words that constitute intents are marked in boldface. In all cases, we observe that words semantically related to and contributing to at least one intent are successfully identified by an attention head. For instance, the second row shows the significance of *'find out'*, *'retweeted'*, *'tweet'* and *'what their Twitter IDs are'* in deciding the intent *"find retweeted Twitter IDs"*. The attention heads are attentive to indicator cues that are likely to precede an actionable intent, such as *'possible to'*, *'want to be able to'*, *'how can I'* and *'I want to'*. Action or object phrases that are irrelevant to the user's intent (e.g. *'watching videos'* in the fourth row) do not receive a high attention score. Our attention mechanism can capture the dependency between distant intent words, such as *'find'* and *'retweeted'* in the second row and *'publish'* and *'scheduled'* in the third row. It also associates the action *'manage'* in the last row with two objects, *'sick notes'* and *'absences'*, generating the intents *"manage sick notes"* and *"manage absences"*.

4.3 Drill-down Analysis

Effect of Human-Annotated Training Data Size: In Tables 2 and 3, we showed that training our OPINE model with absolutely no human-labeled intent data is detrimental to its performance.

Table 3: Precision (P), recall (R), F1-score and semantic similarity of OPINE variants on the Stack Exchange dataset.

OPINE design variant	ACTION P/R/F1	OBJECT P/R/F1	Intent P/R/F1	Semantic Similarity
att+adv + train on MTurk + w-dist	0.75/0.59/0.66	0.74/0.52/0.61	0.74/0.55/0.63	0.74
att + SC (pre-train) + MTurk (fine tune) + w-dist	0.78/0.62/0.69	0.79/0.56/0.66	0.78/0.58/0.67	0.80
adv + SC + MTurk + w-dist	0.81/0.60/0.68	0.76/0.54/0.63	0.78/0.56/0.65	0.77
att+adv + SC + MTurk + w-dist	0.84/0.66/0.73	0.81/0.63/0.71	0.82/0.64/0.72	0.83
att+adv+beam-CRF + SC + MTurk + w-dist	0.84/0.70/0.76	0.81/0.67/0.73	0.82/0.68/0.74	0.84
att+adv+constr-CRF + SC + MTurk + w-dist	0.84/0.72/0.77	0.81/0.67/0.73	0.82/0.69/0.75	0.85
att+adv + SC + MTurk + MLP	0.84/0.68/0.75	0.81/0.67/0.73	0.82/0.67/0.74	0.84
att+adv+beam-CRF + SC + MTurk + MLP	0.84/0.72/0.77	**0.81/0.69/0.75**	**0.82/0.70/0.76**	**0.86**
att+adv+constr-CRF + SC + MTurk + MLP	**0.84/0.73/0.78**	0.81/0.68/0.74	**0.82/0.70/0.76**	**0.86**

Table 4: Studying OPINE's domain adaptation capability on multiple test domains. '+td' in the columns indicates that data from that particular test domain row is included while training, while '-td' indicates its exclusion while training.

Test Domain Name	F1 -td	F1 +td	Sim -td	Sim +td
Data science	0.76	0.8	0.84	0.88
Software engineering	0.69	0.74	0.81	0.86
Web apps	0.73	0.77	0.83	0.88
Webmasters	0.75	0.79	0.83	0.86
Sharepoint	0.71	0.76	0.82	0.85
Productivity	0.73	0.78	0.81	0.86
Development ops	0.71	0.73	0.78	0.83
Open data	0.69	0.73	0.84	0.87
Server fault	0.67	0.72	0.75	0.8
Life hacks	0.635	0.7	0.74	0.8
DIY	0.72	0.76	0.81	0.86
CRM software	0.79	0.83	0.88	0.91

Table 5: Effect of attention. Darker colored highlight shows a higher attention value. Boldface denotes presence of intent.

Input Text Utterance	Intents
Is it possible to **navigate** back ... **to previous page** after save processing? ... I have a page where I click on a link and use navigateURL ... want to be able to go back to the previous calling page and **complete** the **processing** of the **save**...	navigate previous page, complete processing save
The "Your tweets retweeted" page ... **find** out all the users who **retweeted** a tweet of mine? ... have retweeted a tweet and what their **Twitter IDs** are?	find retweeted Twitter IDs
Is there a WordPress plugin that will **tweet when** a scheduled post is posted? ... will tweet when you **publish** a post, but none I have tried will do it on a **scheduled post**.	tweet when publish scheduled post
How can I **keep** my **phone from** just **falling** over when watching videos? ... I also want to **have** my **hands free** to do other things ...	keep phone from falling, have hands free
I'm starting a micro-school... I want to **manage sick notes** and **absences** ... How can I **synchronize** one **central** Google **Calendar** ... Parents should be able to schedule future absences and excuse past absences...	manage sick notes, manage absences, synchronize central calendar

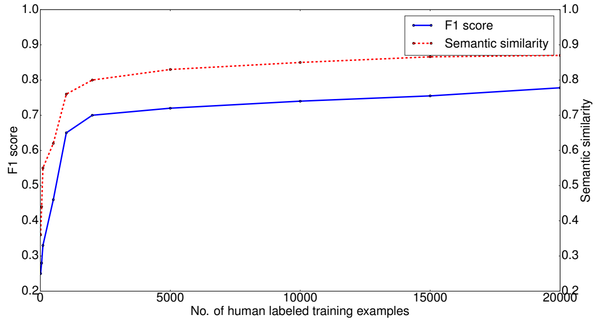


Figure 2: Effect of varying the amount of human labeled training data on the model performance of OPINE.

We now examine the effect of varying the amount of human annotated intent data while training OPINE. Figure 2 shows the F1-score (blue solid plot) and semantic similarity (red dashed plot) values for the predicted intents achieved by OPINE, as the number of human annotated training instances varies. We find that both plots are monotonically increasing. When the total number of human labeled training instances across various domains is less than 1000, the values of the F1-score and semantic similarity are below 50%. Both metrics rise to about 70% and 75% respectively at 1000 annotated training examples (less than 50 labeled examples per unique domain on average). Beyond this point, there is a steady performance improvement, with a less sharper gain than earlier. These observations reinforce OPINE's domain adaptivity and show that it does not require a large number of labeled examples per domain (less than 50 on average) to successfully perform intent discovery.

Grouping Related Intent Categories: The output of OPINE is an intent phrase for each input user utterance, and might therefore use distinct phrases to express similar intents. For instance, for the following semantically equivalent utterances: (i) *"Make a new haircut appointment for next Saturday"* and (ii) *"Can you reserve a time slot in the hair salon on Saturday?"*, OPINE predicts their intents as "make haircut appointment" and "reserve time hair salon". We might want to group together such semantically similar intents into a unified intent category. Therefore, as an additional post-processing step, we provide the deep features output by the attention layer of OPINE as input to an agglomerative hierarchical clustering algorithm, where the number of clusters is given by the number of distinct intent categories or domains that we would like to obtain (e.g. 12 from Table 1 in the case of our Stack Exchange dataset). We visualize the output vectors of the attention

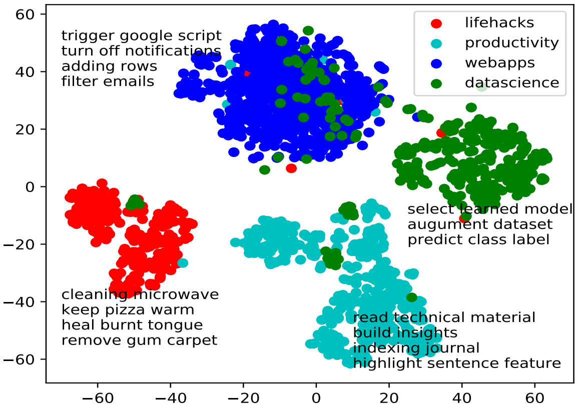


Figure 3: Visualizing the deep features learned by OPINE for four randomly selected Stack Exchange domains.

layer of OPINE for four random test domains from our curated Stack Exchange dataset in Figure 3. We use t-SNE [43] to reduce the vector dimensionality to two dimensions, and color the data points according to their domain labels. Some sample intents discovered by our method for each domain are also denoted in Figure 3. Domain names are treated as ground truth labels for the purpose of evaluation with respect to standard clustering metrics. We obtain a normalized mutual information (NMI) [58, 66] score of 0.71, a silhouette coefficient [53] of 0.69, a cluster purity [51] value of 0.78, and an F-measure [51] of 0.72. Figure 3 and the high purity value of our clustering arrangement demonstrate that good quality features are learnt by OPINE, which are largely compact within domains and separable across domains. Utterances belonging to similar intents and the same domain are also located close by in the embedding space (except for some domains like *Webapps* and *Data Science* that contain some utterances with similar intents). OPINE can thus be used to group together semantically or functionally related intents into higher level categories (the task of *domain discovery*).

5 CASE STUDIES

We now present qualitative and quantitative evaluations of OPINE on additional datasets. Note that we trained OPINE on *unrelated, out-of-domain* intent categories from Stack Exchange, before testing on these datasets to examine the performance of OPINE. This represents a challenging environment, akin to zero-shot learning [49, 55], where no information about the test data is known during training.

Ubuntu Dialog Corpus: We evaluate OPINE on a real, multi-turn conversation from the Ubuntu Dialog Corpus [41], a snippet of which is shown in Table 6. This dataset contains about one million technical support conversations related to the Ubuntu Linux operating system, and highly resembles real-world dialog exchanges between users and customer support agents. The original dialog from which this snippet has been truncated contains more than 100 turns with a lot of extraneous background information. It is between a user with technical issues and another who helps resolve them.

Table 6: Performance of OPINE on a technical support dialog snippet. Words that make up intents are shown in bold.

User: Can someone please help? I'm trying to **fix** a **broken ubuntu**.

Agent: ... how did you break it?

User: i'm on the cd and i'm trying to **mount** and then **chroot** my **hd**, which worked fine. I installed some new libs and now it no longer reboots.

User: what's the easiest way to **get** a **working boot** on my drive again?

Agent: ... sounds like something might be screwed up in your /etc/apt/sources.list file, if it's failing on apt-get update

User: how can i **fix** my **sources.list file**?

Agent: open /etc/apt/sources.list. see if you notice any obvious errors

User: a question on the mounting issue - when i loaded the cd, my local hard drive was mounted in media, can't i just use that as the chroot?

Agent: ... assuming your flgrx is hosed, move the x conf file out of the way so that the radeon driver will be used instead ...

User: drivers all back to normal. thank you so much my friend.

User: ... what do you suggest for a good backup program for ubuntu?

User: ... i installed the latest radeon drivers manually. how do i **upgrade** to the **newest kernel** and default **radeon drivers**?

Agent: first you'd uninstall 10.6 fglrx driver. then you'd grab the three 2.6.34 deb packages and then install xorg-edgers repo. run grub-update so it finds the new kernel and done.

User: where do i **get** the **debs**? and i know how to **uninstall** the **fglrx drivers** ..., and then do i **copy** back the **xorg.conf.original** to xorg.conf?

User: ... do i need to add a source to my source list?

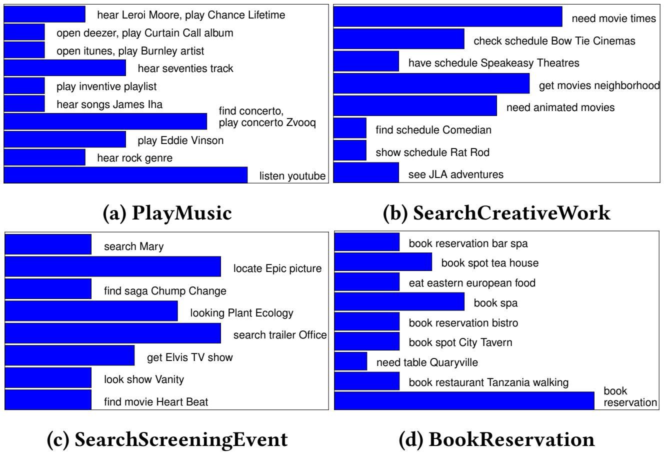
Agent: yes you need xorg-edgers (google it)

User: ok cool. how do i **get rid** of **xorg**, or is that already done?

Agent: ... if you used jockey-gtk to install fglrx and no other method, then you should be able to use the same method to remove them

Such data is often asynchronous with diverse and informal intents, dialog domains and semantic slots; which increases the difficulty of intent discovery. As mentioned earlier, we used OPINE pre-trained on *unrelated* intent genres from our Stack Exchange dataset, before testing on this multi-turn dialog. Our goal here is to understand the actionable intents of the user requesting support (**User** in Table 6), and not the one providing it (**Agent** in Table 6). We also seek to handle utterances containing intents with *both* an action and an object. The words constituting intents inferred by OPINE have been highlighted in boldface. Though each training utterance has up to three labeled intents, OPINE can detect more than three intents for an input if applicable. OPINE recognizes the following user intents in the conversation: *fix broken ubuntu, mount hd, chroot hd, get working boot, fix sources.list file, upgrade newest kernel, upgrade radeon drivers, get debs, uninstall fglrx drivers, copy xorg.conf original* and *get rid xorg*. Once these fine-grained intents have been recognized, they can be subsequently grouped into coarser level intents or domains (for instance, using the technique for Figure 3), depending on the downstream application task. Categorizing such a multi-turn dialog is typically outside the scope of existing intent classification systems, but OPINE provides a fine-grained summary of user intents throughout the dialog. Besides, having a common format to represent an intent contributes immensely in finding user intents irrespective of their target domain or topic.

Performance on SNIPS and ATIS: We next discuss the performance of OPINE on standard intent detection datasets used in the literature, namely the SNIPS NLU [10] and ATIS [11] datasets. We highlight in Figure 4 the benefit of OPINE in drilling down into high-level intent categories, to understand, summarize or hierarchically organize the specific fine-grained intents that they comprise of. An additional side benefit of discovering intents using OPINE is that it can identify relevant accompanying *slots* apart from the intents, without performing a dedicated slot filling task. For instance, in the *PlayMusic* category of the SNIPS dataset in Figure 4(a), OPINE



(a) PlayMusic **(b) SearchCreativeWork**

(c) SearchScreeningEvent **(d) BookReservation**

Figure 4: Fine-grained intents discovered by OPINE for four intent categories in the SNIPS NLU dataset. The length of the bars represents the relative frequency of that particular intent in the input data.

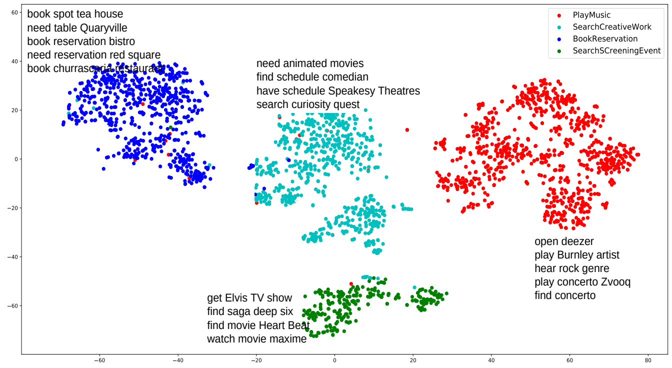


Figure 5: Visualizing the clustering arrangement for utterances belonging to four intent categories in the SNIPS NLU dataset. This illustrates OPINE’s ability to conflate different intent phrases mapping to the same or similar intent category. E.g. ‘book spot tea house’, ‘need table’, ‘need reservation’ and ‘book reservation bistro’ map to the same cluster.

not only recognizes the basic intents of *‘hear song’* or *‘play album’*; but also the corresponding names of singers (e.g. *Leroi Moore, Eddie Vinson*), song albums (e.g. *Curtain Call*, *Concerto*), and music platforms (e.g. *Youtube*, *Zvooq*). In Figure 4(c), users wish to search for screenings of events. OPINE accurately predicts this via keywords like *search, locate, find* and *look*. Moreover, OPINE also provides added information on the specific events that need to be searched, such as the *Chump Change* saga and the movie *Heart Beat.*

We emphasize here that OPINE *cannot* be directly compared to existing intent detection techniques in general since these are formulated as classification tasks over a fixed set of pre-defined intent categories, that need to be known at training time. They also typically require sufficient labeled training data for each intent category. In contrast, OPINE can handle an unlimited number of distinct intent classes, and has no restrictions on the number of training examples needed per intent type. However, to compare OPINE with existing intent classification approaches, we reformulate it to perform intent detection, using the output of the multi-head attention layer (OPINE + classifier in Table 7). As detailed earlier, we provide the deep features output by the attention layer of OPINE as input to an agglomerative hierarchical clustering algorithm, where the number of clusters is given by the number of distinct intent classes in the test set. Thus “OPINE + classifier” works as a good feature extraction layer across datasets without additional training, which is a benefit over current state-of-art techniques.

Table 7: F1 score of various intent classification approaches on the SNIPS and ATIS datasets.

Approach	SNIPS	ATIS
Joint Seq [21]	0.95	0.91
Attention BiRNN [39]	0.95	0.90
Slot-Gated Full Atten. [17]	0.95	0.92
DR-AGG [16]	0.95	0.90
IntentCapsNet [70]	**0.96**	**0.94**
OPINE + classifier	**0.96**	0.93
OPINE + Joint-Seq	0.95	0.91
OPINE + Attention Bi-RNN	0.95	0.90
OPINE + Slot-Gated Full Atten.	0.95	0.92
OPINE + IntentCapsNet	0.95	0.93

Note that OPINE was trained on *out-of-domain* intent data (Stack Exchange), since we do not have Action-Object annotations available for SNIPS or ATIS. In other words, this represents a challenging environment to examine the performance of OPINE (akin to zero-shot learning [49, 55]), where no information is available about the test data. Similar to what we showed in Figure 3 for the Stack Exchange dataset, in Figure 5 we show OPINE’s ability to conflate different intent phrases mapping to the same or similar intent category. We show a 2-D t-SNE visualization of the deep features output by the attention layer of OPINE for the SNIPS dataset. For instance, the intent phrases *play Burnley artist* and *hear rock genre* map to the same intent category (called *PlayMusic* in the SNIPS dataset), while the phrases *book reservation bistro* and *need table Quaryville* map to the same SNIPS category of *BookRestaurant.*

In addition to this qualitative analysis, we present a quantitative evaluation on SNIPS and ATIS in Table 7. It shows that the F1-score of OPINE is on par with state-of-the-art intent classification approaches on these datasets, despite not having seen any SNIPS or ATIS data during training. The SNIPS and ATIS utterance lengths are also about 2-6 times shorter than the Stack Exchange texts. This shows that OPINE is equally effective at finding intents in shorter inputs with no or limited additional context. The last four rows in Table 7 show the performance of the baselines when the intent-tagged words from the output of OPINE are fed to them as inputs. These inputs only contain the words tagged as Action or Object by OPINE, without any additional context. Their lengths range from 20-50% of the original input utterance. We find that just using OPINE as an initial step before intent classification has near identical performance as the baselines themselves (rows 1, 2, 3 and 5 of Table 7), despite OPINE not having seen any SNIPS or

ATIS data during training. This further demonstrates the domain adaptation capability and effectiveness of OPINE.

6 RELATED WORK

We discuss prior work on intent detection, as well as the related tasks of information extraction and semantic parsing.

Intent Detection: Prior work on intent detection encompasses two avenues: asynchronous, written communication (forums, blogs, tweets) and synchronous dialog. In both cases, intent detection is largely modeled as a classification problem, with each class representing the presence or absence of a specific kind of intent. Supervised and semi-supervised learning models based on linguistic and sentiment features have been used to model racial intent [1], and purchase intent online [9, 20, 67]. Wang et al [67] proposed a graph-based optimization approach for semi-supervised classification of tweets with a purchasing intent. Recent approaches including automated dialog response agents like Microsoft LUIS[4] and Google Dialogflow[5] have drawn upon progress in CNN, RNN, and transformer based language models to improve intent detection [6, 17, 30, 32, 39, 45, 54, 71] and domain detection [29, 31, 73]. Performing slot filling jointly with intent detection has improved the performance of both tasks [12, 30, 39, 68, 74]. However, the above approaches are either supervised or semi-supervised, assume a closed-world setting, and require sufficient quantities of labeled data for each intent type to perform intent detection.

In the open world setting, several interesting techniques were proposed to recognize input texts belonging to unseen or novel domains [29] and intents [37, 70] respectively. Xia et al [70] model intent detection as a zero-shot classification problem, but they assume that a list of new or unseen (during training) intent classes is available at test time along with some prior knowledge about the new intents to be discovered. Similarly, the methods proposed by Lin et al [37] and Kim et al [29] can only identify if an input utterance is likely to contain a novel intent. They do not discover the number of new intents, or specify what the new intents are. OPINE takes the next step in this regard.

Cai et al [5] used hierarchical clustering to learn a taxonomy of intent classes [28, 63, 64], and applied a hybrid CNN-LSTM model to classify the intent of medical queries. We take this idea further and learn to identify arbitrary intents beyond even a predefined taxonomy or schema. Improvements in intent detection and slot filling based on adversarial learning have been explored [33, 40, 72]. We exploit adversarial training to generate adversarial input examples to improve the performance of OPINE, and for cross-domain adaptation. Further, adding linguistic structure to existing models has improved their performance across related NLU tasks such as word embedding [47], machine translation [8], named entity recognition [27], and semantic role labelling [23]. We impose linguistic constraints on the CRF layer of OPINE to preserve the semantics of intent actions and their associated objects (Section 3.1).

Open Information Extraction and Semantic Parsing: There has been a lot of recent work in the literature in the areas of Open Information Extraction (IE) (see [48] for a recent survey). IE addresses the task of information extraction, i.e., of relating arguments and phrases expressed in unstructured text using a relation of the form *<arg1, rel, arg2>*. OpenIE refers to the task of building domain independent IE frameworks, where the relations to be extracted need not be specified in advance. In OpenIE frameworks, relations are expressed as triples and each component of the triple must be present in the input text. However, this is quite different from the open intent discovery task we define in this paper in the following ways: First, an OpenIE model does not distinguish between pieces of information that express an *intent*, and those that do not. Post-processing of relevant relations would be necessary to identify expressed intent before OpenIE systems could be used for the intent discovery task. Second, existing OpenIE systems extract information only at the level of sentences, unlike our proposed approach which can extract intents spanning *multiple* sentences (see Table 5 for examples). For these reasons, we do not compare our proposed approach OPINE with existing OpenIE models.

Semantic Role Labeling (SRL) (and also slot filling) approaches aim at creating shallow semantic parses, assigning roles to different phrases in input sentences. Semantic roles typically correspond to slots in predefined frames/templates (eg. Propbank [4]). Recent work on SRL aims to relax the assumption of pre-specified templates to move towards out-of-domain SRL [22, 23, 59]. Though SRL is a task related to intent discovery, SRL labels are typically more dense, requiring not just the intent labels, but other subject-predicate, named entity, etc. relations to be labeled. However for the purpose of completeness in evaluation, we do compare our framework against a state-of-the-art SRL baseline [61] in Table 2. OPINE siginificantly outperforms this baseline for intent discovery.

7 CONCLUDING REMARKS

We introduce and address the novel problem of open intent discovery via a sequence tagging approach, OPINE, in contrast to the common method of detecting intents via classification. OPINE harnesses a Bi-LSTM and a CRF coupled with self-attention and adversarial training. It can extract multiple actionable intent types from user utterances, many of which may be unseen during training. Extensive experiments on real-world datasets show substantial improvements of OPINE over competitive baselines. We also developed and plan to release a large collection of 25K intent-annotated instances from diverse domains on Stack Exchange. We demonstrate OPINE's ability to adapt across domains, minimizing the labeling effort needed on encountering a new domain with potentially new intents. OPINE provides an in-depth, fine-grained understanding of users' prospective actions and intentions from their text utterances, which can greatly benefit downstream conversational applications. Promising future directions include (i) learning generative rather than extractive models for open intents; (ii) inferring implicitly mentioned, or more generic non-actionable (e.g. information-seeking) intents from user utterances; and (iii) using visual and/or auditory inputs to learn intents in a multimodal way (e.g. [65, 69]).

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation grant EAR-1520870, the Ohio Supercomputer Center [7] and gift funding from Adobe Research. All content represents the opinions of the authors, and is not necessarily endorsed by their sponsors.

[4] www.luis.ai

[5] https://dialogflow.com/

REFERENCES

[1] Swati Agarwal and Ashish Sureka. 2017. Characterizing Linguistic Attributes for Automatic Classification of Intent Based Racist/Radicalized Posts on Tumblr Micro-Blogging Website. *arXiv preprint arXiv:1701.04931* (2017).

[2] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*. 344–354.

[3] Aditya Bhargava, Asli Celikyilmaz, Dilek Hakkani-Tür, and Ruhi Sarikaya. 2013. Easy contextual intent prediction and slot detection. In *IEEE ICASSP*.

[4] Claire Bonial, Julia Bonn, Kathryn Conger, Jena D Hwang, and Martha Palmer. 2014. PropBank: Semantics of New Predicate Types.. In *LREC*. 3013–3019.

[5] Ruichu Cai, Binjun Zhu, Lei Ji, Tianyong Hao, Jun Yan, and Wenyin Liu. 2017. A CNN-LSTM Attention Approach to Understanding User Query Intent from Online Health Communities. In *IEEE ICDMW Workshops*.

[6] Giuseppe Castellucci, Valentina Bellomaria, Andrea Favalli, and Raniero Romagnoli. 2019. Multi-lingual Intent Detection and Slot Filling in a Joint BERT-based Model. *arXiv preprint arXiv:1907.02884* (2019).

[7] Ohio Supercomputer Center. 1987. Ohio Supercomputer Center. http://osc.edu/ark:/19495/f5s1ph73

[8] Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. 2017. Improved neural machine translation with a syntax-aware encoder and decoder. *arXiv preprint arXiv:1707.05436* (2017).

[9] Zhiyuan Chen, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Identifying intention posts in discussion forums. In *NAACL-HLT*.

[10] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, et al. 2018. Snips Voice Platform: an embedded Spoken Language Understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190* (2018).

[11] Deborah A Dahl, Madeleine Bates, Michael Brown, William Fisher, et al. 1994. Expanding the scope of the ATIS task: The ATIS-3 corpus. In *Proceedings of the workshop on Human Language Technology*.

[12] Mauajama Firdaus, Shobhit Bhatnagar, Asif Ekbal, and Pushpak Bhattacharyya. 2018. A Deep Learning Based Multi-task Ensemble Model for Intent Detection and Slot Filling in Spoken Language Understanding. In *International Conference on Neural Information Processing*.

[13] G David Forney. 1973. The viterbi algorithm. *Proc. IEEE* 61, 3 (1973).

[14] Yaroslav Ganin and Victor Lempitsky. 2014. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495* (2014).

[15] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of AISTATS*.

[16] Jingjing Gong, Xipeng Qiu, Shaojing Wang, and Xuanjing Huang. 2018. Information Aggregation via Dynamic Routing for Sequence Encoding. *arXiv preprint arXiv:1806.01501* (2018).

[17] Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-Gated Modeling for Joint Slot Filling and Intent Prediction. In *Proceedings of NAACL-HLT*.

[18] Ian J Goodfellow, Jonathon Shlens, and Christian E Szegedy. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

[19] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *IEEE ICASSP*.

[20] Vineet Gupta, Devesh Varshney, Harsh Jhamtani, Deepam Kedia, and Shweta Karwa. 2014. Identifying Purchase Intent from Social Posts.. In *ICWSM*.

[21] Dilek Hakkani-Tur, Gokhan Tur, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-Domain Joint Semantic Frame Parsing using Bi-directional RNN-LSTM. In *Proceedings of Interspeech*.

[22] Silvana Hartmann, Ilia Kuznetsov, Teresa Martin, and Iryna Gurevych. 2017. Out-of-domain FrameNet Semantic Role Labeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, 471–482. https://www.aclweb.org/anthology/E17-1045

[23] Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of ACL*.

[24] Charles Hemphill, John Godfrey, and George Doddington. 1990. The ATIS spoken language systems pilot corpus. In *a Workshop on Speech and Natural Language*.

[25] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* (1997).

[26] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).

[27] Charles Jochim and Lea Deleris. 2017. Named Entity Recognition in the Medical Domain with Constrained CRF Models. In *Proceedings of EACL*.

[28] David Jurgens and Mohammad Taher Pilehvar. 2016. SemEval-2016 task 14: semantic taxonomy enrichment. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. 1092–1102.

[29] Joo-Kyung Kim and Young-Bum Kim. 2018. Joint learning of domain classification and out-of-domain detection with dynamic class weighting for satisficing false acceptance rates. *arXiv preprint arXiv:1807.00072* (2018).

[30] Joo-Kyung Kim, Gokhan Tur, Asli Celikyilmaz, Bin Cao, and Ye-Yi Wang. 2016. Intent detection using semantically enriched word embeddings. In *IEEE SLT*.

[31] Young-Bum Kim, Dongchan Kim, Anjishnu Kumar, and Ruhi Sarikaya. 2018. Efficient large-scale neural domain classification with personalized attention. In *Proceedings of ACL*.

[32] Young-Bum Kim, Sungjin Lee, and Karl Stratos. 2017. Onenet: Joint domain, intent, slot prediction for spoken language understanding. In *ASRU workshop*.

[33] Young-Bum Kim, Karl Stratos, and Dongchan Kim. 2017. Adversarial adaptation of synthetic or stale data. In *Proceedings of ACL*.

[34] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[35] Trausti Kristjansson, Aron Culotta, Paul Viola, and Andrew McCallum. 2004. Interactive information extraction with constrained conditional random fields. In *AAAI*.

[36] John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

[37] Ting-En Lin and Hua Xu. 2019. Deep Unknown Intent Detection with Margin Loss. *arXiv preprint arXiv:1906.00434* (2019).

[38] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130* (2017).

[39] Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. *arXiv preprint arXiv:1609.01454* (2016).

[40] Bing Liu and Ian Lane. 2017. Multi-domain adversarial learning for slot filling in spoken language understanding. *arXiv preprint arXiv:1711.11310* (2017).

[41] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *arXiv preprint arXiv:1506.08909* (2015).

[42] Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354* (2016).

[43] Laurens Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* (2008).

[44] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of ACL*.

[45] Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, et al. 2015. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* (2015).

[46] Takeru Miyato, Andrew Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725* (2016).

[47] Nikola Mrkšić, Diarmuid O Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, et al. 2016. Counter-fitting word vectors to linguistic constraints. *arXiv preprint arXiv:1603.00892* (2016).

[48] Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh. 2018. A Survey on Open Information Extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, Santa Fe, New Mexico, USA, 3866–3878.

[49] Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. 2009. Zero-shot learning with semantic output codes. In *Advances in neural information processing systems*. 1410–1418.

[50] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of EMNLP*.

[51] Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*.

[52] Dan Roth and Wen-tau Yih. 2005. Integer linear programming inference for conditional random fields. In *Proceedings of ICML*.

[53] Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* (1987).

[54] Prashanth Shivakumar, Mu Yang, and Panayiotis Georgiou. 2019. Spoken Language Intent Detection using Confusion2Vec. *arXiv preprint arXiv:1904.03576* (2019).

[55] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In *Advances in neural information processing systems*. 935–943.

[56] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR* (2014).

[57] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387* (2015).

[58] Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *JMLR* (2002).

[59] Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. *arXiv preprint arXiv:1804.08199* (2018).

[60] Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2017. Deep semantic role labeling with self-attention. *arXiv preprint arXiv:1712.01586* (2017).

[61] Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep Semantic Role Labeling with Self-Attention. In *AAAI Conference on Artificial Intelligence*.

[62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, et al. 2017. Attention is all you need. In *NIPS*.

[63] Nikhita Vedula, Pranav Maneriker, and Srinivasan Parthasarathy. 2019. BOLT-K: Bootstrapping Ontology Learning via Transfer of Knowledge. In *The World Wide Web Conference*. 1897–1908.

[64] Nikhita Vedula, Patrick K Nicholson, Deepak Ajwani, Sourav Dutta, Alessandra Sala, and Srinivasan Parthasarathy. 2018. Enriching Taxonomies With Functional Domain Knowledge. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 745–754.

[65] Nikhita Vedula, Wei Sun, Hyunhwan Lee, Harsh Gupta, Mitsunori Ogihara, Joseph Johnson, Gang Ren, and Srinivasan Parthasarathy. 2017. Multimodal Content Analysis for Effective Advertisements on YouTube. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1123–1128.

[66] Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *JMLR* (2010).

[67] Jinpeng Wang, Gao Cong, Wayne Xin Zhao, and Xiaoming Li. 2015. Mining User Intents in Twitter: A Semi-Supervised Approach to Inferring Intent Categories for Tweets.. In *AAAI*.

[68] Yu Wang, Yilin Shen, and Hongxia Jin. 2018. A bi-model based rnn semantic frame parsing model for intent detection and slot filling. *arXiv preprint arXiv:1812.10235* (2018).

[69] Laura Wendlandt, Rada Mihalcea, Ryan L Boyd, and James W Pennebaker. 2017. Multimodal analysis and prediction of latent user dimensions. In *International Conference on Social Informatics*. Springer, 323–340.

[70] Congying Xia, Chenwei Zhang, Xiaohui Yan, Yi Chang, and Philip S Yu. 2018. Zero-shot User Intent Detection via Capsule Neural Networks. *arXiv preprint arXiv:1809.00385* (2018).

[71] Puyang Xu and Ruhi Sarikaya. 2013. Convolutional neural network based triangular CRF for joint intent detection and slot filling. In *ASRU workshop*.

[72] Qian Yu and Wai Lam. 2018. Product Question Intent Detection using Indicative Clause Attention and Adversarial Learning. In *Proceedings of ACM SIGIR*.

[73] Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip S Yu. 2018. Joint Slot Filling and Intent Detection via Capsule Neural Networks. *arXiv preprint arXiv:1812.09471* (2018).

[74] Xiaodong Zhang and Houfeng Wang. 2016. A Joint Model of Intent Determination and Slot Filling for Spoken Language Understanding. In *IJCAI*.

[75] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*.