Underspecification in Natural Language Understanding for Dialog Automation

John Chen
Interactions LLC
41 Spring Street, Suite 106
Murray Hill, NJ 07974 USA
jchen@interactions.com

Srinivas Bangalore
Interactions LLC
41 Spring Street, Suite 106
Murray Hill, NJ 07974 USA
sbangalore@interactions.com

Abstract

With the increasing number of communication platforms that offer variety of ways of connecting two interlocutors, there is a resurgence of chat-based dialog systems. These systems, typically known as *chat-bots* have been successfully applied in a range of consumer and enterprise applications. A key technology in such chat-bots is robust natural language understanding (NLU) which can significantly influence and impact the efficacy of the conversation and ultimately the user-experience. While NLU is far from perfect, this paper illustrates the role of *underspecification* and its impact on successful dialog completion.

1 Introduction

With the coming of age of speech recognition technology, everyday spoken language expressions can be converted into text with a degree of accuracy that creates unprecedented opportunities for realizing natural and sophisticated human-machine conversations to accomplish routine tasks. In addition, there is a resurgence of text-based conversation systems, such as SMS and Chat, owing to the availability of communication platforms that make it convenient to configure human-machine *text* driven conversations. The potential opportunities of speech/text driven human-machine conversational system, collectively known as *virtual agents*, can only be realized if the user's request in spoken or typed form are *understood* by the virtual agent and acted on appropriately to fulfill users' requests.

It is often difficult to represent the precise meaning of a sentence given the inherent ambiguity in natural language. Furthermore, interpreting a user's sentence in the conversation context that is modulated by the user's perspective, further exacerbates the challenge of extracting the precise pragmatic import of a user's request. For practical applications, such as conversational agents and speech/text analytics systems, the meaning of a sentence may be approximated as one or more *actionable* labels associated with the input utterance. Such *actionable* labels are termed as *intents* and an *intent model* is used to uncover the intent from a user's utterance.

In practical applications, intent labels often conflate the linguistic meaning of a user sentence, the conversation context in which the utterance appears, and the rules of business that apply to that application. The intent model then directly disambiguates a user request and maps it to a representation that can be interpreted by the downstream system without the need for further inference mechanisms. Consequently, most of the intent labels in an application domain are specific to that particular domain and sometimes are even crafted specifically to the flow of a particular dialog system. While such a direct approach limits the reusability of the intent labels across application domains, the benefit of such an approach is it circumvents the intricacies of conventional knowledge acquisition and domain modeling that is necessary for inference on a domain-independent meaning representation.

The actionable meaning representation of a user input also identifies the target of the intent, typically called an *entity*. An entity is that part of the user request that identifies the object of the intent. Some examples of application-independent entities include a *person name, date, phone number, credit card, account number*. In addition to such generic entities, domain-specific entities such as names of services and names of products need to be identified. In order to provide an actionable representation, the orthographic differences in ref-

Proceedings of Recent Advances in Natural Language Processing, pages 164–170,
Varna, Bulgaria, Sep 4–6 2017.
https://doi.org/10.26615/978-954-452-049-6_023

erences to an entity need to be normalized and rendered in a canonical form (e.g. *Christmas, 12/25, December 25*).

In this paper, we discuss the NLU component of an enterprise-grade customer care chat system and highlight the benefits of under-specification of intents in automating chats. In Section 2, we discuss the details of the chat system and present the NLU model in Section 3. The impact of underspecification in NLU is discussed in Section 4 along with experiments and results.

2 Chat-based Customer Care

A chat-based customer care system we use in this study is shown in Figure 1. The user, or *Chat Client* interacts with the *Chat Platform* by typing in chat text in response to a system prompt that is displayed by the chat platform. The chat text is first routed to the *NLU* to determine the intent of the user's text. If the NLU determines the intents from the text with high confidence, the text is routed to the Dialog Manager through the Chat Platform. If the NLU is unable to determine the intent, the Chat Platform routes the text to a human (*Intent Analyst*) for intent determination. The intents identified from the user's text are used to update the dialog state in a *Dialog Manager*. Depending on the identity of the new dialog state, the Dialog Manager may prompt the user for more information, via the Chat Platform.

3 NLU in a Care Chatbot

A single chat utterance is tagged with three types of labels – *Intents*, *Entities*, and *Conversational handlers*. Intents are domain-specific labels such as *SALES*, *TECH_ASSISTANCE* and *BILLING* and generally expresses the action intended by the customer. There are 85 different types of intents in the customer care chat system under study. *Entity* labels represent the names of products or services mentioned in the user's text. These include, for example, entities such as *Headphones*, *Mobile Phone*, and *Insurance*. There are a total of 115 types of entities in the chat system. There is an ontology defined over intents and entities. For example, the general intent, *SALES*, is composed of more specific intents such as *PRICING, DISCOUNTS* or *LEASING*. An abridged version of the intent ontology is shown in Figure 2. Finally, conversational handlers are labels which are similar to speech acts, and guide the conversation, such as *LIVE AGENT* or *CONFUSED*. There are 15 types of conversational handlers in the current system.

3.1 NLU Models

Three independent SVM classifiers are trained, one for each of intents, entities, and conversational handlers. Each of the classifier is modeled as a set of binary SVM classifiers, with each binary classifier predicting if the input is assigned or not assigned a particular label type. Consequently, for a given input chat utterance x, classification involves computing the following for each of intent (tn), entity (sv) and conversational labels (en).

$$y^* = \arg\max_{y \in tn,sv,en} F_y(x, y) \quad (1)$$

Subsequently, the joint label is defined as $\langle tn^*, sv^*, en^* \rangle$.

The different classifiers use the same feature set $\mathcal{F}$, which is composed of three types of features – *application context*, *conversation context*, *utterance*. Features from the application context include the URL, the section and the subsection of the web page from which the chat session originated. Conversation context includes the dialog state identifier, and a bag of ngrams (unigrams and bigrams) extracted from all the utterances from the dialog until the current utterance. Utterance features are a bag of word ngrams that are extracted from the current user utterance.

3.2 Confidence Measures

In an attempt to boost the accuracy, we reject the decisions made by the classifier based on a measure of confidence computed using the score generated by the classifier. We experiment with the effect on accuracy that use of different confidence measures elicit.

We use the sigmoid to obtain probabilities from the scores assigned by the SVM classifiers.

$$P(tn^*) = \frac{1}{1 + exp(F_{tn}(x, tn^*))} \quad (2)$$

This can be extended to a simple confidence measure on joint labels (*joint probability*) as in Equation 3.

$$jointprob = min(P(tn^*), P(sv^*), P(en^*)) \quad (3)$$

We define another confidence measure (Equation 4) as the ratio of the probabilities of the first

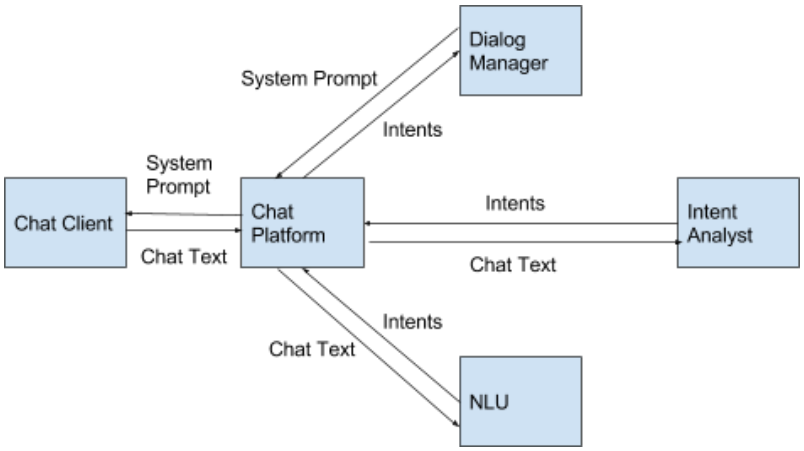

Figure 1: Architecture of the Chat Dialog system

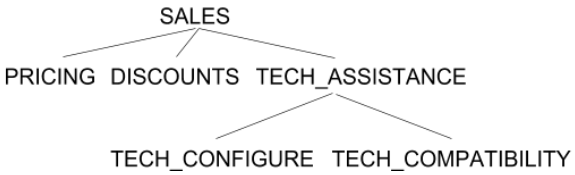

Figure 2: A sample view of the intent ontology.

(tn^*) and second best (tn^{*-1}) labels assigned by individual classifiers.

$$cf(tn^*) = \frac{P(tn^*)}{P(tn^{*-1})} \quad (4)$$

In a similar manner as before, this can be extended to a confidence measure on joint labels (*joint ratio*) as in Equation 5.

$$cf_{jt} = min(cf(tn^*), cf(sv^*), cf(en^*)) \quad (5)$$

4 Experiments and Results

The training corpus of 1.6 million examples of chat turns and test corpus of 57K chat turns are tagged with intents, entities and conversation handler labels.

Table 1 shows accuracies of the resulting classifiers. These are compared against baseline accuracies, which are obtained by tagging every test instance with the majority class type label. Conversational handlers are the easiest to predict, followed by Entities and Intents. Conversational handlers are easy because most turns are labeled as INFORM. Entities are moderately difficult. Its difficulty is similar to that of named entity recognition or entity linking. Complications include noisy text input or the need to find the main entity if multiple entities exist in the input. Unlike named entity recognition, there is no need to find the exact boundary of the entity mention in the input text. Intents are the most difficult.

Classification Task	Baseline	Accuracy
Intents	41.85	90.65
Entities	35.59	93.92
Conversational	90.81	99.01
Joint	7.41	84.86

Table 1: Accuracies of the different classifiers

Figure 3 shows accuraries of these classifiers as a function of training data size. As can be inferred by its relative ease of prediction, the classifier for conversational handlers has an accuracy with respect to training data size that plateaus early. Less than 200K samples can train a classifier with an acccuracy within 1.0% of one trained on 1.6 million samples. In contrast, accuracies of intent and entity classifiers increase more quickly as more samples are introduced into the training data.

The accuracies of the different classifiers as plotted against automation rate are shown in Figure 4. Here, a classifier returns a result if its confidence score is not less than a predefined threshold. The automation rate is the percent of input user utterances for which the classifier returns a result. The classifier accuracy is the percent of returned

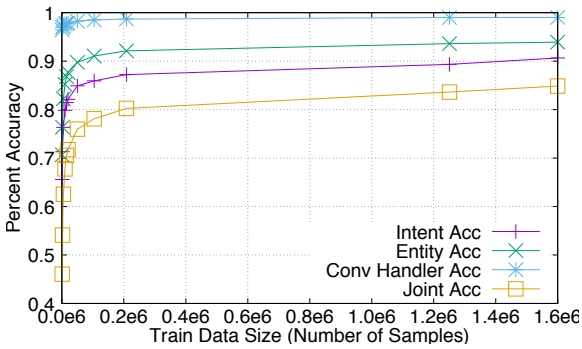

Figure 3: Classifiers' Accuracies as a Function of Training Data Size

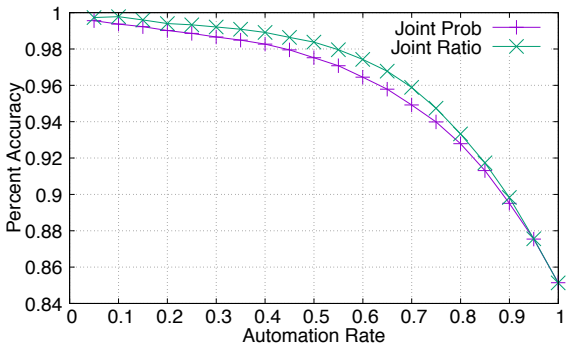

Figure 5: Accuracy vs. Automation for Different Confidence Metrics

results that are correct. Within such a framework, the accuracy of joint label prediction can be 90% or higher at an automation rate of 88% or lower.

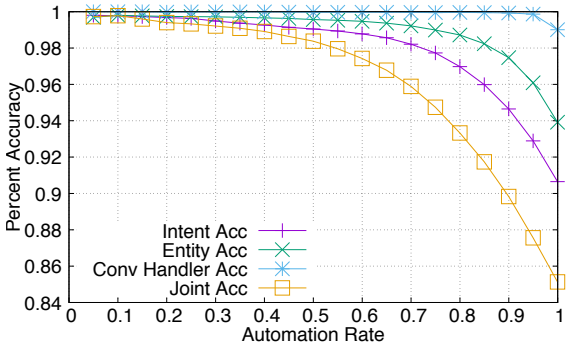

Figure 4: Accuracies of Different Classifiers versus Automation Rate

Instead of thresholding using joint probability, we may threshold the classifiers results using joint ratio. These two kinds of confidence scores are compared in an accuracy versus automation graph for joint labeling, as shown in Figure 5. Using joint ratio always gives better results. At 90% automation, joint ratio gives about 0.3% improvement in accuracy. The improvement in accuracy is only larger for automation rates between 30% and 90%.

The top 10 confusion pairs of the Task Names classifier are shown in Table 2. Most of them involve misclassification of SALES, either as a false positive or a false negative. This is probably because SALES is the most common class occurring in the data. SALES is an intent having the meaning of expressing interest in a product, before a purchase. Many of the intents that it is confused with, such as PRICING and SHIPPING_TIMES, have the same meaning, but they have additional, intent-specific meanings as well. For example, PRICING has the meaning of wanting to know the price of a product while SHIPPING_TIMES has the meaning of wanting to know when a product can be delivered. Confusions arise for the classifier if one chat turn contains some phrases supporting the general SALES intent (want to buy) and other phrases supporting a more specific intent (how much for one?). The most prolific confusion pair, reference SALES and prediction NONE, happens often when the chat turn contains noise, such as misspellings like by instead of buy.

4.1 Underspecified Output and Its Impact on Automation

We are most interested in the models producing an accurate and unambiguous joint classification label. On the other hand, because the joint classification accuracy is only 84.86%, it is also interesting to explore if we can trade off some ambiguity for additional accuracy. In order to obtain n-best joint label results, we first define the probability of a joint label $P(tn, sv, en)$ as the product of probabilities of its constituent labels: $P(tn, sv, en) = P(tn)P(sv)P(en)$

Then, the n-best joint labels are the n joint labels with the n highest probabilities, out of the n^3 possible combinations of each of the n-best labels from the three classifiers.The accuracy versus automation curves for n-best joint labels are shown in Figure 6. The biggest improvement in accuracy comes from moving from 1-best to 2-best results, with accuracy increasing from 84.8% to 92.7% at 100% automation. If we move to 4-best results, we get 95.9% accuracy at 100% automation.

We analyze the 2-best joint labeling results further by breaking down the joint labels into intents, entities, and conversation handlers. We can divide the test instances into different types (and their percentages) based on the kinds of differences be-

Correct Intent	Predicted Intent	% of All Errors
SALES	NONE	5.66
NONE	SALES	5.62
PRICING	SALES	4.31
SHIPPING_TIMES	SALES	3.79
DISCOUNTS_PURCHASE	SALES	3.48
SALES	FOREIGN_PURCHASE	3.29
TECH_ASSISTANCE	SALES	3.15
CHECKOUT_PROBLEM	SALES	3.08

Table 2: Top Confusion Pairs

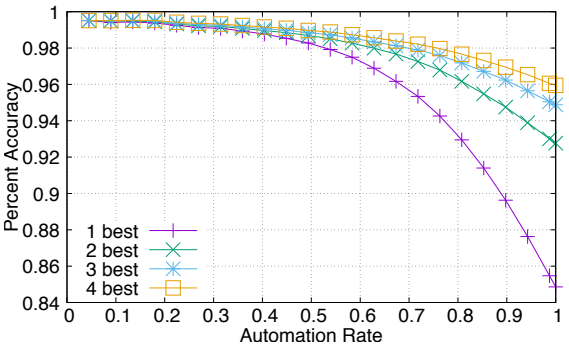

Figure 6: Accuracy vs. Automation for Different N-best Results

tween the 1-best and the 2-best results: (1) Only the intent labels are different (53.6%); (2) Only the entity labels are different (40.4%); (3) Only the conversational handler labels are different (6%) (4) More than two label types are different (1%).

Based on the encouraging analysis of the 2-best joint labeling results, we devise a *N-best Backoff (NBB)* system that uses the 2-best results that more accurately detects user intent. The major difference in NBB is that we allow the system to output *partial* answers if the 1-best answer is low confidence.

Given an input chat utterance, the 2-best joint labels are determined as previously outlined, along with a confidence score. If the score is above a predefined threshold, the system outputs the 1-best joint label. Otherwise, an algorithm *new_joint_label*, synthesizes a new joint label that takes into account the 2-best joint labels and the intent ontology.

The algorithm *new_joint_label* returns its input intent if the 1-best and 2-best intents are the same. The same is true for entities. If the 1-best and 2-best intents differ, then the algorithm checks if one is the parent of the other in the intent ontology. If so, it returns the more general, parent intent. If the 1-best and 2-best entities differ, then no entity label is produced in our current chat system. However, if we had access to a entity ontology, a underspecification similar to the one for intents could be performed on entity labels as well. For conversational handlers, the algorithm simply returns the 1-best label. Therefore, currently, the algorithm focuses mainly on synthesizing a new intent, a focus that is justified by the fact that most of the time, it is the 1-best and 2-best intents that differ.

We evaluate NBB using two measures. The first measure is the *percent error* of the joint label returned by the system. If the system returns an entire label, i.e. none of intent, entity, or conversational handlers is $\langle empty \rangle$, then it is evaluated fully. If the system returns a partial label, then only the parts of the label that have been specified are evaluated. For example, if the system returns intent as *SALES* and entity as $\langle empty \rangle$, then only the intent is checked for correctness. The second measure is *partial miss percentage*. This is the percent of utterances for which the returned joint label is partially specified, i.e. the intent or entity of the joint label is $\langle empty \rangle$, or the intent of the joint label is a more general intent (e.g. *SALES*) than the reference intent (e.g. *TECH_ASSISTANCE*). This measure is meant to capture the percent of time that the user may experience a longer dialog because the system did not fully understand the users intent.

The evaluation results of NBB are shown in Figure 7. The percent on the Y axis is percent error or partial miss percentage, depending on the curve. The automation rate on the X axis is the percent of user utterances for which the classifier returns any result, even a partial result. A partial result is considered fully automated because the system does not rely on a human intent analyst to disambiguate

the partial result. Instead, the system asks the user a clarification question.

These results are quite promising. It shows that at a 96% automation rate, the system can output intents with only 5% error. According to the nature of the algorithm, these intents may be partially specified. Still, the results show that at this automation rate, this will occur only 15% of the time, i.e. only 15% of the time will the system have to ask a clarification question in order to ask the user to repeat information that the system could not understand. The percent error can be driven below 5% as automation rate decreases, but in order to do so, the partial miss percentage increases approximately linearly.

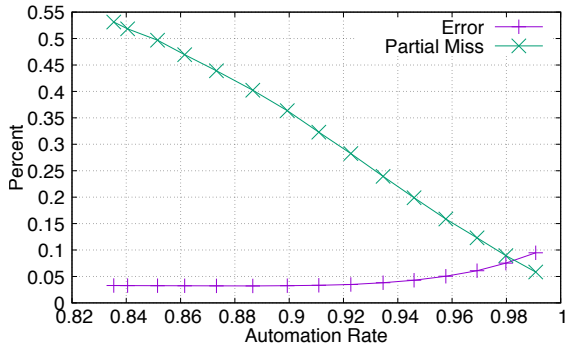

Figure 7: Percent Error/Partial Miss vs. Automation Rate

5 Related Work

Many dialog act ontologies are multidimensional and hierarchical, such that each utterance is tagged with one label from each dimension. DAMSL (Allen and Core, 1997) and its variants for different domains (Di Eugenio et al., 1998; Jurafsky et al., 1997; Dhillon et al., 2004) has four dimensions each of which has its own hierarchy. Utterances may receive more than one label, with each label DIAML (Bunt et al., 2010) has nine dimensions many of which are associated with their own hierarchies. Typically, these ontologies make a distinction between the discourse function of an utterance and its semantic content, the former being how the utterance guides the discourse and the latter concerning the topic under discussion. In our ontology, the former is represented as our conversational handlers. The latter is represented as our intents and entities. Conversational handlers are a flat set of classes rather than the rich hierarchies found in the literature. Conversely, intents and entities are relatively elaborate. These differences are rooted in the subtleties of our application which requires gathering quite specific information about the customer's requirements that can only be represented through a detailed ontology for semantic content.

Originally, work on dialog act classification largely involved processing of spoken input (Stolcke et al., 2000). Recently, it has expanded to processing of other modalities including email threads (Omuya et al., 2013), message board discussions (Kim et al., 2006), and tweets (Zhang et al., 2012). Still, most of these share in common prediction of dialog acts from a flat set of possibilities. One exception is the work of (Allen et al., 1996) where a symbolic parser predicts dialog acts from a hierarchical set. Unlike our work, Allen (1996) predicts a general rather than a specific dialog act only if the input is genuinely non-specific. Another exception is the work of (Kang et al., 2013) where a SVM classifier predicts hierarchical dialog acts. It differs from our work because Kang (2013)'s system always predicts a general dialog act and a specific dialog act for the same input, the rationale being that specific dialog acts are more accurately predicted if the general dialog act is predicted first.

6 Conclusions

In this paper, we have discussed the NLU component of an enterprise-grade, chat-based routing application. We have analyzed the possible error reduction if the NLU were to output n-best labels. Based on the analysis, we have designed and implemented an underspecification algorithm and demonstrated the significant improvement in intent accuracy an automation of a system using underspecified intents derived from an intent ontology.

References

James Allen and Mark Core. 1997. Damsl: Dialogue act markup in several layers (draft 2.1). Technical report, Multiparty Discourse Group, Discourse Resource Initiative.

James F Allen, Bradford W Miller Bradford, Eric K Ringger, and Teresa Sikorski. 1996. A robust system for natural spoken dialogue. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pages 62–70.

Harry Bunt, Jan Alexandersson, Jean Carletta, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Kiyong Lee, Volha Petukhova, Andrei Popescu-Belis, Laurent Romary, et al. 2010. Towards an iso standard for dialogue act annotation. In *Seventh conference on International Language Resources and Evaluation (LREC'10)*.

Rajdip Dhillon, Sonali Bhagat, Hannah Carvey, and Elizabeth Shriberg. 2004. Meeting recorder project: Dialog act labeling guide. Technical report, International Computer Science Institute, Berkeley.

Barbara Di Eugenio, Pamela W. Jordan, and Liina Pylkkänen. 1998. The coconut project: Dialogue annotation manual. Technical report, University of Pittsburgh.

Daniel Jurafsky, Elizabeth Shriberg, and Debra Biasca. 1997. Switchboard swbd-damsl shallow discourse function annotation (coders manual). Technical report, University of Colorado, Institute of Cognitive Science.

Sangwoo Kang, Youngjoong Ko, and Jungyun Seo. 2013. Hierarchical speech-act classification for discourse analysis. *Pattern Recognition Letters* 34(10):1119–1124.

Jihie Kim, Grace Chern, Donghui Feng, Erin Shaw, and Eduard Hovy. 2006. Mining and assessing discussions on the web through speech act analysis. In *Proceedings of the Workshop on Web Content Mining with Human Language Technologies at the 5th International Semantic Web Conference*.

Adinoyi Omuya, Vinodkumar Prabhakaran, and Owen Rambow. 2013. Improving the quality of minority class identification in dialog act tagging. In *Proceedings of NAACL-HLT 2013*. pages 802–807.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics* 26(3):339–373.

Renxian Zhang, Dehong Gao, and Wenjie Li. 2012. Towards scalable speech act recognition in twitter: Tackling insufficient training data. In *Proceedings of the Workshop on Semantic Analysis in Social Media*. Association for Computational Linguistics, pages 18–27.