Automated Conversation Review to Surface Virtual Assistant Misunderstandings: Reducing Cost and Increasing Privacy

Ian Beaver
Verint - Next IT
Spokane Valley, WA USA
ian.beaver@verint.com

Abdullah Mueen
Department of Computer Science
University of New Mexico, USA
mueen@unm.edu

Abstract

With the rise of Intelligent Virtual Assistants (IVAs), there is a necessary rise in human effort to identify conversations containing misunderstood user inputs. These conversations uncover error in natural language understanding and help prioritize and expedite improvements to the IVA. As human reviewer time is valuable and manual analysis is time consuming, prioritizing the conversations where misunderstanding has likely occurred reduces costs and speeds improvement. In addition, less conversations reviewed by humans mean less user data is exposed, increasing privacy. We present a scalable system for automated conversation review that can identify potential miscommunications. Our system provides IVA designers with suggested actions to fix errors in IVA understanding, prioritizes areas of language model repair, and automates the review of conversations where desired.

Verint - Next IT builds IVAs on behalf of other companies and organizations, and therefore analyzes large volumes of conversational data. Our review system has been in production for over three years and saves our company roughly $1.5 million in annotation costs yearly, as well as shortened the refinement cycle of production IVAs. In this paper, the system design is discussed and performance in identifying errors in IVA understanding is compared to that of human reviewers.

Introduction

Intelligent Virtual Assistants (IVAs) such as Amazon's Alexa or Apple's Siri along with specialized agents for customer service and sales support are exploding in popularity (Ram et al. 2018). The continued adoption of IVAs is contributing to a growing problem. How do we refine an IVA's knowledge effectively and efficiently? As IVA use as well as the number of tasks an IVA is expected to perform increases, there is a corresponding jump in the number of human-computer interactions to be reviewed for quality assurance. Therefore, discovering a means to expedite review and analysis of these interactions is critical.

Without scalable and efficient methods of automated conversation review, IVA designers must rely solely on human reviewers to validate expected behavior of the IVAs. As this is a manual and time consuming process, the reviewers are only able to view a limited number of interactions. The result is also subjective since reviewers may disagree on the user intention for any given turn in a conversation. In addition, as the IVA improves, errors in communication appear less often in a random sample due to their dwindling numbers. A recent challenge is public outcry over the human review of IVA conversation logs for the purpose of language understanding verification, due to privacy concerns. By the use of an automated system for conversation review, problematic interactions can still be surfaced without exposing the entire set of logs to human reviewers, minimizing privacy invasion.

In this paper we discuss a scalable system to process all conversations and autonomously mark the interactions where the IVA is misunderstanding the user. Our system provides cost savings to companies deploying IVAs by reducing the time human reviewers spend looking at conversations with no misunderstandings present. It also enables a shorter refinement cycle as problems are surfaced quickly and more reliably than a random sample or confidence metric based review. The core of our system was originally published in (Beaver and Freeman 2016; Beaver 2018) and has been in production as a commercial application for over three years. Initially created as an application internal to our company, recently we have made the system available to external partners due to its success at reducing annotation costs.

Background

Common to all IVAs is a Natural Langauge Understanding (NLU) component (Ram et al. 2018). The NLU maps user inputs, or conversational *turns*, to a derived semantic representation commonly known as the *intent*, an interpretation of a statement or question that allows one to formulate the 'best' response. The collection of syntax, semantics, and grammar rules that defines how input language maps to an intent within the NLU is referred to by us as a *language model*. The language model may be trained through machine learning methods or manually constructed by human experts (Zhao and Eskenazi 2016). Manually constructed symbolic models requires humans to observe and formalize these language rules while machine-learned models use algorithms to observe and approximate them.

Regardless of implementation details, to improve the language models and for quality assurance, human-computer interactions need to be continuously reviewed. Improvements include the addition of vocabulary and new rules

or the revision of existing rules that led to incorrect mappings within the language model. For machine-learned models, identification of incorrect understanding can highlight confusion within the model and prioritize areas of further training. The main focus of misunderstanding detection is on intent classification. It is in the NLU component that the breakdown of communication will begin, assuming adequate Automatic Speech Recognition (ASR), if speech is used as an interface. The detection of ASR error and recovery is well covered in literature (Ogawa and Hori 2015; Kim, Ryu, and Lee 2016) and outside the scope of this work.

Existing IVA Refinement Processes

IVAs for customer service are deployed in a specific language domain such as transportation, insurance, product support, or finance. *Reviewers* are given a sample of recent conversations collected from a live IVA for quality assurance. The reviewers need to be familiar with any domain specific terminology. This poses difficulty in the utilization of crowd-sourced platforms such as Figure Eight[1] or Mechanical Turk[2] as workers must be vetted to ensure they have proper knowledge of the domain and associated terminology. One strategy is to create a series of tests that workers must pass before accessing the task. Another strategy injects conversations with known labels to the sample and removes reviewers that score poorly on them.

The sample to be reviewed can be selected in a variety of ways. If a particular event is important to analyze, such a user requesting an escalation to a human, all conversations containing the event are selected. Such samples are biased and may miss many other important failure scenarios, so for a more holistic view a random sample can be taken. Another strategy selects interactions where the NLU and/or ASR confidence score is lower than some predetermined threshold. In this case, reviewers rely on the system itself to indicate where error lies. While low confidence is potentially more effective than a random sample at finding poor interactions, this requires trusting the very system that is being evaluated for errors. This also creates a dependency on the underlying system implementation that makes it difficult to compare the performance of different IVAs, or, if the system design is ever modified, the same IVA over time.

Sampled conversations are manually graded in an effort to find intents which need improvement. The reviewers may use various grading means, but a star rating system such as one-to-five stars is common (Kuligowska 2015). The result of this review process is a set of conversations along with their grades which are passed to *domain experts*. Domain experts are typically trained in NLP and are responsible for the construction and modification of language models. Only poorly graded conversations require in-depth analysis by domain experts to determine the necessary changes to the language models. The faster this review-correction cycle completes, the more quickly the IVA can adapt to changes in domain language or product or website changes that require additional knowledge.

[1]https://www.figure-eight.com

[2]https://www.mturk.com

Related Works

The QA^{RT} system presented in (Roy et al. 2016) monitors live customer service dialogs and provides supervisors with visualizations and summaries of ongoing chats. It employed features in the categories of customer behavior (emotion and sentiment), conversational characteristics (deviation from typical structure, number of turns, average delays), and organizational compliance (greeted customer, used customer name, assurance, etc.). As the QA^{RT} system is monitoring human-human chats there is no concept of intents nor does it directly detect misunderstanding. However, change in sentiment and emotion proved useful for indicating misunderstanding occurred and was implemented in our system.

In (Jiang et al. 2015), a model to predict intent classification quality of an IVA using numerous ASR, dialog, and tactile features is given. Users were asked to complete tasks with the IVA and then were given a survey to rate their satisfaction with the experience, the quality of speech recognition, and the quality of intent understanding. Authors then compared sequences of user actions to request and response features. The authors rely on the user rating to determine the intent classification accuracy which can be biased by the IVA response. Poor response wording can appear to the user as a misunderstanding when, in reality, the NLU component understood the intent but the generated response was inaccurate. Regardless, features correlated to intent classification errors, such as turn similarity and repeated responses, have been incorporated into our system.

System Design and Components

The core of our system is a learned model of features for predicting intent classification errors in conversational turns. This model is used to generate a score per turn representing the risk of intent misclassification. This *risk score* is used to rank turns for priority review where humans vote on if each turn was misunderstood. The system can also vote if the turn was misunderstood to reduce or eliminate the need for human voting. The voting outcomes generate suggested actions to fix the human-identified errors in the language model. The commercial name of this system is Trace AI.

Trace is designed with three primary functions. The first is detecting features of intent error and aggregating these features into a risk score. The risk analysis engine applies various heuristics and classifiers to detect indications of misunderstanding in each conversational turn and score them between $[0, 1]$. As each indication is independently scored, and each conversation is independent, this task is done in parallel on a compute cluster for scalability.

Once each turn is annotated with all applicable risk indicators, the risk score for a particular turn is calculated as the weighted sum of all indicator scores in that turn. Weights are initialized to 0.5 and tuned over time using odds ratios. The odds ratio represents the odds that an outcome will occur given a particular exposure, compared to the odds of the outcome occurring in the absence of that exposure (Szumilas 2010). As reviewers grade turn-intent mappings, the odds of each risk indicator predicting a misunderstanding is recalculated and the weight of that risk indicator is adjusted,

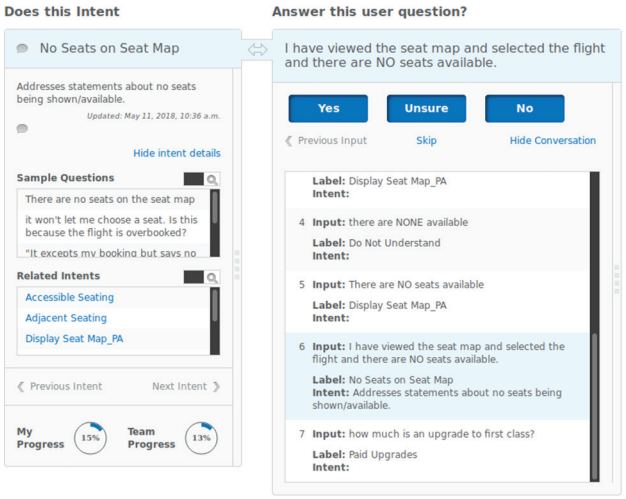

Figure 1: The Trace voting interface used by reviewers

Circumstance	Recommended Action
A turn-to-intent map is voted to be correct	None. These are added as training and regression samples for Trace.
A turn-to-intent map is voted to be incorrect	Fix or retrain the language model to prevent the turn from reaching the associated intent.
The reviewer majority votes *Not Sure*	Determine if the intent was appropriate for the turn or if a new intent should be created.
There is no reviewer consensus	Determine if the intent was appropriate for the turn or if a new intent should be created.
Voters are conflicted as they approved the turn in more than one intent	Clarify definitions of both intents and re-release for voting.

Table 1: Voting outcomes and recommended actions

improving the scoring model. As indicators may be domain-dependent, weights are adjusted per domain.

The other two functions of Trace are provided by a Django web application with interfaces for two types of users. The first type are the domain experts who create projects, linked to a live IVA, and select a time range over which to do analysis. Once they have defined a project and a time range for review, Trace prioritizes all conversational turns within that range by their risk score. The second type are the human reviewers whose work flow involves logging into a project (a collection of conversations from a live IVA) and reviewing turns that have been prioritized by risk score. They read each turn in the context of the conversation and vote on whether or not they agree with the intent chosen by the NLU in the live IVA. If a reviewer does not feel they have enough information or domain knowledge to decide, they may also vote **Unsure**. A turn is shown to a minimum of three reviewers to reduce subjectivity and provide domain experts with a sense of inter-reviewer agreement.

Trace is implemented entirely in Python and deployed as four components on Amazon Web Services[3]. The components are a t3.xlarge webserver, two r5.2xlarge Celery nodes, three r5.2xlarge MongoDB nodes, and six m4.10xlarge Slurm[4] HPC nodes. This single deployment is sufficient to support analysis on the 40+ live IVAs we currently maintain.

The Reviewer Interface

A screen shot of this interface is shown in Figure 1. In the left-hand column the intent the reviewer is currently voting on is displayed along with additional information to give insight. The label of the intent is displayed at the top, followed by a text description of its purpose, which is maintained by the domain experts. If the reviewers do not fully understand the purpose of an intent, they can submit questions to the domain experts by clicking on the comment bubble below the description text. The experts can then update the description to clarify the purpose of the intent so that voting is accurate.

Next, a set of sample questions that have been previously human-validated to belong to this intent are displayed. This is to give the reviewer some intuition on the language intended for the current intent. Following that is a list of related intents to help the reviewer decide if a more suitable intent exists in the language model. Both lists are searchable to speed analysis. Finally, controls to navigate through the intents to be reviewed and, at the bottom, metrics on how many turns have been completed by the current reviewer and all reviewers combined on the displayed intent are shown.

On the right-hand side the user turn is shown followed by voting buttons. Keyboard shortcuts are provided to speed voting. The entire conversation with the current turn highlighted is displayed to give the reviewer the conversational context needed to determine if the responding intent was appropriate. Notice that nowhere does the actual response *text* from the IVA appear. The response is not shown in order to separate the evaluation of the NLU component from that of the natural language generation (NLG) component. Recall that in this work we are primarily interested in the evaluation and improvement of the language model, therefore this isolation is necessary. Once it has been established that the NLU is performing acceptably the NLG can be evaluated separately, which is outside the scope of Trace.

The Analysis Interface

After the risk analysis and voting processes are complete, Trace provides voting data and additional recommendations to the domain experts to facilitate language model development. To optimize domain experts' time, Trace uses the reviewer voting outcomes to determine a recommended action per turn, shown in Table 1 and visualized in Figure 2. These actions help the domain experts quickly determine what to do with the voting results for a particular turn.

To prioritize language model repair work by the impact it will have on the live IVA, the set of turns and their voting

[3]https://aws.amazon.com/ec2/instance-types
[4]https://slurm.schedmd.com/

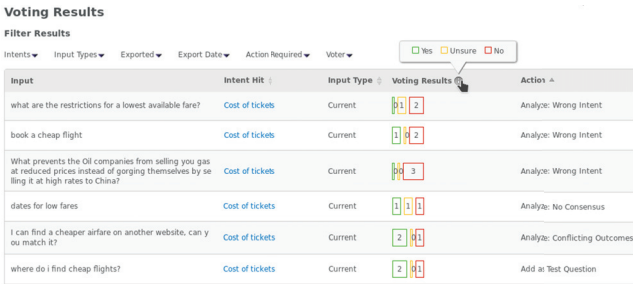

Figure 2: Analysis interface within Trace used by domain experts to view voting results and reviewer agreement.

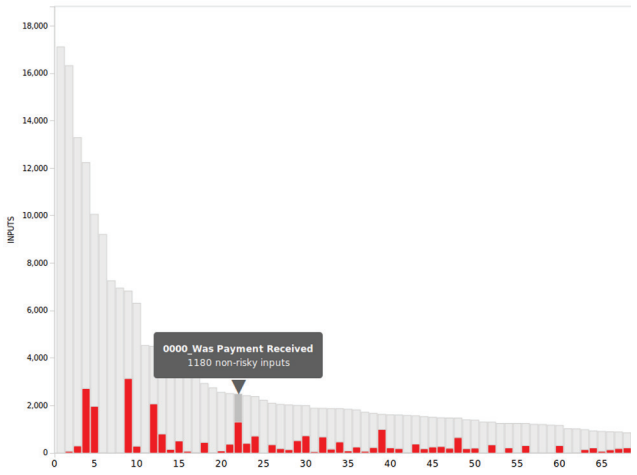

Figure 3: Trace presents the ratio of misunderstood to correct inputs per intent to prioritize work. The red bar is the count of misunderstood inputs assigned to that intent in the live IVA. The grey bar is the count of correct inputs.

outcomes are first grouped by responding intent and then ordered by the frequency of response within the conversation logs. A screen shot of this prioritization from the analysis interface is shown in Figure 3. By looking at this chart, domain experts can quickly determine which malfunctioning intents have a greater impact on user experience. If two intents have a similar ratio of misunderstood inputs, the intent with the higher response frequency would be prioritized for repair as its malfunction will have a larger impact on overall user experience.

To help domain experts quickly analyze the voting results and voter consensus the analysis interface provides the tabular view shown in Figure 2. The filters at the top provide the ability to explore the results from many angles such as per intent, per voter, date range, recommended action, etc. In the left hand column the original user turn text is displayed. In the next column is the intent that the reviewers evaluated the text against. The "Input Type" column shows whether the intent evaluated was from the current NLU or a different source, such as regression tests used in developing the language model or live chat logs. Trace is designed in such a way that it can perform misunderstanding analysis on any textual data labeled with intent or topic. The "Voting Results" column provides a visual indicator of the voting outcome and inter-reviewer agreement. The final column on the right hand side is the recommended action from Table 1. Filtering this table by an action type will quickly surface all turns where a particular action should be performed.

From this view the domain experts can quickly find areas of the language model that need attention and export the text data with any detected risk indicators and voting results. They can then use this data along with the NLU-specific means to make the necessary changes in the language model.

Indicators of Intent Error

This section describes the individual indicators of intent error that the risk analysis engine tests for. These were derived from literature review on miscommunication in conversation combined with our own emperical evidence from 18 years of IVA development. Further discussion on these indicators and how they are detected was presented in (Beaver 2018).

Conversation Level Features

The following features apply risk equally across all turns within the single conversation where they are present. These features are used to detect miscommunication over the course of the conversation and elevate the risk score for turns in conversations where miscommunication was likely to have occurred.

I Don't Know (IDK) in conversation
An IDK occurs when the language model does not find an intent that satisfies the user query with a high enough confidence. The IVA may respond with something like "I'm sorry, I didn't understand you." If a conversation contains one or more IDK responses, this may indicate that the user is talking about some subject the IVA has no knowledge of.

same intent(s) hit
The same intent is hit more than once within the conversation. This is an indication of risk within a customer service conversation because it is unlikely the user would want to see the same response repeated. If these are successive in a conversation they are considered to be **sequential hits**. This usually indicates that the response to the first input did not satisfy the user; he or she is rewording the question to get a different response. If the system has the initiative, this may mean that the system is repeating a prompt, a common indication of miscommunication (Aberdeen and Ferro 2003).

tie in conversation
The responding intent for one or more turns in the conversation had a nearly identical score as one or more different intents. This indicates confusion in the NLU around the input language for the tying intents. If a conversation contains such ties it may surface subject matter that is not well defined in the language model.

user rating scores
Users may be asked for feedback on how helpful the IVA was for their session. However, feedback is not entirely reliable as we have observed users who give negative feedback if the IVA rightly upholds business rules. For example,

business rules may prevent transferring a ticket to a different passenger, and, when a user attempts to do so, the IVA will not let them. In retribution the user grades the conversation poorly but the IVA was not at fault. The user may also say the IVA was unhelpful when the NLU was indeed working correctly, but the response text was poorly worded. Therefore this feedback is only a measure of risk in our system and not the final determination as in (Jiang et al. 2015).

conversation should escalate
An escalation occurs when a user requests an alternative channel for the completion of a task. Whether or not there was explicit user request for an escalation in the conversation, an algorithm (Freeman and Beaver 2017) has determined that the conversation *should* have been escalated due to IVA failures to complete the task at hand.

sentiment change over time
The user began the conversation with positive or neutral sentiment, but by the end of the conversation their sentiment was negative. This may be caused by either the IVA preventing them from completing a task due to business rules, or due to IVA misunderstanding.

Turn Level Features

The following features only apply risk to a single turn. However, they may still use features of the conversational context in their determination.

triggers IDK response
If the response to this turn is an IDK, this may indicate that the user has asked about a subject the IVA does not have knowledge of.

contains backstory
Users may give backstory on their task that is unnecessary for determining the correct intent. The presence of this language can add confusion in the NLU and result in an intent error (Beaver, Freeman, and Mueen 2020). For example, a user may tell the IVA that he or she needs to fly to Boston for a son's graduation party. The fact that the user has a son and is attending his graduation party is irrelevant to the task. The additional language can interfere with determining the user's primary task of booking a flight. We apply (Kim, Ryu, and Lee 2016) to segment intents in the text, and if the NLU is unable to determine the intent of a segment, we consider it the presence of out-of-domain/unnecessary language.

precedes corrections
The following user turn contains error correction language, such as *"no, ..", "I said ..", ".. not what I .."* (Bulyko et al. 2005; Freeman and Beaver 2017).

abandonment
The user left the conversation immediately after the IVA asked them a question. This indicates that the IVA did not have all the information it needed to complete the task, but the user abandonment indicates it was likely trying to accomplish the *wrong* task and the user left in frustration.

contains multiple intents
If multiple intents are present it can add confusion to the NLU. We assume the IVA under review does not support multiple intents within a single turn as multi-intent parsing is still an unsolved problem for IVAs (Khatri et al. 2018). Using the method given in (Kim, Ryu, and Lee 2016), we detect if multiple intents are present in the user turn.

triggers sequential hit or impasse
The turn hit the same intent as the previous turn. This usually indicates that the previous response did not satisfy the user, so he or she is rewording the question to get a different response but failed to do so. An **impasse** occurs when the same intent is returned more than two times in a row. In which case the IVA may respond with something like *"I think you are asking for more information than I have."*

precedes escalation
As escalations may be due to previous IVA failures, risk is assigned to the turn preceding any escalation request.

precedes unhelpful
The input directly preceded a turn stating the unhelpfulness of the IVA. This is a common reaction when the user is frustrated at the inability to make progress in their task.

precedes profanity
The input directly preceded an interaction containing profanity. With a customer service or product support IVA, profanity is usually a sign of user frustration or irritation.

precedes negative sentiment
If a turn contains negative sentiment, this may be due to the user's reaction to the previous IVA response. Therefore, risk is assigned to the preceding user turn.

restated
If a turn is very similar to one or more following turns, this may indicate the user was dissatisfied with the response and rewords the question. Similarity is defined as a rephrasing of the same question or statement as measured by cosine similarity of sentence vectors; it may not have triggered the same intent in the IVA (Jiang et al. 2015).

precedes IDK
We have observed that IDKs may follow misunderstood turns. This type of IDK can happen when the user reacts in surprise or frustration (*"What??"*) or changes the subject to complaining about the IVA due to the misunderstanding (*"This is the dumbest thing I have ever used!"*).

triggers tie
The responding intent had a nearly identical score as one or more different intents. This indicates confusion in the language model around the input language.

contains unknown words
The user turn contains words that are out of vocabulary for the underlying language model. This may indicate that the user is talking about some subject the IVA does not have knowledge of.

should escalate point
There was no explicit user request for escalation in the conversation, but an algorithm (Freeman and Beaver 2017) determined that the conversation *should* have escalated at this point in the conversation due to task failures.

Dataset	# Conv	Total User Turns	Textual User Turns	Majority Agreement
Train	2,030	13,930	7,270	6,331
Telecom	1,342	20,485	7,313	5,252
Airline	1,611	9,103	9,103	6,978
Average	1,661	14,506	7,895	6,187

Table 2: Dataset statistics for the evaluation data.

Evaluation

The purpose of Trace is to reduce the human burden and costs in maintaining conversational agents. To demonstrate its utility, we measure it's performance in automating the reviewer voting process on real datasets from three live IVAs as well as performing a cost analysis of human review.

Data

Due to annotation budget for this study, we limited our average user turns per dataset to 8,000. All turns in a conversation need to be reviewed. However, conversations have varying numbers of turns and, with multi-modal IVAs, not all user turns consist of natural language. For example, some user turns in a conversation may be events such as user interface clicks or web page navigations which the IVA responds to. Using the average natural language turns per conversation we estimated the sample size per domain. We then selected a random sample of full conversations, using the estimated sample size per domain, from the conversation logs of a live virtual agent we maintain in each domain.

All natural language turns were selected for voting and released to a group of 14 voters. Three votes per turn was required to control for subjectivity. Voters were all employees of Verint - Next IT who were trained on the Trace user interface and voting process prior to actually voting, and many were domain experts. After voting, the average number of turns per dataset with a clear majority (agree or disagree with the intent chosen by the live IVA) was $6,187$. If there was no clear majority, the turn was not used for evaluation. Although all three datasets had 14 voters, not all 14 were the same people; there were 17 unique voters overall.

Evaluation dataset statistics are given in Table 2. Total User Turns involve all forms of user input including clicking on controls and web page navigation events. Textual User Turns are only those that were processed by the NLU component for intent classification. As we are only interested in the discovery of error in the NLU, it is these user turns that are evaluated by humans. Majority Agreement are the number of Textual User Turns where a majority (at least 2) of the three voters agreed. Note that voters can choose **Not Sure** (see Figure 1) so a majority is not guaranteed.

From these counts we can see that the Telecommunications IVA is very interactive, less than half of user turns are actually in the form of natural language. This IVA responds to many user activities besides typed or spoken input. In contrast, the Airline IVA does not accept anything but typed or spoken input. The Train IVA appears a good balance of the two interaction styles. The Train IVA had the highest level of overall voter agreement, at 87%. The Airline had less at 76.7% followed by the Telecommunications IVA with 71.8% agreement. Inspecting the conversations and IVA knowledge bases, it appears these differences are due to the complexity of the IVA and the number of intents understood. The Train IVA has 930 distinct intents in its knowledge base, compared to 1,223 for the Airline IVA and 2,173 for the Telecommunications IVA. Not surprisingly, the increase in possible intents to select from appears to decrease voter agreement on the correctness of an intent chosen by the IVA.

Comparison Metrics

Due to the multiple layers of random sampling used to create the datasets and gather the votes, fairly comparing humans to each other and Trace can be difficult. As the human voters did not see all of the user turns in a dataset, but were merely given a subset of turns ensuring each turn had three votes each, we cannot calculate a recall, and therefore a F1 score, for the humans. Furthermore, no two humans saw the exact same subset of the turns to ensure a pair of voters who only choose one value (always vote **Yes**, for example) could generate an inaccurate majority on an entire subset. Therefore, to compare the human reviewers to each other and to Trace we considered only the class unweighted (micro) and class weighted (macro) precision. The equations for both in the binary case are given where $tp =$ true positive and $fp =$ false positive:

$$P_{Micro} = \frac{tp_{\mathbf{Yes}} + tp_{\mathbf{No}}}{tp_{\mathbf{Yes}} + tp_{\mathbf{No}} + fp_{\mathbf{Yes}} + fp_{\mathbf{No}}} \quad (1)$$

$$P_{Macro} = \frac{1}{2}\left(\frac{tp_{\mathbf{Yes}}}{tp_{\mathbf{Yes}} + fp_{\mathbf{Yes}}} + \frac{tp_{\mathbf{No}}}{tp_{\mathbf{No}} + fp_{\mathbf{No}}}\right) \quad (2)$$

The micro-averaged precision gives a sense of how many "correct" votes a reviewer made over the sample size they reviewed. Equal weight is given to each turn classification decision without regard to class imbalance (Schütze, Manning, and Raghavan 2008). However, as the two classes are very imbalanced (only 14.45% are class **No** averaged over the three datasets) this can be misleading if viewed alone.

In contrast, the macro-averaged precision gives a sense of effectiveness on small classes (Schütze, Manning, and Raghavan 2008). Taking these two measurements together we can get a sense of a classifier's (human or machine) performance overall and performance equally favoring the under-represented class of misunderstandings.

Automating Reviewer Voting

Beyond prioritizing human reviewer time we wish to automate the entire voting process where possible. To do this we train a binary classifier to vote **Yes** or **No** for each <turn, intent> pair identical to the reviewer task. Humans would not be entirely replaced however, as the risk indicator weights and voting classifier would need periodic retraining to account for changes in the set of intents within the language model. In this way, Trace is a human-in-the-loop system which automates many of the human review tasks without entirely replacing the valuable human decision making.

To select the voting classifier, we performed an extensive evaluation of various classification methods on each dataset. The voting classifiers were trained using the unweighted risk indicator values as features and the majority decision as the outcome. If voters agreed that turn t belongs to the intent assigned by the IVA, the class is 1. If they disagree, the class is 0. For each turn with a voter consensus we add a row to a feature matrix M, with a column for each risk indicator and a final column for the class.

$$M = \begin{array}{c} \\ t_1 \\ t_2 \\ \vdots \\ t_n \end{array} \begin{array}{c} \begin{array}{cccc} backstory & restated & & class \end{array} \\ \begin{pmatrix} 0 & 1 & \cdots & 1 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 1 \end{pmatrix} \end{array}$$

This feature matrix M is then used to train a binary classification model using a stratified 30-fold cross validation. When a new turn is under review, the risk indicators present are represented as a vector and fed to the voting classifier to predict the majority vote of **Yes** or **No**. The classifiers were trained and evaluated on each dataset in isolation. The classification method with the highest combined precision across all three datasets and fastest training time training time was chosen. Training time and scaling are important considerations as Trace is continually retraining these models per dataset as human voting data is added. Our final selection for the voting classifier was a Random Forest model with 30 estimators, which required on average 2 seconds to train.

Voting Classifier Evaluation

Having selected a voting classifier, we compare its performance on each dataset to the human voters in Table 3. For each voter, we calculated the micro and macro precision of their votes to the majority vote. It is obvious from this table that as the IVA complexity increases, T_{Macro} compared to H_{Macro} suffers, although T_{Micro} does not seem as affected indicating the performance on class **Yes** still holds.

There is a bias favoring the humans here in that the gold standard was produced by the majority of human reviewers. The bias arises when a third reviewer votes on a user turn where there exists one **Yes** and one **No** vote. In this case the third reviewer is forming the majority either way they vote and cannot be penalized.

In light of this, the human voter precision scores given may be higher than a true outside observer predicting the existing majority vote as Trace does. It is dangerous to try to correct for this by ignoring votes that form majority however, as turns will not be scored *for* the two humans choosing the majority, but will be scored *against* the one that didn't. This gives more chances for penalty than reward. We also cannot only consider turns where all three reviewers agree as the human performance will always be perfect and turns with some disagreement are potentially harder cases we want to evaluate Trace on. Therefore, we only note that the bias exists and favors the human voters.

Annotation Cost Savings

In production, we use an annotation service (under NDA) which costs our company \$0.10 per turn. For our Telecom IVA, which responds to 1.8 million user turns per month, reviewing 5% for quality control requires 90,000 turns to be reviewed every month. Recall this is a very difficult domain in which there are over 2,000 unique intents and complex intent classification logic, and is the domain in which Trace performed poorly. At \$0.10 per turn the annotation cost for 90,000 is \$9,000 per month for just one subjective review per turn. Using crowd-source platforms such as Mechanical Turk would cost \$0.54 per turn using qualified workers[5] and paying them \$0.10 per turn, which would greatly inflate monthly annotation costs. In addition, this IVA has a historical intent error rate of 14.35%, meaning only 12,915 turns of the 90,000 actually need review, assuming the 5% is a truly random sample. We wish to minimize reviewing turns with no error as they are not used for IVA improvement directly.

Dataset	H_{Macro}	T_{Macro}	H_{Micro}	T_{Micro}
Airline	0.85±0.24	0.74	0.93±0.12	0.89
Telecom	0.82±0.24	0.59	0.89±0.16	0.86
Train	0.79±0.28	0.74	0.84±0.38	0.83

Table 3: Human (H) voter mean precision $\pm$ 95% on majority agreement compared with Trace (T) voting classifier that a turn's intent was misclassified, by domain.

Using Trace to prioritize the data for review by the risk score, as the voting classifier is not trustworthy in this domain (see Table 3), we have measured 28% of the riskiest 5% in a month to be actually misunderstood. This doubles the number of turns that truly needed review in the sample, and therefore we find a similar number of misunderstandings in a 2.5% sample from Trace as in a 5% random sample, saving 50% in monthly annotation costs for a *single* IVA.

In some domains with less complex language models, such as the Train IVA, Trace has similar performance to the average human reviewer (see Table 3). In these cases we have been able to eliminate the human voters, with the exception of periodic batches to tune the risk indicator weights and voting classifier. The annotation cost savings for these IVAs have been closer to 90%. Considering that the Telecom IVA is a worst-case for Trace and that we maintain 40+ production IVAs and growing, by introducing Trace company-wide we save nearly 75% in monthly annotation costs, or roughly \$1.5 million yearly, over random samples.

Scalability

To measure the limits of the architecture, we conducted a scaling test. We start with a single 40-core compute node enabled and feed Trace 400k turns, roughly 1 week of conversations from the live Telecom IVA, and measure the wall clock time to complete the risk analysis and apply the voting classifier to all turns. Then the entire system was restarted to clear out any caches. After restarting, an additional compute node is added to the cluster and the test is repeated.

In Figure 4, we see the results of this scaling test. With a single compute node, it takes roughly 5.5 hours to process 1

[5] https://requester.mturk.com/pricing

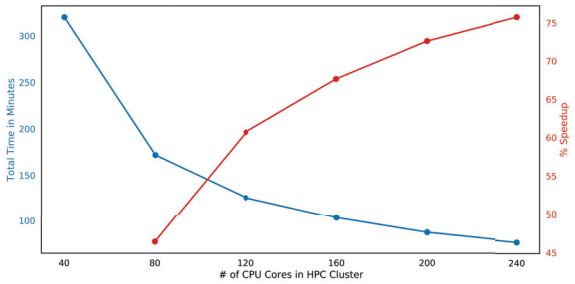

Figure 4: The performance impact on processing 400k turns increasing the compute cluster one 40-core node at a time.

week of data. With six compute nodes, Trace can process the data in roughly one hour, at which point MongoDB and the network overhead begins to bottleneck linear scaling. With six compute nodes in our production Slurm cluster, the architecture has proven capable of processing the conversation data influx of 40+ live IVAs concurrently.

Conclusion

We have presented Trace, a system to prioritize human reviewer time and reduce annotation costs associated with maintaining production IVAs at scale. In addition, by minimizing the amount of human review necessary, we reduce the amount of user data exposed through the review process. In the best case, where review can be reduced to periodic system reinforcement, the vast majority of conversations are not seen by humans, while still ensuring IVA quality.

We discussed the design of the system as it is presented to the reviewers and domain experts, and how it can help domain experts prioritize their time for language model repairs that will have the largest impact on user experience.

Trace relies greatly on previous research in human-computer interfaces, communication, and natural language processing in the development of it's indications of risk. To our knowledge there exists no other similar application for the improvement of IVAs. Trace has been used in a production capacity for over three years, processing hundreds of millions of conversational turns per year.

Trace presents voting results and actions to the domain experts through the same interface regardless if the voter was human or machine (see Figure 2) and Trace always votes on every turn. Therefore, the source of votes can be chosen based on current system performance in a particular language domain or human reviewer availability. As Trace is implemented as a web application, domain experts can easily use internal or external annotation sources for voting.

Our system uses only conversational features for misunderstanding detection and is not dependent on the implementation details of the underlying IVA or the domain of language it is deployed in. This, combined with the flexibility of annotation sources, its ability to scale to real-world volumes of data, and its proven ability to lower costs make it a beneficial application to our company, and any company that maintains enterprise IVAs or chatbots.

References

Aberdeen, J., and Ferro, L. 2003. Dialogue patterns and misunderstandings. In *ISCA Tutorial and Research Workshop on Error Handling in Spoken Dialogue Systems*.

Beaver, I., and Freeman, C. 2016. Prioritization of risky chats for intent classifier improvement. In *Florida Artificial Intelligence Research Society Conference*, volume 29, 167–172. The AAAI Press, Palo Alto, California.

Beaver, I.; Freeman, C.; and Mueen, A. 2020. Towards awareness of human relational strategies in virtual agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*. The AAAI Press, Palo Alto, California.

Beaver, I. 2018. *Automatic Conversation Review for Intelligent Virtual Assistants*. Ph.D. Dissertation, University of New Mexico.

Bulyko, I.; Kirchhoff, K.; Ostendorf, M.; and Goldberg, J. 2005. Error-correction detection and response generation in a spoken dialogue system. *Speech Communication* 45(3):271–288.

Freeman, C., and Beaver, I. 2017. Online proactive escalation in multi-modal automated assistants. In *Florida Artificial Intelligence Research Society Conference*, volume 30, 215–220. The AAAI Press, Palo Alto, California.

Jiang, J.; Hassan Awadallah, A.; Jones, R.; Ozertem, U.; Zitouni, I.; Gurunath Kulkarni, R.; and Khan, O. Z. 2015. Automatic online evaluation of intelligent assistants. In *Proceedings of the 24th International Conference on World Wide Web*, 506–516. ACM.

Khatri, C.; Hedayatnia, B.; Venkatesh, A.; Nunn, J.; Pan, Y.; Liu, Q.; Song, H.; Gottardi, A.; Kwatra, S.; Pancholi, S.; et al. 2018. Advancing the state of the art in open domain dialog systems through the Alexa prize. *arXiv preprint arXiv:1812.10757*.

Kim, B.; Ryu, S.; and Lee, G. G. 2016. Two-stage multi-intent detection for spoken language understanding. *Multimedia Tools and Applications* 1–14.

Kuligowska, K. 2015. Commercial chatbot: Performance evaluation, usability metrics and quality standards of embodied conversational agents. *Professionals Center for Business Research* 2.

Ogawa, A., and Hori, T. 2015. ASR error detection and recognition rate estimation using deep bidirectional recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, 4370–4374. IEEE.

Ram, A.; Prasad, R.; Khatri, C.; Venkatesh, A.; Gabriel, R.; Liu, Q.; Nunn, J.; Hedayatnia, B.; Cheng, M.; Nagar, A.; et al. 2018. Conversational AI: The science behind the Alexa prize. *arXiv preprint arXiv:1801.03604*.

Roy, S.; Mariappan, R.; Dandapat, S.; Srivastava, S.; Galhotra, S.; and Peddamuthu, B. 2016. QART: A system for real-time holistic quality assurance for contact center dialogues. In *AAAI*, 3768–3775.

Schütze, H.; Manning, C. D.; and Raghavan, P. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press.

Szumilas, M. 2010. Explaining odds ratios. *Journal of the Canadian Academy of Child and Adolescent Psychiatry* 19(3):227.

Zhao, T., and Eskenazi, M. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. *arXiv preprint arXiv:1606.02560*.