

ARTICLE

A randomized prospective study of a hybrid rule- and data-driven virtual patient

Adam Stiff^{1,*} , Michael White², Eric Fosler-Lussier¹, Lifeng Jin², Evan Jaffe² and Douglas Danforth³

¹Department of Computer Science and Engineering, The Ohio State University, Columbus, OH, USA, ²Department of Linguistics, The Ohio State University, Columbus, OH, USA, and ³The Ohio State University Wexner Medical Center, Columbus, OH, USA

*Corresponding author. E-mail: stiff.4@osu.edu

(Received 20 September 2019; revised 5 August 2022; accepted 8 August 2022; first published online 23 September 2022)

Abstract

Randomized prospective studies represent the gold standard for experimental design. In this paper, we present a randomized prospective study to validate the benefits of combining rule-based and data-driven natural language understanding methods in a virtual patient dialogue system. The system uses a rule-based pattern matching approach together with a machine learning (ML) approach in the form of a text-based convolutional neural network, combining the two methods with a simple logistic regression model to choose between their predictions for each dialogue turn. In an earlier, retrospective study, the hybrid system yielded a nearly 50% error reduction on our initial data, in part due to the differential performance between the two methods as a function of label frequency. Given these gains, and considering that our hybrid approach is unique among virtual patient systems, we compare the hybrid system to the rule-based system by itself in a randomized prospective study. We evaluate 110 unique medical student subjects interacting with the system over 5,296 conversation turns, to verify whether similar gains are observed in a deployed system. This prospective study broadly confirms the findings from the earlier one but also highlights important deficits in our training data. The hybrid approach still improves over either rule-based or ML approaches individually, even handling unseen classes with some success. However, we observe that live subjects ask more out-of-scope questions than expected. To better handle such questions, we investigate several modifications to the system combination component. These show significant overall accuracy improvements and modest F1 improvements on out-of-scope queries in an offline evaluation. We provide further analysis to characterize the difficulty of the out-of-scope problem that we have identified, as well as to suggest future improvements over the baseline we establish here.

Keywords: Text classification; Machine Learning; Dialogue; Natural Language Interaction

1. Introduction

Validation studies of prior work are important for identifying and overcoming potential sources of bias in the initial studies, revealing unexpected issues, and generally confirming that an engineered system behaves as expected. Randomized prospective trials are generally regarded as the gold standard of study design, whether for validation or not, since they inherently eliminate many biases that can arise from other designs. We previously developed a virtual patient question-answering dialogue agent for training medical students to interview patients (Jin *et al.* 2017), which is the target of our validation efforts in this work. The educational purpose of the application demands a high degree of robustness to a wide range of interactions, because the intended audience are



“novice experts.” In other words, the users are already highly trained, but still learning the specific task for which our agent is designed, which means the agent must gracefully handle potentially irrelevant questions that would not be asked by an experienced professional. Our previous work developed a novel hybrid rule- and data-driven natural language understanding (NLU) component, which we evaluated retrospectively against a relatively small corpus of collected dialogues. Remarkably, that evaluation identified a nearly 50% error reduction compared to the rule-based system alone, so we aim to validate the hybrid design under the more rigorous prospective standard, by deploying it into a live setting with the intended audience. Fortuitously, this validation work highlights a number of interesting issues in the design of dialogue systems generally, particularly with respect to the management of out-of-scope queries and to iterative processes of dialogue system design.

Paid actors—called *standardized patients*—are traditionally used to train medical students in the strategies and techniques of interviewing patients. Employing these actors for training has a few drawbacks. Perhaps the most obvious are cost and convenience: actors must be paid and trained, so individual student access to the patient actors is necessarily limited and must be carefully coordinated across a cohort of students. More importantly, consistency may often be lacking across live actors or over time for a single actor, who may become bored or fatigued. This can lead to interactions which are less instructive for some students, for example, a patient providing answers to questions that were not asked, to speed up the interview. More generally, simple human error and variability can produce substantially different experiences for different students, which is usually not ideal in instructional situations. Finally, in order for students to receive feedback on their performance, the interview must be graded by a medical educator, whose time is necessarily limited. This can introduce delays from when the interview is performed, which is when feedback would be most valuable for learning.

Our virtual patient (i.e., a chatbot with a graphical avatar, see Figure 1) is intended to address all of these issues. Once developed, a single virtual patient with a particular case history can be deployed at a low cost, with constant, concurrent availability through a chosen electronic interface. This low cost also means that the students can practice with more scenarios and repeat interactions, which is prohibitively expensive with standardized patient actors. The patient can be programmed to give consistent responses to equivalent questions, and feedback about topic coverage and other aspects of student performance can be provided immediately following the interview. Of course, programming a virtual patient introduces significant technical challenges that paying an actor does not. The graphical interface must be believable enough not to distract from the educational goals of the patient interaction. Ideally, it will also retain features of non-linguistic communication that are important for doctors to respond to, including expressions of pain, anxiety, etc. Among the most significant challenges, and the main concern of the present work, is the ability of the virtual patient to correctly identify the natural language question being asked by the student. This not only directly impacts the ability of the chatbot to correctly answer the question, thus informing further questioning, but also affects the ability to rapidly evaluate the student’s identification and coverage of topics that are relevant to the patient’s condition.

A number of decisions were made in the design of our agent to balance the tractability of implementation with the agent’s ability to realize the educational objective. An exhaustive enumeration of those design decisions is beyond the scope of this paper, but we list a few of the most important ones over the remainder of this section, for the sake of introducing the system and framing the rest of the paper. First, we focus on a single patient case: a middle-aged male presenting with acute back pain. The dialogue agent is modeled as a basic question-answering agent, which dramatically simplifies the dialogue policy. The user always has the initiative, which creates a simplified turn-taking dialogue in which the patient simply responds to the user’s questions, obviating the need for any model to decide when the system should take the initiative. Furthermore, we assume each input is independent of the dialogue history. Obviously, this is not a valid assumption in general, but careful design of the details of the patient’s case makes this approach work well without



Figure 1. The user interface of the Virtual Patient. Note the dialogue input box at the top of the screen, where a user has typed “Hi Jim, how are you?” After the user presses the Enter key, the patient will respond with “Pretty good, except for this back pain.”

sacrificing the educational objective of developing interviewing skills. As an example, we limit the patient to a single active medication, to minimize context-dependent coreferences to multiple medications. Among others, these assumptions and design simplifications allow us to formulate the challenging NLU task as a slightly simpler *question identification* task—that is, mapping a wide variety of natural language inputs into the set of hundreds of questions to which the patient is programmed to provide fixed responses. One can envision multiple approaches to this problem, including ranking inputs for matches against known queries or directly classifying inputs as one of many known classes. We adopt the latter approach.

The design of the NLU component itself is a combination of a rule-based system and a data-driven machine learning (ML) classifier, although this hybrid architecture was not chosen deliberately at the outset of the project. The earliest versions of our virtual patient (Danforth *et al.* 2013) were built with a rule-based dialogue management engine called ChatScript (Wilcox 2019) to handle the necessary NLU task. ChatScript is a pattern matching engine that automatically performs some input regularization and analysis, provides a straightforward pattern matching syntax, maintains dialogue state, can remember facts and dialogue history, and responds to input with prepared answers or further questions (Wilcox and Wilcox 2013). ChatScript-based chatbots have been successful entrants at the annual Loebner Prize (Bradeško and Mladenčić 2012), and ChatScript makes it fairly easy to author new chatbots, which is important for making the technology of dialogue systems available to other fields, such as medicine. However, rule-based approaches to dialogue agents do exhibit some drawbacks. It is something of an art form to write patterns and answers that are both adequately specific and sufficiently general to correctly match the questioner’s intent.

ChatScript provides many advanced features for interpreting input and formulating responses, but taking advantage of all of them requires ample expertise and is labor intensive. Furthermore, as the number of authored patterns increases, the potential for conflicting patterns and interactions multiplies. For these reasons, there was interest in exploring an ML approach.

A successful ML system can automatically capture some of the variability of natural language in a limited domain, given sufficient labeled data. The clinical skills training setting imposes some important constraints on the ML approach, however. Most significantly, both the users and annotators are, for the purpose of application development, experts. Students using the system to develop their skills, besides having all of the background education necessary for admission to medical school, have already received detailed instruction about how to interview patients by the time they use the virtual patient application. Users are trained in strategies and techniques to query the patient about various aspects of the history of the present illness, as well as the patient's past medical history, family history, and social history. This level of expertise makes it infeasible to collect large amounts of data from a crowd-sourcing platform such as Amazon's Mechanical Turk (Kittur, Chi, and Suh 2008). Furthermore, while this is an important skill to develop, it is far from the only skill taught in a medical curriculum; students are busy, and opportunities to capture reasonable numbers of interactions with the application are limited. Annotation of the correct interpretation of each query also requires medical expertise and familiarity with the set of possible answers programmed into the virtual patient, since subtle linguistic distinctions may have significant medical implications (consider "Have you used drugs?" versus "Are you using drugs?"). Thus, the virtual patient is situated in a relatively data-scarce problem space. Exacerbating the general data scarcity issue is a label imbalance issue. As designed, there are a relatively small number of questions that nearly everyone asks; on the other hand, there are also a large number of rarely asked, but perfectly valid, questions that the patient should be able to answer. That is, the label frequencies exhibit a Zipfian long tail, as discussed in Section 3.

As alluded to earlier, in prior work we developed a ML model based on convolutional neural networks (CNNs) to perform the question identification task in our domain. In a retrospective study using over four thousand dialogue turns collected from a ChatScript-only system, the machine-learned model was seen to have complementary performance with the existing ChatScript system as a function of label frequency, and combining the two systems led to a nearly 50% reduction in errors. We replicated that work for the sake of incorporating it into a system suitable for a live educational deployment, and even though it overlaps substantially with the previous paper, we present that replication in Section 4 for the sake of completeness. And while iterative development and data collection for dialogue systems are nothing new in the literature (e.g., Glass 1999), this work illustrates a case where we can bootstrap sufficient data to train ML models using a rule-based system instead of a more typical Wizard-of-Oz scenario and then incorporate those models into the deployed system over successive design iterations. Meanwhile, even after enough data have been collected to enable machine-learned models to be of benefit on the whole, the rule-based component of the hybrid design can continue to shine on the numerous less frequent questions, or even new ones.

The core hypothesis motivating this paper is that, based on the work of Jin *et al.* (2017), the retrospectively developed hybrid system should continue to be more accurate than a purely rule-based system in a live, randomized, prospective setting. The experiment reported in Section 5 provides strong support for this hypothesis, although performance in absolute terms is substantially lower than expected based on the retrospective study. This is due in part to the fact that we evaluate the deployed system without omitting any individual turns from whole conversations, which contrasts importantly with related work (see Section 2). Our analysis explaining the performance discrepancy under this evaluation regime highlights the final major design decision that we consider throughout this paper, which is the manner in which misunderstandings are handled.

When the system fails to find any interpretation of a given input query, it replies with a *generic prompt* (Stoyanchev, Liu, and Hirschberg 2014), which we call the No Answer response—for

example, “I’m sorry, I didn’t understand. Could you rephrase the question?” This meets the need to signal misunderstanding in a reasonably plausible way without requiring the technical effort to implement more elaborate clarification strategies. It does, however, come with some trade-offs, particularly with respect to evaluations. While the reprompt represents a failure to provide the user with the expected response to their question, it is in some sense a minor error, since it provides no incorrect information. Contrast this with the major error of providing an answer to a question that was not asked, which could mislead or confuse the user, and damage the educational value of the agent. Modeling choices might reduce the number of such minor errors, but those reductions may come at the expense of more major errors. Thus, we treat these misunderstandings as a special case in our evaluations throughout the paper, and we identify these responses as the main driver of the difference in absolute performance of the hybrid system between the experiments in Sections 4 and 5.

However, completely eliminating reprompts through retraining and redesign is not actually desirable, since a substantial number of queries are off topic or otherwise out of scope. In such situations, the *correct* action for the agent is to misunderstand the user. Thus, we offer an experimental analysis in Section 6 of the data collected in our prospective experiment, in which we demonstrate the difficulty of identifying out-of-scope queries and retrospectively develop new models that modestly improve performance on the problem. This then illustrates the beginning of another iteration in our design process, where we retrospectively develop further performance and design improvements, to be prospectively validated in future deployments. We briefly discuss such follow-on work in Section 7 to round out the paper.

2. Background and related work

Our earliest efforts to incorporate ML techniques into our virtual patient domain formulated the question identification task as a pairwise paraphrase identification task (Jaffe *et al.* 2015). That is, each input sentence was evaluated against every known canonical question for whether the pair were paraphrases or not, the evaluations were ranked, and the best match was returned as the answer. Consistent with other results (e.g., Ravichandran, Hovy, and Och 2003; DeVault, Leuski, and Sagae 2011a), our subsequent work found that a direct multiclass classification of the input was more successful when more data were available (Jin *et al.* 2017), although the former approach shaped the development of the core dataset in important ways (see Section 3). Besides improved accuracy, the multiclass classification approach is faster than pairwise comparisons over hundreds of classes, lending itself better to real-time use, as we deploy here.

Although ChatScript offers advanced context awareness and dialogue management capabilities, our deployment makes comparatively little use of these features beyond what is inherent in the topic-based organization of the rules. One exception is that the agent can track the number of times certain questions are asked. This allows the patient to enumerate secondary or tertiary complaints, for example, as replies to repeated queries of “Is there anything else you would like to talk about today?” However, the ML components are—by design, at this stage of development—ignorant of the history of a given conversation; this positions our agent among other so-called question-answering dialogue systems (e.g., Robinson *et al.* 2008; Traum *et al.* 2012), which generally yield the conversational initiative to the user and avoid most of the problems of language generation by providing fixed responses to known sets of questions.

Our virtual patient has some similarity to successful entrants in the Alexa Prize competition (see Khatri *et al.* 2018; Ram *et al.* 2018), in that we deploy a hybrid rule-based and data-driven approach to build a dialogue agent. Of course, as an open-domain challenge, Alexa Prize entrants face a variety of problems that our task definition obviates. Most of the rule-based components of successful systems in the competition are involved in dialogue management and response generation, as opposed to the NLU stage of the pipeline, where our rules are most prominent.

Data-driven approaches to response generation are topics of active research (e.g., Li *et al.* 2017; Zhao and Eskenazi 2018); however, generative models are not to the point where they can be trusted to respect appropriate entailment values, so they are inappropriate for the virtual patient domain, since the educational objective depends on students receiving correct information from the patient. In addition, the medical educators providing content have a strong investment in providing standard answers to questions students may ask.

Looking further afield, hybrid rule- and data-driven approaches are fairly common in biomedical information extraction tasks. Many successful approaches to tasks such as clinical concept normalization have long relied on cascades of progressively less exact dictionary lookups. Such systems incorporate machine-learned modules for specific decisions in the extraction pipeline, or as fallbacks when rule-based approaches fail (e.g., Pradhan *et al.* 2015), although a recent approach that used only deep learning outperformed such systems (Luo *et al.* 2020). This kind of strategic search for a canonical match bears a strong resemblance to how ChatScript decides to respond to input, but the way we join rules with statistical models is by learning which system to trust, rather than using a heuristic to determine when to employ the learned model.

Dialogue agents in medical settings are not unheard of outside of the present program, and there are a variety of goals associated with their use. Talbot *et al.* (2012) provide a relatively recent review and taxonomy of various types of virtual patients and reasons for deploying them. Reports of prospective validations of the techniques used in their deployment are mixed. Morbini *et al.* (2012, 2014) use an avatar to explore mixed-initiative conversations in a troop-deployment, counseling-related scenario, although they do not report full details of a prospective user validation study. DeVault *et al.* (2013) use another such agent to assess diagnostic vocal characteristics of patients with post-traumatic stress disorder, in a retrospective study using Wizard-of-Oz data. In a randomized prospective experiment, Triola *et al.* (2006) found no differences in an array of educational criteria between subjects trained with virtual patients as compared to human standardized patients, although their virtual patient did not emphasize the use of free-form natural language questions. More recently, a team at LIMSI has built a virtual patient with very similar educational goals to ours, which takes a heavily engineered approach to defining rules and ontologies that allow for accurate question answering given an authored patient health record (Campillos-Llanos *et al.* 2016, 2019, 2021). This system emphasizes flexibility in deploying virtual patients to represent a wide array of conditions and histories, at the expense of some naturalness in responses. Accordingly, they do not make use of ML techniques, since adequate conversational data to represent all possible ailments are not available. Due to their goal of flexibility, they take an entity recognition and linking approach to the NLU component of their agent, which they evaluate on traditional entity recognition/extraction metrics such as F1 or slot error rate. They also evaluate the correctness of the system's response in their more recent work, which aligns closely with the accuracy metric that we use to evaluate our system. Their correctness is comparable or slightly lower in absolute terms than the results of our prospective experiment, although their result is averaged over many cases instead of one. Notably, they omit individual dialogue turns which they deem to be out-of-scope or which represent dialogue acts that their system is unable to handle, where we consider whole conversations. This has a smaller effect on their correctness metric than it would for us, since their test subjects all have a minimum of 3 years of medical education, where ours are first-year students who are less focused in their interviewing technique. Handling this issue is a focus of the present work. Uniquely, our system's immediate student feedback has been validated as demonstrating high agreement with expert human graders (Maicher *et al.* 2019).

In accordance with our goal of creating an effective tool for the education of our “novice experts,” we confront the issue of handling out-of-scope queries in our analysis in Section 6. The related problem of distinguishing in- and out-of-domain data when only in-domain data are available is well-studied in the literature, and known by several variants falling under the banner of one-class classification, including novelty detection, anomaly detection, outlier detection, etc. (Khan and Madden 2009). Techniques leverage various assumptions about the data in some

representation space, for example, that it be maximally separated from the origin (Schölkopf *et al.* 2000), or that the measure of a hypersphere surrounding it be minimized (Tax and Duin 1999). Many published techniques in the text domain make use of large corpora of unlabeled in- and out-of-domain data to provide a basis for the negative class (e.g., Yu, Han, and Chang 2004). In the absence of negative examples, adequately characterizing the boundaries of the positive class requires a large number of positive examples (Yu 2005). However, if a large number of unlabeled examples are available, good results can be achieved by maximizing the number of them that are classified as negative, while imposing a hard constraint that positive examples be correctly classified (Liu *et al.* 2002). Neural approaches to one-class classification have been popular more recently, and these focus on learning a representation distribution that is amenable to outlier detection, for example, using ordinary multiclass classification as a joint auxiliary task for one-class training with a class compactness objective (Perera and Patel 2019). Recent work in a command-and-control spoken dialogue context uses joint out-of-domain and domain classification objectives along with an externally supplied false acceptance target rate to improve domain classification accuracy (Kim and Kim 2018).

In a task related to our need to identify out-of-scope queries, Yu and Sable (2005) show some success at identifying *unanswerable* questions in a very small dataset from the perspective of a physician consulting an expert system. Notably, their dataset is fairly balanced, and queries about patient-specific information are deemed unanswerable, which stands in contrast to our system which *only* cares about patient-specific information. In another study of particular relevance, the LIMSI team mentioned above retrospectively developed an ML classifier that detects whether or not an input query will be handled well by their rule-based system (Campillos-Llanos, Rosset, and Zweigenbaum 2017), which has many similarities to our chooser and to our out-of-scope detection issue. However, their in-scope queries seem to be limited to the patient's history of present illness, and they apparently do not implement an alternative question-answering approach, for the case in which a query is identified as unsuitable for the rule-based system.

In the following section, we introduce in more detail the data used to train our machine-learned model, as well as the data collected over the course of our prospective validation experiment, which then serves as the basis for our analysis of the out-of-scope problem in our domain.

3. Data

The main dataset used in the following experiments was collected from students interacting with the initial ChatScript-based virtual patient as voluntary, extracurricular practice for an examination with a standardized patient. Each data point consists of the student's query sentence, the interpretation of the query by ChatScript, ChatScript's response, and the correct label. Labels in this case are a canonical sentence with an equivalent meaning of the query sentence. As discussed in the introduction, this is in some sense a paraphrase identification task, but since there are a finite number of relevant queries to which the patient can respond, we treat the question identification problem as a multiclass classification problem. So for example, the queries:

- (1) "And since then you have had the constant back pain[?]"
- (2) "OK, and does it hurt you all the time?"

should both be identified as belonging to the same class and should produce the same response starting with "It is pretty constant, although sometimes it is a little better or a little worse." This class is labeled with the canonical sentence "Is the pain constant?" A list of canonical labels is provided in Appendix B.

Preliminary analysis identified a small number of classes that were deemed semantically equivalent for the purposes of assessing the validity of the data-driven approach, and these were manually collapsed to representative labels. As an example, the two queries:

- (1) “Tell me about your parents.”
- (2) “Are your parents still alive?”

are semantically distinct in reality, and the ChatScript engine recognizes them as distinct inputs. However, by design, they each produce the response “Both of my parents are alive and well,” and they are mapped to the same class for the purpose of training the ML system. We further note that the class counts presented below are drawn from the correct labels, exclusive of the potentially incorrect interpretations that ChatScript may have made at the time of data collection.

This core dataset was originally developed as a follow-up to preliminary work by Jaffe *et al.* (2015), which did formulate the question identification task as explicit paraphrase identification. In this setup, pairwise comparisons between an input and all labels were performed, ranked, and the best match was returned. To evaluate this approach, certain classes whose instances were not paraphrases were excluded—chiefly negative classes defined mainly by an absence of some information as opposed to its presence. These include a catch-all class for questions about asymptomatic physiological systems which are unrelated to the patient’s chief complaint, which can include non-paraphrases such as “Do your fingers ache?” and “Do you have any shortness of breath?” We call this class the Negative Symptoms class. Another such class—which we call the No Answer class due to our agent being unable to meaningfully respond—are questions that are out-of-scope entirely, for various reasons we discuss in detail in later sections. After development of the dataset for evaluation of the paraphrase identification approach, it was discovered that the multiclass classification approach was both more effective with the volume of data available and faster, making it more conducive to a real-time deployment. Nonetheless, the exclusion of these nonparaphrastic classes was not reevaluated, and development was standardized around this dataset. This innocent historical decision had significant impacts on our prospective experiment, as described in Section 5.

The original dataset in the above format consists of 94 dialogues, comprising a total of 4330 queries and responses, representing 359 classes (after collapse). We refer to this set as the *core dataset*. Our second experiment was a prospective, blinded comparison of a ChatScript-only system and combined ChatScript/CNN system, using 110 first-year medical student volunteers as subjects. In this experiment, we collected a further 154 dialogues, comprising 5293 turns. We refer to the entirety of this second set as the *enhanced dataset*, with further subdivisions of *CS-only* and *hybrid* for the data obtained from the ChatScript-only and hybrid systems, respectively (as discussed in Section 5). The enhanced dataset contains 258 classes that were seen in the core dataset, plus 74 rare classes that were new or had been previously unseen. The unseen classes comprise 142 turns of the enhanced dataset, or approximately 2.7% of the data. A selection of queries from a conversation in the enhanced dataset is available in Table C1 in Appendix C.

Since the data are essentially natural dialogues, the occurrence counts of classes are extremely unbalanced, exhibiting a Zipfian (Zipf 1949) long tail. For example, nearly every interaction will include some variant of “Is there anything else I can help you with today?” while “Does moving increase the pain?” is far less common, but clearly clinically significant. A quintile analysis of the core dataset is shown in Figure 2, showing the frequency of occurrence of each class. The top quintile consists of only 10 classes, while the bottom quintile consists of 256 classes, many of which appear only once in the dataset, but which nevertheless should be recognized and answered correctly.

4. Experiment 1: Retrospective development

We note again that the work in this section is a reproduction of work published elsewhere (Jin *et al.* 2017), but as the remainder of work builds directly upon it, we present it here to provide a complete context. We used the same code base and data, with a different random permutation of the dataset, while also incorporating data corrections and bug fixes. In particular, we corrected an

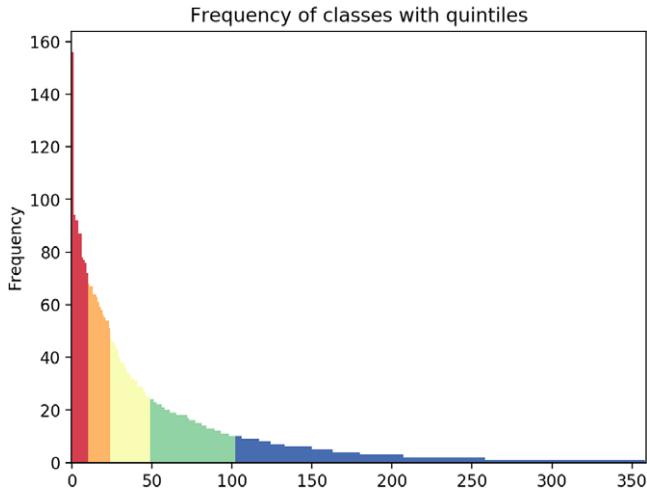


Figure 2. Frequency of labels in the core dataset by rank, with quintiles by color.

error in the calculation of the *CS Log Prob* feature (described below), which in turn highlighted approximately 2% of the core data in which the ChatScript response was labeled inconsistently with the true label set. Chronologically, these corrections took place after the experiment reported in Section 5, but before the reproduction work described in this section.

A number of difficulties associated with the use of rule-based dialogue systems were enumerated in the introduction. These challenges motivated the model we present here. Given the challenges of the rule-based approach, we sought to determine the effectiveness of a more data-driven approach. We hypothesized that a ML system might offer some benefits; specifically, the use of word embeddings could capture similarity and relatedness of words (Mikolov *et al.* 2013), as observed through co-occurrence statistics in a large corpus of text. This could allow for external data to effectively supplement the relatively small dataset available for the task in an unsupervised way. More direct supervision, by training a model to perform the question identification task, could also discover discriminative features of the input, without requiring manual analysis on a per-pattern basis, as would otherwise be required with the rules-based approach.

Toward this end, we used a CNN on the query text input (Kim 2014) to do the question identification, described in detail below. We found that ensembling multiple CNN models was important for mitigating noise in the data due to general data sparsity and label imbalance, and ensembling models trained on different representations of the same input (word or character sequence) offered further benefit. Finally, we showed that the error profiles of the ensembled CNN models and the rule-based ChatScript system have some complementary properties, and that combining these systems with a simple binary classifier led to a substantial reduction in error compared to ChatScript alone.

As discussed in the previous section, we refer to the data used in this experiment as the *core dataset* in order to distinguish it from our subsequent data collection (the *enhanced dataset*) in Section 5.

4.1. Stacked CNN

The overall structure of the model presented here (the *stacked CNN*) is an ensemble of ensembles. Later, we show further benefits from choosing between the output of the stacked CNN and the output of the original rule-based system.

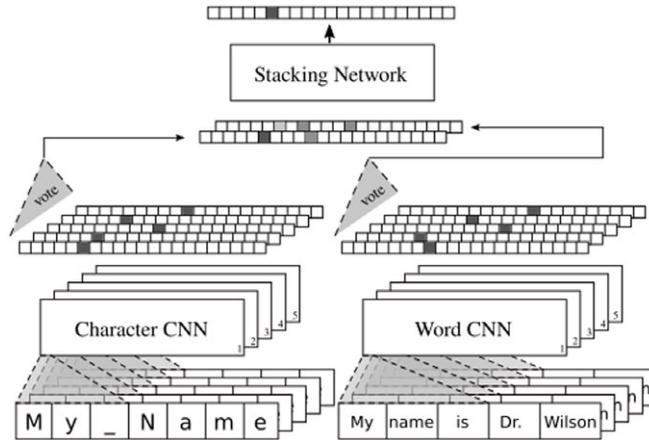


Figure 3. Overview of the stacked CNN architecture.

At the highest level, the stacked CNN makes a classification decision based on the outputs of two ensembles of CNNs operating on different forms of the same input. One of these two sub-ensembles operates on the sequence of words in the query sentence (*word CNNs*), represented as word embeddings (Mikolov *et al.* 2013); the other sub-ensemble works on the sequence of characters in the same query (*character CNNs*), again represented as embeddings, but in a different space than the words. Each sub-ensemble consists of five CNNs, which we call the submodels, described in detail below. Each CNN is trained on a different split of the data, described in Section 4.2. See Figure 3 for a graphical overview of the stacked CNN.

4.1.1. CNN submodel

In this section, unless otherwise indicated, the architectures of the word CNNs and character CNNs are identical.

The structure of each CNN is a one-layer convolutional network with max-pooling, which then feeds into a single fully connected layer, whereafter the input is classified as one of the 359 classes in a final softmax layer.

The input of the word CNN model is a sequence of k -dimensional word embeddings, making the input a $T \times k$ matrix of real values for a sentence of length T . Word embeddings are held fixed during training. We use pretrained Word2Vec vectors trained on the Google News corpus,^a thus $k = 300$. We experimented with GloVe vectors (Pennington, Socher, and Manning 2014) but observed a slight performance degradation. Static embeddings trained on biomedical corpora (e.g., Chiu *et al.* 2016) may present a compelling area for future research but are expected to offer little benefit, since students are taught to avoid jargon when communicating directly with patients; indeed, a typical patient should *misunderstand* the majority of precise medical terminology.

Analogously to the word CNN, input to the character CNN is a sequence of randomly initialized 16-dimensional character embeddings. In contrast to the word embeddings, these are tuned during training.

Let W be a set of integers, representing a set of kernel widths. The convolutional layer is a set of m filters each of size $w \times k$, $w \in W$, for a total of $m \times |W|$ kernels. These are all convolved over the time axis of the input (i.e., the length of the sentence) to produce a feature map for the sentence. Since the kernels span the full length of the input vector, k , they output a single value per input element, which takes into account a variable amount of context depending on the kernel width w ,

^a<https://code.google.com/archive/p/word2vec/>

leaving the output in $\mathbb{R}^{T \times m \times |W|}$. For the word CNNs, m is 300 and $W = \{3, 4, 5\}$, while for the character CNNs m is 400 and $W = \{2, 3, 4, 5, 6\}$. The output of each kernel is passed through a rectified linear unit (Nair and Hinton 2010).

We used max-pooling over the length of the sentence (Collobert *et al.* 2011) to represent the sentence in a fixed size. This fixed dimensional output of the convolutional layer is passed through a final fully connected linear layer containing as many units as there are classes, and the final classification is determined by a softmax operation over the output of this layer.

We found it beneficial to employ a few regularization strategies. We used 50% dropout (Srivastava *et al.* 2014) at the output of the max-pooling layer and used a variant of the max-norm constraint employed in Kim (2014). Specifically, where they renormalize a row in the weight matrix to the specified max norm only if the 2-norm of that row exceeds the max, we observed a benefit from *always* renormalizing to the specified norm. This renormalization strategy came from a reimplementation of Kim (2014).^b We set the max norm to 3.0.

4.1.2. Ensembling

Since our core dataset is relatively small, we validated our training progress using cross-validation development sets; furthermore, since the label frequencies are very unbalanced, any given random split can produce significant differences in test performance. We sought to minimize this variance in model outputs by ensembling multiple CNN submodels, each trained on different splits of the data.

For each input form (words and characters), we trained five of the CNN submodels and combined their outputs with simple majority voting. During evaluation in the training of sub-ensembles, ties were arbitrarily broken in favor of the class with the lower index.

We combined the outputs of each ensemble of CNNs using stacking (Wolpert 1992). This is essentially a weighted linear interpolation of system outputs, where the weights assigned to each system are trained on the data. Thus, the final output of the stacked CNN \hat{y}_t is

$$\hat{y}_t = \text{softmax}(\alpha_w \hat{y}_{e,w} + \alpha_c \hat{y}_{e,c}) \quad (1)$$

where $\hat{y}_{e,w}$ is the output of the word ensemble, $\hat{y}_{e,c}$ is the output of the character ensemble, and α_w and α_c are the trained coefficients for word and character ensembles, respectively.

4.1.3. Parameters

Any words that did not appear in the set of pretrained embeddings were assigned random values with each dimension drawn from the distribution $Unif(-0.25, 0.25)$. Care was taken to ensure that this distribution fell within the range of the rest of the embeddings, as values outside this range were seen to reduce performance of the word CNNs.

Convolutional kernels in the word CNNs were initialized with $Unif(-0.01, 0.01)$, and the linear layer was initialized using a Normal distribution with zero mean and variance of 1×10^{-4} . In the character CNNs, weights for both the convolutional kernels and linear layer were initialized by drawing from $Unif(-1/\sqrt{n_{in}}, 1/\sqrt{n_{in}})$, where n_{in} is the length of the input, as recommended by Glorot and Bengio (2010). The character embedding matrix was initialized from $N(0, 1)$, and all bias terms (for both the word and character CNNs) were initialized to zero.

4.2. Training

We used Adadelta (Zeiler 2012) to optimize submodel weights, using recommended parameters ($\rho = 0.9$, $\varepsilon = 1 \times 10^{-6}$, initial learning rate = 1.0), and a cross-entropy loss criterion.

^b<https://github.com/harvardnlp/sent-conv-torch>

Table 1. Mean accuracy across 10 folds, with standard deviations.

	Single	Ensemble
ChatScript	80.93%	n/a
Baseline	76.93 ± 1.58%	n/a
Word	76.17 ± 1.23%	76.95 ± 1.87%
Character	75.28 ± 2.08%	77.09 ± 1.53%
Stacked	n/a	78.01 ± 1.62%

We used 10-fold cross-validation to train each CNN submodel and validate performance; thus, for each of the 10 test folds, we trained 5 CNN submodels, reporting average performance across folds. Each of the five submodels in a fold was trained on a different training/development split of the 90% of data remaining after the test fold was held out, with the development set comprising the same number of examples as the test set, and the remaining 80% used for training. Importantly, each training set was supplemented with the label sentences—that is, the canonical example—for each class. This ensured that no class was completely unseen during training, which would otherwise be likely, given the label imbalance of the data.

We trained for 25 epochs with minibatches of size 50—empirically, models always converged within this time frame—and we took the last model with the best dev set performance for test validation. Note that ensembling the submodels by majority voting is a non-differentiable function, so each submodel was trained independently of the others. The performance of the sub-ensembles was validated by using the single majority decision of the vote, but the input to the stacking network was the vector of vote counts for all classes, which was effectively an unnormalized distribution. The stacking network was trained by holding the submodel parameters fixed (again, voting is non-differentiable) and training for another 25 epochs. The optimizer was the same as that used for the submodels above, that is, Adadelta with recommended parameters.

4.3. Baseline

As a baseline comparison for the stacked CNN, we trained a simple maximum entropy (logistic regression) classifier implemented using SciKit-learn (Pedregosa *et al.* 2011) with n -grams as input features, which is very similar to the method of DeVault, Sagae, and Traum (2011b). Specifically, we used 1, 2, and 3 g of both the raw word forms and their Snowball-stemmed (via NLTK; Porter 2001; Bird, Klein, and Loper 2009) equivalents, as well as 1 through 6 g of characters. The model was optimized using the stochastic average gradient for up to 100 epochs. We used the same cross-validation strategy and the same splits as described above.

4.4. CNN results

We report performance in terms of overall accuracy, averaged over the 10 folds. Results, including standard deviations, are summarized in Table 1. We present a comparison of the performance of single models in the cross-validation setup vs. the full ensembles, to clearly illustrate the benefit of ensembling in our limited data regime. The *Word* and *Character* entries are for the corresponding components of the full stacked CNN. We again note that these numbers vary slightly from previously published work (Jin *et al.* 2017) due to data corrections, different random cross-validation splits, and random model initializations.

The naive baseline exhibited surprisingly strong performance, which proved to be difficult to beat. Indeed, none of the single models were able to do so, which, along with the high variance

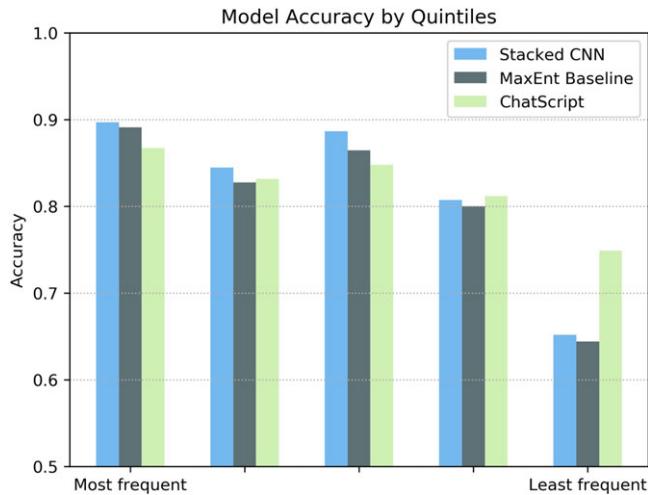


Figure 4. Accuracy of the tested models by label quintiles.

across folds, motivated the ensembling approach. We largely attribute the performance of the baseline to the scarcity of the data, and expect, based on further analysis below, that more data would widen the gap between it and the stacked CNN.

The ensembling strategy generally seemed to boost performance, although the effect was larger for the character-based models than for the word-based models. Individually, the word and character ensembles exhibited similar performance to the maximum entropy baseline, but stacking the word and character sub-ensembles gave a significant gain over the baseline ($p = 0.0074$, McNemar's test). This suggests that the word and character sub-ensembles are picking up on complementary information.

Some examples suggest that the stacked CNN was able to take advantage of similarity information that is latent in the word embedding space. For instance, where the maximum entropy baseline chose the label “Do you have any medical problems?” in response to the query “Has your mother had any medical conditions?” the stacked CNN correctly chose “Are your parents healthy?” Presumably similar embeddings for “mother” and “parent” were helpful here. Another benefit of the stacked CNN seems to be robustness to some particularly egregious misspellings, as it identified the correct class for the input sentence “could you tell me more about the pback pain,” for example.

Despite improving over the baseline, the stacked CNN still failed to outperform ChatScript, which highlights the benefit of rule-based approaches in the face of limited data. However, a closer look at the accuracy on specific classes when grouped by frequency, shown in Figure 4, reveals some important trends. First, the stacked CNN consistently performed slightly better than the maximum entropy baseline at all label frequencies. More interestingly, for the 60% of examples comprising the most frequently seen labels, the learned models outperformed ChatScript. For the least frequently seen data, the learned models exhibited a substantial degradation in accuracy, especially relative to ChatScript. ChatScript was less sensitive to the data scarcity in comparison to its performance on more frequent labels, although it did exhibit some degradation at lower label frequencies.

This difference in error profiles between ChatScript and the stacked CNN suggested that some combination of both systems might yield further improvements in accuracy. Indeed, one or both of the systems answered correctly in 92.45% of examples, establishing a rather impressive performance for a hypothetical oracle that could always correctly choose which system to trust.

Table 2. Features used by the chooser.

Feature	Description
Log Prob	The log of the probability of the class chosen by the stacked CNN
Entropy	The entropy of the distribution over the classes, as determined by the stacked CNN
Confidence	The average over submodels of the unnormalized score for the chosen class
CNN Label	The one-hot label predicted by the stacked CNN
CS Label	The one-hot label matched by ChatScript, or zero if there was no match
CS No_ans	Boolean indicator that is true if and only if ChatScript did not find an answer
CS Log Prob	The log probability of the class chosen by ChatScript, according to the stacked CNN

This oracle performance motivated us to develop a simple model that could decide to override ChatScript's answer with the stacked CNN's answer, which is described in the next section.

4.5. Hybrid system

Since ChatScript and the stacked CNN exhibited complementary error profiles, we sought to determine to what extent we could practically benefit from a system utilizing both approaches. To do this, we constructed a simple binary logistic regressor using SciKit-learn (Pedregosa *et al.* 2011) to choose either the rule-driven or data-driven prediction, dependent upon some features that could be extracted from each system. We (unimaginatively) refer to the logistic model as the *chooser*, or the *two-way chooser* to distinguish from the three-way variant we introduce later, and hereafter refer to the overall system (the stacked CNN combined with ChatScript using the chooser) as the *hybrid system*.

Obviously, the correct label is not known at test time, so the chooser must use features of the input to make its decision. However, the chooser does also have access to meta-information about the decisions made by the stacked CNN and ChatScript, such as confidence of the decision, entropy of the distribution of output labels, or agreement between systems. As a rule-based system, ChatScript does not provide any statistical information about its decision, so meta-information is more limited there. Based on our experiments with the available features, we found those enumerated in Table 2 to be most effective.

For the majority of examples in the dataset (over 66%), both systems were correct, so choosing either system would have been acceptable; similarly, in approximately 7.5% of examples, either choice would have been wrong. Conversely, when both systems agreed, they were correct 98% of the time. Therefore, we only trained the chooser on the examples where the systems disagreed, taking agreed-upon classes as the answer in every case. This did, however, raise the question of which system's answer to use as the default to be overridden when the chooser deemed it appropriate, and further, how the choice of that default affected the accuracy on unseen data. The main difference between the systems in this regard is the previously mentioned No Answer behavior. That is, if ChatScript cannot match any available patterns, it provides a default response that indicates that the question was not understood and then asks the user to reformulate the question. On the other hand, the stacked CNN, as discussed in Section 3, was effectively only trained on positive classes. From a user's perspective, a No Answer response is a system failure: the user asked a question and did not receive an appropriate response. As mentioned earlier, however, sometimes receiving no answer is better than receiving an *incorrect* answer; such an answer could give the illusion of understanding while providing an unintended response to the actual question asked,

Table 3. Hybrid system results. ChatScript and stacked CNN baselines are repeated from above. “Conf” is using only the confidence feature from Table 2; “base” is using Log Prob, Entropy, and Confidence. CS = ChatScript, CNN = stacked CNN.

System	Features	Default	Accuracy%	No Answer%
ChatScript			80.93	10.88
Stacked CNN			78.01	0.00
Hybrid	Conf	CS	83.58	6.56
Hybrid	Base	CS	85.43	5.52
Hybrid	All	CS	89.40	3.23
Hybrid	All	CNN	89.97	0.00
Oracle			92.45	4.16

potentially leading to further unanswerable questions, for example. For these reasons, we examined the effects of the two different choices of default response, that is, ChatScript or the stacked CNN, including the effect on frequency of No Answer responses. The performance of the hybrid system was evaluated using 10-fold cross-validation.

Results of the hybrid system, including overall response accuracy and the percentage of No Answer responses, are summarized in Table 3. Note that, consistent with previous tables, this is the question classification accuracy, not the accuracy of the decision of which system to choose. We include the results of ablating some of the features used by the chooser. Statistical measures from the stacked CNN alone served to improve performance over the ChatScript baseline fairly substantially, but the more detailed information about which classes were chosen by each model, and the extent of the two models’ agreement, provided a dramatic boost in performance. Using all features while defaulting to ChatScript’s response constituted a 44% relative reduction in error from ChatScript alone and recovered almost 74% of the oracle performance.

Using the stacked CNN’s response as the default yielded a slightly higher overall accuracy, but since the stacked CNN never gave the No Answer response, the whole system never did either, in this condition. This means that for the approximately 3% of answers that were incorrect but had the relatively benign No Answer response with the ChatScript default, most of them were still incorrect under the stacked CNN default, but potentially misleadingly so. We assume this trade-off favors using the ChatScript default in further experiments. As we will show in those experiments, it turns out to be very likely that some No Answer responses are necessary, so their complete absence is probably undesirable.

5. Experiment 2: Randomized prospective validation

Having established the benefit of a hybrid data- and rule-driven approach on a finite dataset, we now seek to probe how effective the hybrid system is in live conversations. In other words, we have seen retrospectively how, all else being equal, a hybrid system might have improved the performance for a specific set of queries from a specific cohort of medical students who were actually interacting with a different system. We recognize, however, that all else would not be equal—better responses, or even just different ones, can affect the course of a conversation, and that could affect the performance in ways that are difficult to predict. Therefore, we deployed the hybrid system as part of a controlled experiment in which a new cohort of medical students was randomly assigned to interact with the hybrid system or a system using only ChatScript. In this way, we can prospectively determine the efficacy of the system. The production deployment also serves as a verification

of the various usability factors that must be considered for modern human-computer interaction, given the latency and computational demands of the data-driven model.

Even though live interaction introduces unknown variables, we expect that the core dataset is representative of the distribution of questions that should be asked of our virtual patient; thus, this is essentially a replication experiment, and we hypothesize that the deployed system should corroborate our main findings from the first experiment. In particular, the hybrid system should be more accurate than ChatScript alone, and we should see comparable performance in terms of absolute accuracy. In the following sections, we describe the architecture of the production deployment, the experimental design that we use to test our hypotheses, results of the experiment, and a discussion of those results.

5.1. System architecture

At the most superficial level, the virtual patient is deployed as a simple client/server architecture, with the client bearing responsibility for user interface functions, including rendering the animated image of the patient to the user, displaying responses, and passing the user's queries to the back-end server. In the present work, the client was deployed as a web page, but there are many possible alternatives, including mobile applications or virtual reality headsets. The server, on the other hand, houses the apparatus for actually interpreting the user's query, formulating a response, tracking the state of the conversation to the extent that such tracking is done, storing data for later analysis, and automatic assessment of student performance. The server is a simple web service communicating with the client via HTTP, which in turn communicates with the ChatScript instance over socket connections, allowing the ChatScript instance and the stacked CNN to be developed by separate teams.

When a question is received, the web service sends the input to both the stacked CNN and ChatScript, and from their outputs, calculates the input features for the chooser. If the chooser indicates that the stacked CNN has the better interpretation, the canonical question for that class is sent in a second volley to the ChatScript instance, and its answer is then sent to the user. If the chooser chooses ChatScript, the first response is just returned immediately.

Subjectively, the latency of the hybrid system is greater than for the ChatScript-only system, but within acceptable limits. We observe that most of the latency is involved with running the CNN ensembles on a single CPU, and the effect of network latency for communication with ChatScript, even over two volleys in the case that the stacked CNN is chosen, is negligible.

For a full accounting of technical details, please refer to Appendix A.

5.2. Experimental design

Our experiment aims to determine if the performance improvement on the fixed dataset translates to users interacting with a system dynamically. To do this, we prepared a simple blinded experiment, where, upon starting a conversation, students are randomly assigned to either use the hybrid system or a ChatScript-only version of the virtual patient, with equal probability. Use of the virtual patient application was completely voluntary and presented to students as an opportunity to practice their interviewing skills prior to an exam with a standardized patient. Because of the voluntary participation, the population as a whole may reflect some self-selection bias, but the test and control groups should be affected equally, due to the random assignment at the beginning of each conversation. The virtual patient himself has the same symptoms and condition but has minor changes from the agent represented in the core dataset—including, for example, a different name, a different reported activity at the time of pain onset, and changes in responses that previously tended to elicit unanswerable follow-up questions. None of these changes are expected to affect the semantics of the incoming questions, or the task of classifying those questions, although there may be a change in the frequency distribution of questions.

After the participation period, conversation logs were collected from the ChatScript instance, and the authors annotated each ChatScript response for correctness given the corresponding input query. In the event of any uncertainty about the correctness of a response, the dialogue turn was flagged for assessment by the medical expert. Incorrect responses were then annotated by the medical content expert for the correct class. Correctness of the stacked CNN's response was determined by its match to the annotation of the correct ChatScript response (after collapse of semantically similar classes, see Section 3). This seems obvious and is reasonable, but we note that it introduces some bias in the annotations, since in some cases multiple classes may provide valid answers to the question asked. A very simple example is the question "Are you single or married?" ChatScript interprets the question as "Are you single?" while the stacked CNN interprets it as "Are you married?" The separate labels exist for the possibility of producing a response that matches the polarity of the question, but for simplicity, both correctly produce the response "I am single." In such cases, ChatScript's acceptable response would always be favored over the stacked CNN's acceptable response.

5.3. Results

Data were collected over the course of approximately 2 weeks, resulting in well over 5000 individual conversation turns. Participants were identified only through self-provided names, which were not normalized against known students. Accommodating for trailing spaces, obvious uses of initials, and common nicknames, we identified 110 unique users in the dataset. Eighteen of those users had conversations with both systems, 10 had more than one conversation with the ChatScript system, 14 had more than one conversation with the Hybrid system, and 2 had multiple conversations with each.

The interface design inadvertently created some confusion among users about submitting individual queries versus submitting the entire conversation for scoring, which led to some conversations ending after only a turn or two. To the extent that this was apparent in the data, we removed these conversations, as well as some which were judged to have been cursory probes of the system's capability for testing or demonstration purposes (e.g., "Hello Mr. Wilkins. I like your shirt," followed by ending the conversation, or a conversation including the query "Are you a rodeo clown?"). We limited our data removals to be entire conversations, in an attempt to ensure that difficult or unusual queries made in good faith would be retained in our evaluations. This pruning resulted in the removal of 543 collected turns, which gave a total of 5296 turns in 165 conversations, which constitute the *enhanced dataset*. Lengths of individual conversations range from 1^c to 103 turns. Summary statistics are provided in Table 4, in which the control group is labeled as *CS-only*, the test group is labeled as *Hybrid*, and statistics for the whole enhanced dataset are provided in the *Overall* column. Although the Hybrid conversations were three turns shorter on average, the difference in conversation length between the CS-only and Hybrid data is not statistically significant ($p = 0.38$, Mann-Whitney U-test).

Raw accuracy results are summarized in Table 5. Note that 1.6% of answers are both No Answer responses *and* correct—that is, a No Answer response is not necessarily incorrect, despite our previous assumption. This is an important point that we focus on later. Otherwise, a few things are obvious from the summary results. First, and most positively, the hybrid (test) system outperforms the ChatScript-only (control) system significantly ($p = 1.9 \times 10^{-15}$, one-sided Pearson's χ^2). Second, the stacked CNN performs worse than the ChatScript component of the combined system but still provides a fairly large benefit when combined with ChatScript using the chooser. Of the approximately 10% absolute accuracy improvement that the Oracle results indicate is possible, about half is recovered by using the chooser. Third—disappointingly—in terms of absolute performance, all systems seem to perform much worse than hoped for based on the

^cSome students quit conversations and later returned to ask single legitimate questions, which were kept in the data.

Table 4. Summary statistics of collected data.

		CS-only	Hybrid	Overall
Conversations		74	91	165
Unique participants		55	73	110
Turns	Total	2,497	2,799	5,296
	Mean	33.7	30.7	32.1
	Median	32	30	31.0
	Std dev	20.8	20.5	20.7

Table 5. Raw accuracies of control and test conditions, and subcomponents of the hybrid system (CS = ChatScript, CNN = stacked CNN).

	Control		Test	
	Acc%	NoAns%	Acc%	NoAns%
Oracle	73.45	11.69	85.21	5.64
CS	73.45	11.69	74.85	12.22
CNN	—	—	67.56	0.04
Hybrid	—	—	80.03	6.57

results of the first experiment. Finally, and somewhat more subtly, the ChatScript component of the combined system outperforms the system relying only on ChatScript (barely reaching significance at $p = 0.049$ in a one-sided Pearson’s χ^2 test). We provide further analysis of the collected data below, which aims to explain these phenomena.

5.4 Discussion

Perhaps the most obvious place to begin accounting for the difference in absolute performance from the first experiment is in differences in the distribution of labels between the core dataset and the enhanced set. The enhanced dataset is labeled with 77 ChatScript patterns that never appeared in the core dataset. These appear as the labels in 142 turns. For the purpose of evaluating the performance of the stacked CNN, we map these into a single *unknown* class, since it could never have produced these classes.

Most of the unseen classes originate from an unanticipated team synchronization issue. As mentioned in Section 5.1, the ChatScript instance and stacked CNN deployment were developed by separate teams for this experiment, and classes were added to the ChatScript instance in an effort to improve performance, unbeknownst to the team deploying the stacked CNN. From the perspective of maximizing performance of the stacked CNN, this would appear at first blush to be an embarrassing source of what might be called engineering noise in the model development process. However, out of 61 questions with unseen labels in the test condition, 60.7% were answered correctly by the hybrid system, with only one instance of the chooser choosing the stacked CNN when ChatScript answered correctly. This is less accurate than ChatScript’s average for the least frequent quintile of data, but certainly far better than the 0% maximum performance of the stacked CNN on these examples. Thus, we count this as a serendipitous illustration of the benefit of a hybrid

Table 6. Top changes in class frequency by magnitude of difference between core and enhanced datasets. *Change* is the count of examples of the class in the enhanced set minus the count in the core set.

Label	Change
No answer	441
Any other problems	184
What should I call you	−156
Unknown	142
Tell me more about your back pain	125
Describe the pain	95
Are you taking any medication	80
Can you rate the pain	78
How much do you work	−76
Are you having any other pain	−75

rule- and learning-based approach to the NLU component of a dialogue system. It allows us to relatively easily handle completely new classes with a baseline-level accuracy, without even having to retrain the ML components.

An additional source of unseen data was the *Negative Symptoms* class. As mentioned in Section 3, these examples had been intentionally removed from the core dataset, since members of this class are not necessarily paraphrases, and a pairwise matching method would not be expected to succeed. The core dataset was developed with this approach in mind, and when the multiclass classification approach was seen to yield better results, the omission of this class was just incidentally not reconsidered. Bringing this oversight to light is certainly an argument in favor of taking the time to conduct the prospective validation study.

Besides such completely unseen classes, the enhanced dataset reflects some substantial shifts in the frequency distribution of the seen classes. Of the remaining 359 classes in the core dataset, only 261 appear in the enhanced set. To give a sense of the differences, Table 6 shows the 10 largest changes in class frequency from the core dataset to the enhanced set.

The No Answer class is by far the biggest difference between the core and enhanced datasets. As described in Section 4, this is the default class, which ChatScript produces when a question strays outside of the dialogue agent's knowledge base. Similar to the Negative Symptoms class, it is a negative class, defined by the absence of a match to a positive class, rather than on any specific content of the query. While the core dataset does contain examples of the No Answer class, it is almost never chosen by the stacked CNN in practice. Effectively, the only way for the hybrid system to produce a No Answer response is for the chooser to choose ChatScript when ChatScript fails to match any other pattern. In the enhanced dataset, there are two main sources of No Answer annotations. The first source is questions which the patient should be able to answer, but which had previously not been considered, or had otherwise not been prioritized by the content author. We refer to these types of queries as being *currently out-of-scope*. The second source is questions which are entirely outside the scope of the educational objective, for example, "What would you consider your biggest strengths?" We call these *permanently out-of-scope*, which generally overlaps with what would be called out-of-domain in dialogue literature. Note that while we make a conceptual distinction between currently and permanently out-of-scope for the purpose

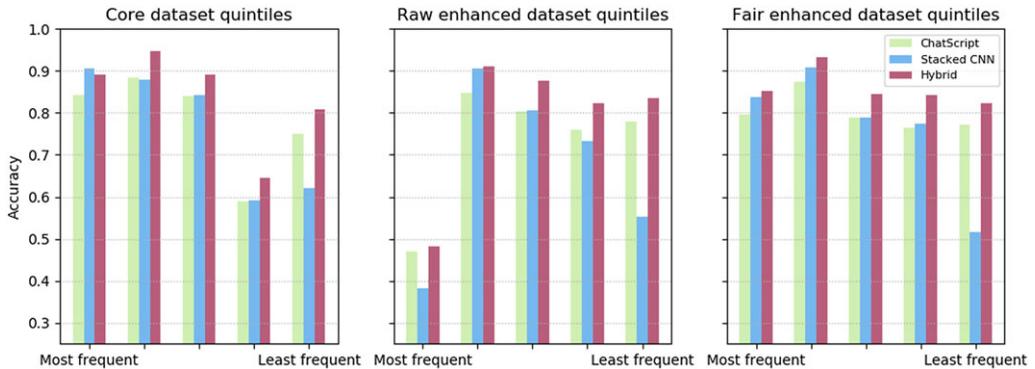


Figure 5. Accuracy of the hybrid system and constituent components by label quintiles. “Core dataset quintiles” have the same label membership as in the previous experiment; “raw enhanced dataset quintiles” are the quintiles according to the frequencies observed in Experiment 2. “Fair enhanced dataset quintiles” remove No Answer labels and other unseen labels from consideration.

of discussion, our annotation scheme only identifies out-of-scope queries generally and assigns them the No Answer label. Unlike what had been generally assumed in the first experiment, the No Answer response is the *correct* response in these situations, not just a least-risk fallback strategy. But since the system was not trained to identify such situations, this class accounts for a large volume of the errors. Addressing this issue is the motivation for our work in Section 6, although it turns out to be a very difficult problem to handle effectively.

The remaining differences in the label distribution between the datasets, exemplified in Table 6, are harder to explain definitively. The data in each set are collected from different cohorts of medical students; as such, they may have received slightly different training, either as natural variation among instructors or from intentional change in emphasis on particular topics from one year to the next. For example, the “Tell me more about your back pain” class reflects a conversational strategy of asking open-ended questions, which is explicitly taught, and may have been a bigger focus during our collection period. The instructions given to students to introduce the software from one year to the next were also not tightly controlled, so they may have made certain queries seem redundant. For example, it does not make much sense to ask a patient’s name if you have instructions telling you their name (note the large drop in the “What should I call you?” class). ChatScript pattern definitions can also change, which may introduce a bias for some labels, given the annotation methods. That is, if two responses are appropriate to a particular query, and a change to a template suddenly makes one occur more frequently than the other, the annotation methods will probably not flag that response as incorrect, and thus the true label will be annotated as whatever the response was.

We can examine accuracy of the hybrid system as a function of label frequency by quintiles, as we did above in the comparison of ChatScript and the stacked CNN; however, given the distributional differences between experiments, this raises the question of which label/quintile assignments to use. A comparison is presented in Figure 5. The first chart shows the accuracy according to the same quintile assignments as used in Experiment 1, and the second shows a full recalculation of the quintiles according to the data collected in the enhanced dataset. The first chart is broadly consistent with the previous experiment, although it shows a big drop in performance in the fourth-most-frequent quintile. That the trend is preserved is reassuring; the models generally exhibit consistent performance on the same semantic content. The second chart, however, shows an alarming drop in performance on the most frequent labels in the enhanced dataset. This is consistent with the drop in performance in the fourth quintile of the core dataset labels—the No Answer class is among the most frequent in the enhanced dataset and is almost always

Table 7. Adjusted accuracies of control and test conditions, and subcomponents of the combined system.

	Unseen labels omitted				Unseen and NoAns labels omitted			
	Control ($N = 2399$)		Test ($N = 2700$)		Control ($N = 2157$)		Test ($N = 2482$)	
	Acc%	NoAns%	Acc%	NoAns%	Acc%	NoAns%	Acc%	NoAns%
Oracle	74.70	10.38	85.63	5.15	81.08	9.55	91.42	3.87
ChatScript	74.70	10.38	74.89	11.96	81.08	9.55	79.73	11.28
Stacked CNN	—	—	70.04	0.04	—	—	76.15	0.00
Hybrid System	—	—	80.33	6.19	—	—	85.74	5.08

incorrectly identified. In the core dataset, No Answer occurred in the fourth quintile, explaining the performance drop for that data according to the core dataset quintile assignments. If we omit the No Answer class and other unseen classes from the analysis but otherwise assign quintiles according to label frequency in the enhanced dataset, we see a performance profile generally consistent with that shown in Figure 4 in the previous experiment, suggesting that the major distributional differences between the datasets are fairly localized to the occurrence of those few unseen classes.

Given the expected effects of the distributional differences discussed above, we here consider the results when omitting certain examples, to illustrate how much those question classes drive the difference in absolute performance from the previous experiment. Primarily we want to see how the system performs when omitting unseen classes (including the Negative Symptoms class) from the result set, and see the magnitude of the effect of the No Answer class, given its huge change in frequency. These results are given in Table 7. While the overall performance of the hybrid system under the reasoned adjustments does not reach the same level as the prior experiment, the control experiment is much more consistent with it, and it is very plausible to attribute the remainder of the difference in the stacked CNN's performance to the other distributional differences that manifested in the enhanced dataset.

One curious aspect of the results in Table 7 is that when omitting the No Answer class, ChatScript in the control condition outperforms ChatScript in the test condition, where it had trailed when including all data. This mostly follows from the fact that the No Answer class is a larger proportion of the CS-only (control) dataset.

Another available measure of the quality of the systems is the number of times a user has to rephrase a query to get a relevant answer. A natural response to a misunderstood question, whether it manifests as an incoherent reply or a request to rephrase, is to try again; anecdotally, we indeed find that users most often follow this pattern. Thus, it is easy to estimate performance in this regard, by simply finding contiguous queries with the same labels in the dataset. Doing so, we find 249 repeated queries in the control set of 2497 turns and 209 repeated queries in the test set of 2799 turns. This turns out to be a statistically significant reduction in repeated queries due to the hybrid system ($p = 1.9 \times 10^{-6}$, Pearson's χ^2). This may explain some of the difference in raw performance of the ChatScript subsystem between the control and test conditions.

6. Experiment 3: Retrospective out-of-scope improvements

The analysis of the previous experiment offers a number of promising avenues for improving the system going forward. The most obvious deficit that it brought to light is that out-of-scope

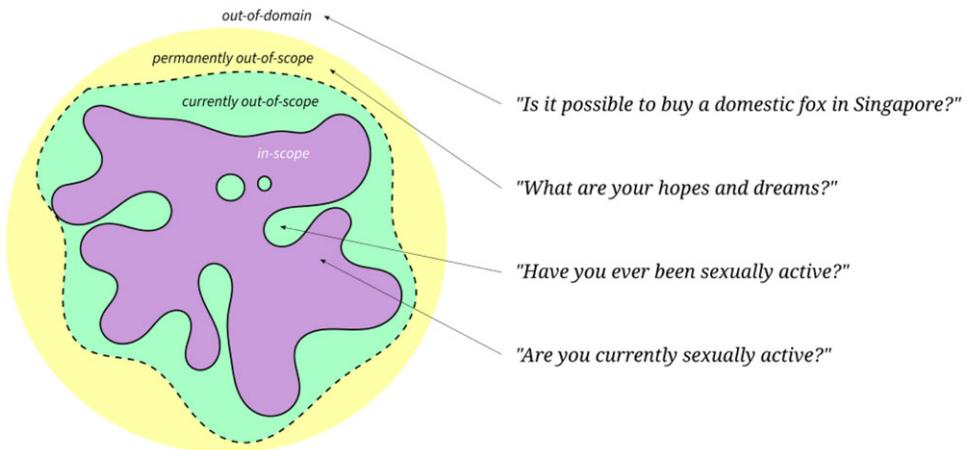


Figure 6. A simple illustration of scope and domain boundaries, with specific examples.

questions are far more frequent than the core dataset reflects. Furthermore, while we had assumed that a response of “I don’t understand that question” was always, in some sense, incorrect, it is in fact an appropriate response from a back pain patient to a query about, for example, their greatest strengths. The data collected also provide an opportunity to improve overall performance while examining the interplay between the components of the hybrid system in the face of additional training data.

In analyzing the results with our domain expert, we discovered that there were subtle but important distinctions in the types of questions currently unanswered by the system. The easier distinction to draw is between *in-domain* and *out-of-domain* data, which is a well-explored problem in dialogue systems; for the latter type of question, a default “I don’t understand” answer suffices. However, there is a more subtle set of questions that are relevant to the medical domain, but currently not covered by the content in the system; we deem these *out-of-scope*. Some types of questions were determined to be not germane to the particular patient case and were intentionally left to the default answer (permanently out-of-scope), whereas other questions were relevant to the case (currently out-of-scope).

The model changes that we propose in this section are mainly focused on improving performance for out-of-scope queries, that is, the No Answer class, which we are able to do with limited success. The problem of accurately identifying out-of-scope queries is especially challenging for several reasons. For one, the distinction between in- and out-of-scope is sometimes fairly arbitrary in our case, having little to do with the inherent semantics of the question. For example, a patient’s marriage status can be quite relevant to clinical outcomes, and our virtual patient can accurately answer the question “Are you married?” with the response “I’m single.” However, the question “Have you ever been married?”—implying the possibility of that single status being due to divorce, another clinically relevant status—was deemed by the medical content author to not be worth the development cost to distinguish, due to subtle syntactic differences, difficult polarity issues, etc., that can arise in various paraphrases of the two questions. Thus, the boundary between in- and out-of-scope queries can be quite narrow and convoluted in our application domain. A simple illustration of the relationships between scope and domain boundaries can be seen in Figure 6. Furthermore, the in-scope (meta-)class comprises a few hundred distinct query classes, making it fairly heterogeneous. Thus, it is harder to distinguish from out-of-scope data, particularly given the relatively small size of the dataset and the dearth of adequately similar out-of-scope data.

Text-based techniques utilizing unlabeled data augmentation might be appealing for our use case, except for the aforementioned narrow, convoluted boundary between in- and

out-of-scope data. We expect that adequately characterizing that boundary would require a large amount of “in-domain, but currently out-of-scope” data, and such data are hard to come by. Available datasets are too easy to distinguish from the virtual patient data simply on the basis of obvious syntax and vocabulary features. Techniques depending on a robust characterization of the distribution of the positive class are similarly expected to perform poorly, due to the heterogeneity and relatively small size of the positive data, also mentioned above. For these reasons, we attempt to improve recognition of the No Answer class largely within our existing framework, by treating it as just another query class and augmenting our dataset with more representative examples. We recognize that more sophisticated one-class classification techniques may further improve performance over what we show here, but we generally consider this work the establishment of a baseline for further research.

Thus, we propose some simple changes to our existing system with the aim of improving our handling of the No Answer class, which we can evaluate with the enhanced dataset. The division of the enhanced dataset into the CS-only and hybrid portions leaves us with a natural partition to use for testing, and we use the hybrid dataset to evaluate our model changes. This leaves us with the CS-only dataset to explore the benefits of additional training data in our hybrid system, which makes the training data more reflective of the distribution of classes seen in the test set. We call the union of the CS-only and core datasets the *augmented dataset*.

Architecturally, we focus our attention on the chooser component of the system. We reason that since the No Answer class is a negative class—defined only by the lack of relevant content—that the stacked CNN is likely to fail to generalize to the full variability of the class, absent some sophisticated training techniques. The chooser, on the other hand, has access to statistics produced by the stacked CNN and may more easily be able to recognize the situation that the stacked CNN is not highly confident in any particular positive class.

Concretely, we extend the binary chooser to a three-way choice between the response produced by ChatScript, that of the stacked CNN, or the No Answer class, experimenting with two different labeling schemes. We also explore additional features for input to the chooser.

6.1. Models

We implement two variations on a three-way chooser and evaluate their performance trade-offs. The first is a fairly straightforward extension of the original model to a multiclass setup, while the second employs a multilabel setup. The multilabel model was developed in recognition of the fact that sometimes multiple choices are equally acceptable, since either the input models or the chooser can produce an appropriate No Answer response. We use the same logistic regression model from SciKit-learn (Pedregosa *et al.* 2011) as used in the previous experiment, using the built-in one-vs-rest classification scheme to handle the three-way multiclass classification, and using a binary relevance strategy (Boutell *et al.* 2004) in the multilabel case, via the `OneVsRestClassifier` class. We use LIBLINEAR (Fan *et al.* 2008) to solve. Details of training and model evaluation are described below.

We experiment with the addition of new input features and particular combinations of them. In general, the new features attempt to infer additional information about ChatScript’s “opinion” of the stacked CNN’s output. As discussed in Section 4, ChatScript does not output a probability distribution for its possible responses, so any kind of measurement of the confidence ChatScript has in its response, or possible alternative responses, must be inferred in some way. ChatScript decides its output by searching over possible responses, and returning the first one that matches. The search order is determined implicitly, by ChatScript’s search priorities and the author’s organization of the patterns. For example, a pattern within the current topic, as the author defines the topic, will always match before a pattern in another topic. Each *pattern* is essentially a regular expression that may match an input sentence; we call a set of patterns that all produce the

Table 8. New features used by the three-way chooser.

Feature	Description
CS ratio	The ratio of the score of the chosen response to the sum of all scores, i.e. $\frac{s_0}{\sum_j s_j}$.
CNN agreement	The score s_j corresponding to the class chosen by the stacked CNN, if it is present in the list of matches. Otherwise, zero.

same response a *class*,^d consistent with the definition of a class learned by the stacked CNN, after the collapse of certain semantically similar classes; and a *topic* is an author-defined collection of notionally related classes. In our case, one topic is the patient's *social history*. This includes classes such as "Do you smoke," "Are you in a relationship," "What do you do for a living," or "Who supports you?" The relationship status class may be defined by several patterns; one such pattern may focus on the keyword "single" in certain syntactic constructions, while a different pattern may focus on the keyword "relationship" in different syntactic contexts.

One source of additional information that can be made available from ChatScript is a ranked list of all of the patterns that would have matched the input, if the search had not stopped after the first hit. This includes multiple patterns within the same class. Since patterns and classes are manually authored, we reason that a class with more patterns should usually be more "mature," in that more time has been spent developing it. In other words, a larger number of edge cases in the language that can represent the same semantics have been identified and encoded as patterns. For example, a large variety of syntactic constructions should be interpreted as paraphrases of the question "Is the pain constant," including, "Does the pain wax and wane," "Does the pain ever fluctuate in intensity," "Is it continuous that you have this pain," etc. It takes a lot of rules to adequately cover this variety of language, and for the classes where many such rules are written, we expect the coverage to be better. Accordingly, we expect ChatScript to be more confident about a match on a pattern within a mature class, and even more confident when there are multiple matches to the same class. Finally, due to ChatScript's inbuilt heuristic priorities, we expect higher ranked matches to correspond to higher confidence. Note that the No Answer class is always the last item in the list.

Given all of the above, we assign a score to each class that is just the normalized sum of the inverse ranks of pattern matches for that class. That is, we assign a score s_j such that, for the 1-indexed sequence of patterns matching the input $M = (p_i)$ containing the set of classes $\{c_j\}$, and the set $P_j = \{i: C(p_i) = c_j\}$, where $C(p_i)$ is the class that pattern p_i belongs to:

$$s_j = \frac{\sum_{i \in P_j} \frac{1}{i}}{\sum_{i \in M} \frac{1}{i}} \quad (2)$$

For notational convenience, we denote the score for the template chosen by ChatScript as s_0 .

Given Equation (2), the new features are summarized in Table 8. Additional features were explored but did not show a benefit during development.

6.2. Training and evaluation

While the modifications of the chooser models themselves are straightforward uses of off-the-shelf software, specific aspects of the training of the models merit detailed enumeration.

Besides augmenting the training data with the extra examples from the CS-only dataset, we add two labels, one each for *Unknown* classes and *Negative Symptoms*. These are largely to facilitate evaluation of the features which measure agreement between the stacked CNN and ChatScript

^dChatScript documentation refers to this as a *template*.

when ChatScript produces these responses, but the added examples include items belonging to the new classes. The Negative Symptoms class, while being incompatible with the early paraphrase identification approach, could reasonably be handled by the direct classification approach, due to similarities in questions relating generically to medical conditions and anatomy; treating all unseen classes as a single class for purposes of training the stacked CNN, however, is more problematic. Ideally, the chooser should learn to defer to ChatScript when it produces a class unseen by the stacked CNN, but this is a hypothesis to be verified, since the behavior of the stacked CNN when trained on these examples is less predictable.

We retrain the stacked CNN on the augmented data using the same 10-fold cross-validation scheme, with new splits to accommodate the extra data. We collect statistics for calculating the chooser's training input features from the predictions collected from the cross-validation. Chooser inputs for the test set are calculated from a single stacked CNN model trained on the entire augmented training set, where the test set of course was unseen during the retraining. We encode the labels for training the chooser using different conventions, depending on which three-way variant we are training.

In the case of the *exclusive* configuration, which refines the earlier approach, each example is labeled as exactly one of ChatScript, stacked CNN, or No Answer being the correct choice. All instances of the No Answer class label are labeled with the No Answer choice for the chooser, regardless of whether or not either ChatScript or the stacked CNN correctly identified it. All examples in which none of the available choices would be correct are labeled as ChatScript being the correct choice, to most take advantage of the hybrid system's ability to handle classes unseen by the stacked CNN.

In the case of the *multilabel* three-way variant, every label is a binary vector of length three, with the separate dimensions indicating independently which of the three choices provides an acceptable answer. Thus, if ChatScript provided an accurate No Answer response, then the dimensions for both ChatScript and No Answer would be set to one. In contrast to the exclusive setup, if no choice is correct, we leave the example unlabeled (the zero vector), based on development performance. To evaluate the multilabel setup at test time, we select the label with the maximum probability predicted by the chooser as the single model output.

As with the baseline binary chooser, we only train and test on examples where the component systems disagree, always taking agreed-upon classes where they exist. We use 10-fold cross-validation on the augmented dataset to measure development performance to find the best configuration of each three-way variant for testing. Baseline development performance is taken as the weighted average of both a 10-fold cross-validation result of the two-way model trained on the core dataset, and the performance of a single model trained on all of the core dataset and evaluated on the CS-only dataset. In other words, baseline development performance is a cross-validation result on the entire augmented training set, but where training folds are only comprised of the core dataset.

We test on the entirety of the hybrid dataset. As our primary evaluation metric we report class accuracy. Note that this is not the accuracy of the three-way choice but of the class returned by the chosen system. Since our model adjustments are directed at improving the performance of the No Answer class, we also report the percentage of No Answer responses—irrespective of which system produced them, and whether or not they were chosen correctly—as well as precision and recall on the No Answer class.

6.3. Baseline

A brief discussion of the baseline used for evaluation of the test set is warranted. In theory, this should be the same accuracy number as reported in Section 5.3, that is, 80.03% accuracy in the unadjusted case. However, for the same reasons that our reproduction in Section 4 gave slightly worse results than the original work (Jin *et al.* 2017), our baseline in the present experiment is

Table 9. Development accuracy with additional features.

System	Dataset	Base feats	+agree	+agree+ratio
Two-way baseline	Core	84.41	—	—
Two-way baseline	Augmented	84.85	84.73	84.72
Three-way exclusive	Augmented	84.92	84.98	84.99
Three-way multilabel	Augmented	85.70	85.83	85.83

Table 10. Effects of retraining on test performance.

	Core	Aug.
CS Acc	74.85	74.85
CNN Acc	67.27	73.17
Oracle	84.82	86.89

slightly lower than what was annotated in the prospective experiment. Between the bug fix, data corrections, and a change in splits resulting from a different random shuffle—see Gorman and Bedrick (2019) for an excellent discussion of the impact of splits on performance—we consider this a justifiable drop in performance without invalidating any conclusions. We further confirm that the changes result in different behavior of the chooser with a simple comparison of the frequency of the choices: the chooser in the prospective experiment chose the stacked CNN in 15.5% of turns in the test condition, while the baseline chooser for the present experiment chooses the stacked CNN in 10.8% of turns, using the same input sentences.

6.4. Results

Development results are shown in Table 9, with the best-performing features for each system (row) highlighted in bold font. Note that the absolute accuracy numbers are not directly comparable to test results due to different label distributions. In general, the additional features add very little to the performance, but we test the features that give the highest accuracy in development. The *CS Ratio* feature does not increase accuracy for the three-way multilabel system, but it does slightly increase F1 score on the No Answer class relative to the *CNN Agreement* feature alone (0.164 vs. 0.153), so we take that configuration as the best for running test results.

Retraining the stacked CNN on the augmented dataset unsurprisingly improves its performance on the test set, although the resulting boost in oracle performance does not match (see Table 10), implying that much of the improvement overlaps with examples that ChatScript was already answering correctly. In absolute terms, the stacked CNN performance under retraining is much closer to ChatScript as well, which is more consistent with the results from Section 4. This supports the claim that basic differences in the distribution of class labels were a source of the unexpected performance discrepancy observed in the prospective experiment.

Test results for the two three-way systems that were identified as having the best development performance are shown in Table 11. We also include the best-performing binary chooser when using the augmented training data, to isolate the effects of the training data relative to the baseline and the model enhancements. The retraining alone leads to a modest improvement in overall accuracy, with a small boost in recall on the No Answer class and an even smaller drop in precision. The model changes aimed at improving performance on the No Answer class introduce

Table 11. Test results for three-way choosers with retraining.

System	Training data	Accuracy	No Answer		
			Percent	Precision	Recall
Two-way baseline	Core	78.78	5.11	0.210	0.133
Two-way baseline	Augmented	79.64	7.34	0.199	0.183
Three-way exclusive +feats	Augmented	79.92	7.04	0.198	0.174
Three-way multilabel +feats	Augmented	81.10	4.14	0.328	0.174

some trade-offs. The exclusive setup improves over the baseline on recall, being generally more likely to produce the No Answer response, while the multilabel setup favors precision without improving recall over the exclusive setup. Both three-way systems exhibit a significant boost in accuracy over the two-way baseline, particularly the multilabel setup (exclusive: $p = 0.009$; multilabel: $p = 4.6 \times 10^{-6}$, McNemar's test), and given the No Answer recall numbers, in both cases this is due to increased performance on the positive classes. We show in the next section that the large increase in the multilabel system is due to the presence of the unlabeled examples in the case that no correct choice exists; forcing these examples to belong to a class during training serves mostly to confuse the true boundaries of that class.

We offer analyses to gain further insight into all of these results in the next section.

6.5. Discussion

Despite significant improvements in accuracy using the multilabel chooser, the improvements on the No Answer class can only be described as modest, at best. Some analysis illuminates the issue a bit better.

As we supposed above, the distinction between in- and out-of-scope proves to be hard to discover, and we can characterize the problem more fully with quantitative techniques. The most intuitive illustration comes from a t-SNE (Maaten and Hinton 2008) plot of the training data. T-SNE is a stochastic technique that maps high-dimensional data points into visualizably low-dimensional space, while attempting to preserve distances between points. By maintaining such distances through the mapping, the low-dimensional visualization should preserve some properties of the dataset as a whole, such as clusterings and class separability. Using this technique to visualize the chooser inputs can give a good intuition for the separability of the data, and therefore the feasibility of the three-way classification task. Figure 7 shows the result of applying t-SNE to those points in the chooser training data where ChatScript and the stacked CNN do not agree. The blue circles are inputs where the chooser should choose the CNN, the green circles are correct ChatScript choices, the red dots are No Answer choices, and the gray dots are instances where any choice gives an incorrect response. In terms of the visualization, the job of the chooser is to find a set of lines that separates the differently colored dots from each other. The simpler the set of lines, the more likely they can be reused to correctly classify unseen points. As discussed below, the figure illustrates why the multilabel approach yields an accuracy improvement, but it also shows that modifications of the chooser are unlikely to ever result in significant improvements on the out-of-scope question.

The plot makes it apparent that the No Answer labels are thoroughly enmeshed throughout the data space. There is just no way for a logistic regression model to find a boundary which adequately and generalizably separates the red dots from the other data points. This confirms our intuition that the boundary around these points would be convoluted and narrow. Further, there are several

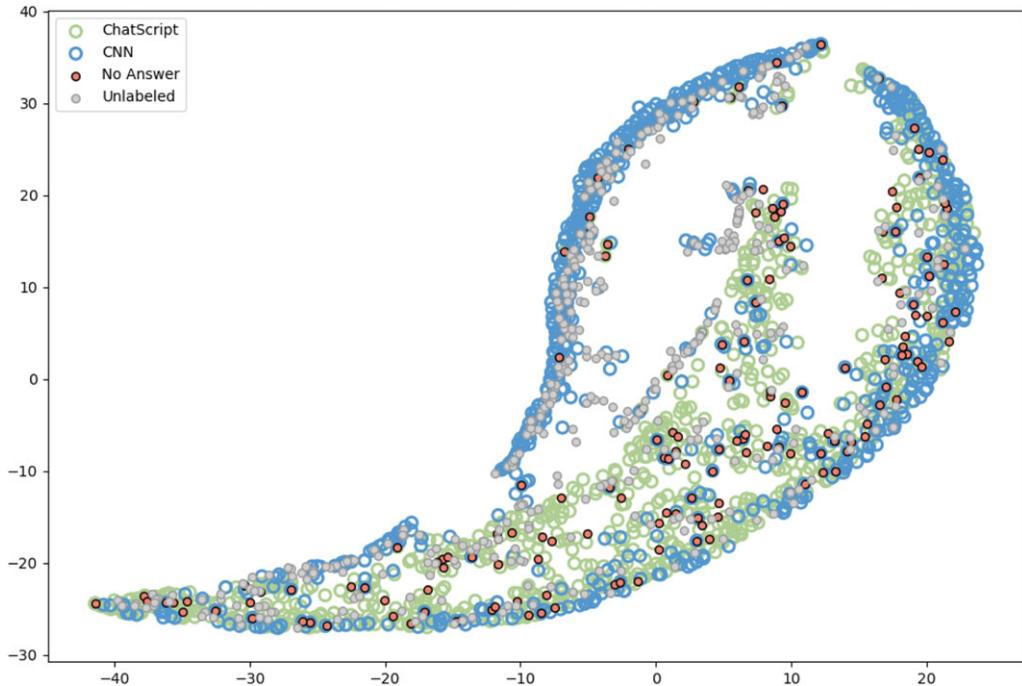


Figure 7. t-SNE plot of Multilabel chooser training data. Units are included to enable discussion of features of the visualization, but note that they are meaningless in terms of the original data, given that this is a stochastic, lossy projection of the original space.

No Answer points where the CNN produced a No Answer response (not explicitly identified in the figure), so the CNN choice would be correct. However, these are often buried in a region that is predominated by ChatScript choices. Finding a boundary to separate these points from the nearby ChatScript choices is no easier if they are viewed as CNN choices or No Answer choices by the three-way chooser. Overall, this does a lot of the work to explain the limited improvements on the No Answer class.

The separation between CNN and ChatScript choices is much more encouraging, however. This clearly illustrates why the chooser provides such a significant accuracy boost; large regions full of high-purity clusters of CNN choices—that is, the curved bands of blue points from approximately $(-10, -10)$ to $(10, 35)$, and from $(15, 30)$ to $(15, -10)$ —are easy to separate from the ChatScript choices. We can also clearly see why not labeling points that have no correct choice (i.e., the labeling scheme in the multilabel scenario) provide such a big benefit for the overall accuracy. Many of these unlabeled points (gray dots in the figure) cluster together with the otherwise very pure cluster of CNN data points (the band from $(-10, -10)$ to $(10, 35)$). If these are treated as ChatScript labels, as would be the case in the exclusive condition, this easily separable region of the space becomes much more confusing for the classifier, which in turn damages the model's generalization to the test set.

While the multilabel scheme creates higher accuracy by allowing the chooser to trust the stacked CNN more in the region where it is accurate, the inevitable outcome is that the system will choose the CNN on many of the occasions when neither subsystem is correct. This revisits the trade-off that was introduced earlier: a boost in accuracy also comes with more incorrect replies, instead of the presumably lower risk of a No Answer response. To quantify the trade-off, consider the major and minor errors introduced at the beginning of the paper: *major* errors, where the system replies to a question that was not asked, but which the user can potentially

interpret as the answer to the question they did ask; and *minor* errors, in which the system should have answered the question, but instead replied with “I don’t understand,” implying that the user should try again. While the three-way exclusive setup has the higher total error rate at 20.1%, its major errors are 14.4% compared to 16.1% in the multilabel case, which has the lower overall error rate at 18.9%. Accordingly, the minor errors are more than doubled in the exclusive case relative to the multilabel case.

Exactly how bad these errors are—especially with respect to the unlabeled examples that effectively end up as a CNN choice in the multilabel setup—is a more subjective question that depends on how the CNN actually interpreted the input. In one of the subjectively worst cases, a user asks, “Okay. So back pain and urinating more frequently. Is that all?” This should be interpreted as the class “Any other problems?” ChatScript incorrectly provides the No Answer response, and the retrained stacked CNN incorrectly interprets the question as “Have you noticed pain while urinating?” The programmed reply to this question is “I haven’t had any pain like that,” which could be construed as rejecting the back pain issue among the user’s summary of topics to discuss. In this case, the No Answer response is the preferable option. A more innocuous, and much more typical, example is the input “So it seems to get worse with prolonged use?” According to our annotation, this should be interpreted as “Does the pain improve with exercise?” ChatScript again provides the No Answer response, while the CNN interprets it as “Is the pain improving?” The multilabel chooser picks the CNN, and the reply then comes as “I don’t know if it is getting worse or not. It pretty much just hurts all the time.” The reply is mostly relevant, but it is also not adequately responsive to the question. This lack of coherence then is a sufficient cue to the user that they should try language more focused on the physical activity component of their question. Other similarly benign confusions include “How much ibuprofen do you take?” for “How much ibuprofen have you taken?” and “When did the pain start?” for “Did anything happen to cause the pain?” Looking at the erroneous differences between the exclusive and multilabel setups, our subjective impression is that the multilabel system is more helpful than harmful, suggesting that the so-called major errors quantified above are not usually catastrophic.

A majority of the No Answer responses in the multilabel condition (65%) come from agreement between the stacked CNN and ChatScript, although of these, 68% are incorrect. The chooser overrides both subsystems to provide 18% of the No Answer responses, with the remainder all coming from choosing the stacked CNN’s response. A No Answer response from ChatScript is never chosen. The No Answer responses from the stacked CNN are the most accurate, at 55%. The live subjects were surprisingly focused on questions about the patient’s employment, to a level of detail that was not clinically relevant. Accordingly, many such questions were out of scope, and trends in the chooser’s No Answer responses reflect this. Two of the three correct No Answer responses that came from the chooser override were about back pain symptoms at work, but more often, the chooser returned No Answer for valid employment questions, such as “Are you working?” The majority of the benefit to performance on the No Answer class, then, seems to come from an increased opportunity to trust the stacked CNN’s determination of a No Answer response, along with the stacked CNN’s improved ability to detect out-of-scope questions through extra training data.

Finally, we note that even after retraining the stacked CNN with over 50% more data, the hybrid system still outperforms both the rule-based and data-driven components individually, reconfirming the benefit of combining both approaches in our relatively data-scarce domain. Figure 8 shows another breakdown of model test performance by label frequency quintiles, this time using quintile assignments derived from the label frequency of the augmented dataset used for training the multilabel model. Results from the binary chooser baseline replication are included for comparison. Again, the general trend of the stacked CNN outperforming ChatScript on high-frequency labels persists, and for the most part this translates to higher performance of the hybrid system than either of the subcomponents. Notably, though, the hybrid system underperforms the stacked CNN in the most frequent quintile. The large jump in the accuracy of the stacked

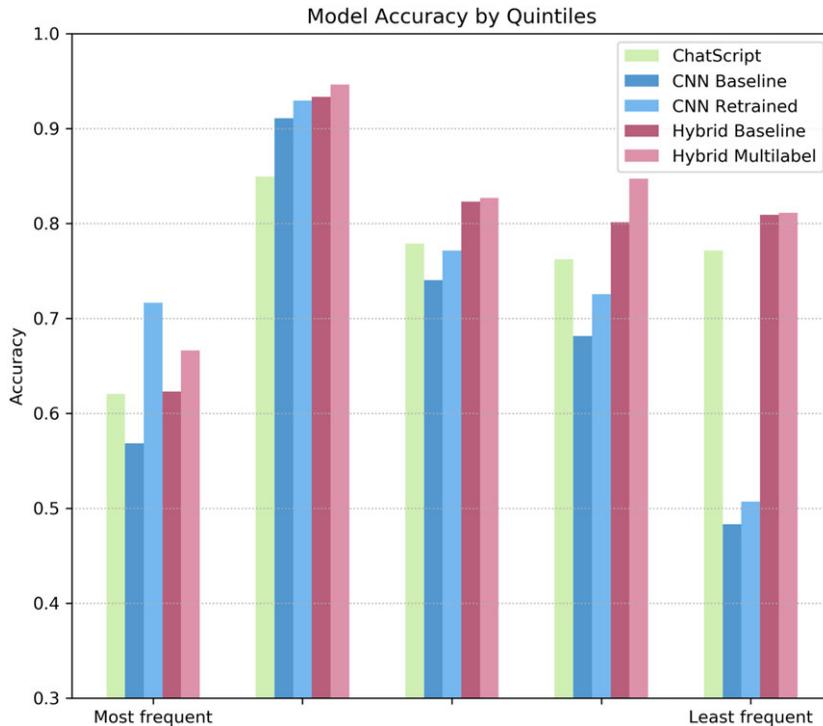


Figure 8. Quintile accuracies for multilabel components, with baselines.

CNN over its baseline in this quintile is mostly driven by a boost in accuracy on the No Answer class (about 36% vs. less than 2%), which remains difficult for the chooser to distinguish, as discussed at length above. We believe that the difficulty of the out-of-scope question does most of the work to explain this divergence from the otherwise consistent trend of the hybrid system outperforming either component. However, this may reflect a threshold of training data volume beyond which the stacked CNN is simply more effective on its own than in combination with ChatScript. Nonetheless, the benefit at lower frequencies remains very clear.

7. Future Work

While our alterations of the chooser demonstrated a modest improvement in F1 score on the out-of-scope class, further improvements will be a prime target for future research. The large improvement on out-of-scope questions by the stacked CNN when given adequate training data suggests that the components of the system with more direct access to the content of the question will be more successful than the chooser at handling this challenging problem. A multitask approach similar to Perera and Patel (2019) for training the stacked CNN seems promising, and we plan to explore it further.

Ongoing work to enhance the immersiveness of the interaction has extended the virtual patient to a spoken interface, which presents many additional technical challenges. The language used in spoken communication is quite different from typing, so evaluating how well models trained on text translate to speech will be important. Errors that occur are also quite different—for example, automatic speech recognition systems do not misspell words, but they may misrecognize audio—so developing ways to be robust to such errors is also a prime objective. Initial efforts in this direction have seen success by adapting the character models in the stacked CNN to use phonetic representations of the input (Stiff, Serai, and Fosler-Lussier 2019).

Since the semantic space of the virtual patient is focused on a specific domain but covers a large number of meanings within that domain, it presents unique opportunities to explore paraphrase phenomena. In particular, we are exploring various methods of paraphrase generation to augment our training set and improve performance for many of the rare classes in our dataset and have carried out initial experiments in this direction (Jin *et al.* 2018).

Our study concluded that part of the benefit from selecting between rule-based and machine-learned question classification is from rule-based systems helping in the infrequent classes. Indeed, when beginning to work with contextualized representations from pretrained models (e.g., Peters *et al.* 2018; Devlin *et al.* 2019), this disparity is exacerbated (Stiff, Song, and Fosler-Lussier 2020)—for example, fine-tuned Bidirectional Encoder Representations from Transformers models perform worse than the CNN models and are computationally expensive for real-time deployment with large student classes. More recent work has pointed to training methods that improve accuracy for low-frequency classes (Sunder and Fosler-Lussier 2021), but we still continue to see benefit in deployment of hybrid systems for virtual patients.

8. Conclusion

A holistic, prospective study of a dialogue system requires the system to be reasonably complete. Unexpected, unnatural, or omitted behavior by the agent can break users' immersion and make them hyper-aware that they are not conversing with a human. This can bias their language in ways that are not helpful to a researcher seeking a sincere evaluation of a designed system. And of course, dialogue systems are complicated. They require refinement from extensive feedback to *become* reasonably complete, which creates a bit of a chicken-and-egg situation.

Fortunately, the patient interview presents a domain that, through many carefully considered design decisions, allowed us to develop a fairly complete agent using relatively few resources. And yet, that design process was an iterative one. So while an agent that can go through a robust validation is an accomplishment in itself, that validation step was necessary, to confirm that the iterative process arrived at a design that met the original goals. Our work here showed that, even under randomized prospective experimental protocols, our hybrid NLU component continued to exhibit superior accuracy over rule-based and data-driven systems individually.

However, this validation study also pointed to opportunities for further improvement. Our early work with paraphrase identification biased our subsequent approach to question classification, and while the assumptions made were reasonable in their isolated context, the prospective experiment presented here forced us to recognize important issues for live deployment that those assumptions obscured. In particular, the prevalence of out-of-scope queries was far greater than expected based on our early efforts. The work done here to address the issue certainly leaves room for improvement, but our analysis also demonstrates why this is a very difficult problem. The lesson here is not just to eliminate some particular source of bias in the development of a given dataset; in fact, we believe such biases are inevitable results of the process of developing and refining research questions. Rather, we emphasize the importance of the prospective validation of the previous retrospective results in a robust, controlled experiment.

The most important positive outcome of the experiments presented here was reinforcing the benefit of the hybrid approach to NLU in our data-limited domain. The power of the hybrid approach repeatedly proves to be the complementary performance of the rule-based and data-driven components by label frequency, even to the point that we see some success with labels that are entirely unseen by the data-driven model. Having a baseline level of performance on unseen classes offers content authors highly desirable flexibility in adding new classes in the deployed system. A further positive result of our experiments within the hybrid framework was the insight not to force a choice during training if both rules and data fail to find a correct answer. Even though it was born out of an analysis aimed at understanding the difficulty of the out-of-scope problem, shedding light on the shape of the data in the chooser's input space resulted in a more informed approach to the hybrid system.

The analysis presented here suggests that the out-of-scope distinction is something of a separate task from the question classification task we began with and is somewhat different than the typical out-of-domain task that dialogue systems face. This may suggest either making the determination prior to the question classification or parallel with it. Corresponding architectural adjustments may involve joint tasks for the stacked CNN or end-to-end neural models that implicitly incorporate the chooser by conditioning the stacked CNN's response on ChatScript's output. We leave these explorations for future work.

Acknowledgments. We are grateful to the reviewers for their thoughtful feedback, which improved this manuscript substantially. This material is based upon work supported by the National Science Foundation under Grant No. 1618336. We gratefully acknowledge the support of NVIDIA Corporation with the donation of a Quadro P6000 GPU used for portions of this research. This project was also supported by funding from the United States Department of Health and Human Services Health Resources and Services Administration (HRSA D56HP020687), and the Medicaid Equity Simulation Project. The Medicaid Equity Simulation Project is funded by the Ohio Department of Medicaid and administered by the Ohio Colleges of Medicine Government Resource Center. The views expressed in this publication are solely those of the authors and do not represent the views of the state of Ohio or any U.S. federal funding agencies.

Competing interests. The authors declare none.

References

- Bird S., Klein E. and Loper E. (2009). *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, Inc.
- Boutell M. R., Luo J., Shen X. and Brown C.M. (2004). Learning multi-label scene classification. *Pattern Recognition* 37(9), 1757–1771.
- Bradeško L. and Mladenčić D. (2012). A survey of chatbot systems through a loebner prize competition. In *Proceedings of Slovenian Language Technologies Society Eighth Conference of Language Technologies*. Slovenian Language Technologies Society, pp. 34–37.
- Campillos-Llanos L., Bouamor D., Zweigenbaum P. and Rosset S. (2016). Managing linguistic and terminological variation in a medical dialogue system. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. European Language Resources Association, pp. 3167–3173.
- Campillos-Llanos L., Rosset S. and Zweigenbaum P. (2017). Automatic classification of doctor-patient questions for a virtual patient record query task. In *BioNLP 2017*. Association for Computational Linguistics, pp. 333–341.
- Campillos-Llanos L., Thomas C., Bilinski É., Neuraz A., Rosset S. and Zweigenbaum P. (2021). Lessons learned from the usability evaluation of a simulated patient dialogue system. *Journal of Medical Systems* 45(7), 1–20.
- Campillos-Llanos L., Thomas C., Bilinski É., Zweigenbaum P. and Rosset S. (2019). Designing a virtual patient dialogue system based on terminology-rich resources: Challenges and evaluation. *Natural Language Engineering* 26(2), 1–38.
- Chiu B., Crichton G., Korhonen A. and Pysalo S. (2016). How to train good word embeddings for biomedical nlp. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*. Association for Computational Linguistics, pp. 166–174.
- Collobert R., Weston J., Bottou L., Karlen M., Kavukcuoglu K. and Kuksa P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, 2493–2537.
- Danforth D., Price A., Maicher K., Post D., Liston B., Clinchot D., Ledford C., Way D. and Cronau H. (2013). Can virtual standardized patients be used to assess communication skills in medical students. In *Proceedings of the 17th Annual IAMSE Meeting, St. Andrews, Scotland*. International Association of Medical Science Educators.
- DeVault D., Georgila K., Artstein R., Morbini F., Traum D., Scherer S., Rizzo A.S. and Morency L.-P. (2013). Verbal indicators of psychological distress in interactive dialogue with a virtual human. In *Proceedings of the SIGDIAL, 2013 Conference*. Association for Computational Linguistics, pp. 193–202.
- DeVault D., Leuski A. and Sagae K. (2011a). An evaluation of alternative strategies for implementing dialogue policies using statistical classification and rules. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP)*. The Association for Computer Linguistics, pp. 1341–1345.
- DeVault D., Sagae K. and Traum D. (2011b). Incremental interpretation and prediction of utterance meaning for interactive dialogue. *Dialogue and Discourse* 2(1), 143–170.
- Devlin J., Chang M.-W., Lee K. and Toutanova K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, pp. 4171–4186.

- Fan R.-E., Chang K.-W., Hsieh C.-J., Wang X.-R. and Lin C.-J. (2008). Liblinear: A library for large linear classification. *Journal of Machine Learning Research* 9, 1871–1874.
- Fielding R.T. (2000). *Rest: Architectural Styles and the Design of Network-Based Software Architectures*. Doctoral dissertation, University of California.
- Glass J. (1999). Challenges for spoken dialogue systems. In *Proceedings of the 1999 IEEE ASRU Workshop*, vol. 696. Cambridge, MA: MIT Laboratory for Computer Science.
- Glorot X. and Bengio Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 9. JMLR Proceedings, pp. 249–256.
- Gorman K. and Bedrick S. (2019). We need to talk about standard splits. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 2786–2791.
- Jaffe E., White M., Schuler W., Fosler-Lussier E., Rosenfeld A. and Danforth D. (2015). Interpreting questions with a log-linear ranking model in a virtual patient dialogue system. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, pp. 86–96.
- Jin L., King D., Hussein A., White M. and Danforth D. (2018). Using paraphrasing and memory-augmented models to combat data sparsity in question interpretation with a virtual patient dialogue system. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, pp. 13–23.
- Jin L., White M., Jaffe E., Zimmerman L. and Danforth D. (2017). Combining cnns and pattern matching for question interpretation in a virtual patient dialogue system. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, pp. 11–21.
- Khan S.S. and Madden M.G. (2009). A survey of recent trends in one class classification. In *Irish Conference on Artificial Intelligence and Cognitive Science*. Springer, pp. 188–197.
- Khatri C., Hedayatnia B., Khatri C., Venkatesh A., Nunn J., Pan Y., Liu Q., Song H., Gottardi A., Kwatra S., Pancholi S., Cheng M., Chen Q., Stubel L., and Gopalakrishnan K. (2018). Advancing the state of the art in open domain dialog systems through the alexa prize. In *Alexa Prize SocialBot Grand Challenge 2 Proceedings*. Amazon.
- Kim J.-K. and Kim Y.-B. (2018). Joint learning of domain classification and out-of-domain detection with dynamic class weighting for satisficing false acceptance rates. In *Proc. Interspeech 2018*, pp. 556–2560.
- Kim Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pp. 1746–1751.
- Kittur A., Chi E.H. and Suh B. (2008). Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 453–456.
- Li J., Monroe W., Shi T., Jean S., Ritter A. and Jurafsky D. (2017). Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 2157–2169.
- Liu B., Lee W.S., Yu P.S. and Li X. (2002). Partially supervised classification of text documents. In *ICML*, vol. 2. Citeseer, pp. 387–394.
- Luo Y.-F., Henry S., Wang Y., Shen F., Uzuner O. and Rumshisky A. (2020). The 2019 n2c2/umass lowell shared task on clinical concept normalization. *Journal of the American Medical Informatics Association* 27(10), 1529–e1.
- Maaten L. and Hinton G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research* 9, 2579–2605.
- Maicher K.R., Zimmerman L., Wilcox B., Liston B., Cronau H., Macerollo A., Jin L., Jaffe E., White M., Fosler-Lussier E., Schuler W., Way D.P., and Danforth D.R. (2019). Using virtual standardized patients to accurately assess information gathering skills in medical students. *Medical Teacher* 41(9), 1–7.
- Mikolov T., Sutskever I., Chen K., Corrado G.S. and Dean J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26 (NIPS 2013)*. Curran Associates, Inc., pp. 3111–3119.
- Morbini F., DeVault D., Sagae K., Gerten J., Nazarian A. and Traum D. (2014). Flores: A forward looking, reward seeking, dialogue manager. In *Natural Interaction with Robots, Knowbots and Smartphones*. New York: Springer, pp. 313–325.
- Morbini F., Forbell E., DeVault D., Sagae K., Traum D. and Rizzo A. (2012). A mixed-initiative conversational dialogue system for healthcare. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Association for Computational Linguistics, pp. 137–139.
- Nair V. and Hinton G.E. (2010). Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML 2010)*, vol. 3. International Machine Learning Society, pp. 807–814.
- Paszke A., Gross S., Chintala S., Chanan G., Yang E., DeVito Z., Lin Z., Desmaison A., Antiga L., and Lerer A. (2017). Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*.
- Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D., Brucher M., Perrot M., and Duchesnay E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12, 2825–2830.
- Pennington J., Socher R. and Manning C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pp. 1532–1543.

- Perera P. and Patel V.M.** (2019). Learning deep features for one-class classification. *IEEE Transactions on Image Processing* 28(11), 5450–5463.
- Peters M., Neumann M., Iyyer M., Gardner M., Clark C., Lee K. and Zettlemoyer L.** (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237.
- Porter M.F.** (2001). Snowball: A language for stemming algorithms. Available at <http://snowball.tartarus.org/texts/introduction.html> (accessed 5 August 2022).
- Pradhan S., Elhadad N., South B.R., Martinez D., Christensen L., Vogel A., Suominen H., Chapman W.W. and Savova G.** (2015). Evaluating the state of the art in disorder recognition and normalization of the clinical narrative. *Journal of the American Medical Informatics Association* 22(1), 143–154.
- Ram A., Prasad R., Khatri C., Venkatesh A., Gabriel R., Liu Q., Nunn J., Hedayatnia B., Cheng M., Nagar A., King E., Bland K., Wartick A., Pan Y., Song H., Jayadevan S., Hwang G. and Pettigree A.** (2018). Conversational AI: The science behind the alexa prize. *arXiv preprint arXiv:1801.03604*.
- Ravichandran D., Hovy E. and Och F.J.** (2003). Statistical qa-classifier vs. re-ranker: What's the difference? In *Proceedings of the ACL, 2003 Workshop on Multilingual Summarization and Question Answering-Volume 12*. Association for Computational Linguistics, pp. 69–75.
- Robinson S., Traum D., Ittycheriah M. and Henderer J.** (2008). What would you ask a conversational agent? Observations of human-agent dialogues in a museum setting. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco: European Language Resources Association (ELRA).
- Schölkopf B., Williamson R.C., Smola A.J., Shawe-Taylor J. and Platt J.C.** (2000). Support vector method for novelty detection. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., pp. 582–588.
- Srivastava N., Hinton G., Krizhevsky A., Sutskever I. and Salakhutdinov R.** (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15, 1929–1958.
- Stiff A., Serai P. and Fosler-Lussier E.** (2019). Improving human-computer interaction in low-resource settings with text-to-phonetic data augmentation. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 7320–7324.
- Stiff A., Song Q. and Fosler-Lussier E.** (2020). How self-attention improves rare class performance in a question-answering dialogue agent. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Association for Computational Linguistics, pp. 196–202.
- Stoyanchev S., Liu A. and Hirschberg J.** (2014). Towards natural clarification questions in dialogue systems. In *AISB Symposium on Questions, Discourse and Dialogue*, vol. 20. The Society for the Study of Artificial Intelligence and Simulation of Behavior.
- Sunder V. and Fosler-Lussier E.** (2021). Handling class imbalance in low-resource dialogue systems by combining few-shot classification and interpolation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 7633–7637.
- Talbot T.B., Sagae K., John B. and Rizzo A.A.** (2012). Sorting out the virtual patient: How to exploit artificial intelligence, game technology and sound educational practices to create engaging role-playing simulations. *International Journal of Gaming and Computer-Mediated Simulations (IJGCMS)* 4(3), 1–19.
- Tax D.M. and Duin R.P.** (1999). Support vector domain description. *Pattern Recognition Letters* 20(11-13), 1191–1199.
- Traum D., Aggarwal P., Artstein R., Foutz S., Gerten J., Katsamanis A., Leuski A., Noren D. and Swartout W.** (2012). Ada and grace: Direct interaction with museum visitors. In *International Conference on Intelligent Virtual Agents*. Springer, pp. 245–251.
- Triola M., Feldman H., Kalet A., Zabar S., Kachur E., Gillespie C., Anderson M., Griesser C. and Lipkin M.** (2006). A randomized trial of teaching clinical skills using virtual and live standardized patients. *Journal of General Internal Medicine* 21(5), 424–429.
- Wilcox B.** (2019). Chatscript. Available at <https://github.com/ChatScript/ChatScript> (accessed 23 July 2019).
- Wilcox B. and Wilcox S.** (2013). Making it real: Loebner-winning chatbot design. *ARBOR Ciencia, Pensamiento y Cultura* 189(764), 1–13.
- Wolpert D.** (1992). Stacked generalization. *Neural Networks* 5(2), 241–259.
- Yu H.** (2005). Single-class classification with mapping convergence. *Machine Learning* 61(1-3), 49–69.
- Yu H., Han J. and Chang K.C.-C.** (2004). Pebl: Web page classification without negative examples. *IEEE Transactions on Knowledge & Data Engineering* 1, 70–81.
- Yu H. and Sable C.** (2005). Being erlang shen: Identifying answerable questions. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence on Knowledge and Reasoning for Answering Questions*. Citeseer, pp. 6–14.
- Zeiler M.D.** (2012). *ADADELTA: An Adaptive Learning Rate Method*. CoRR. ACM.
- Zhao T. and Eskenazi M.** (2018). Zero-shot dialog generation with cross-domain latent actions. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*. Association for Computational Linguistics, pp. 1–10.
- Zipf G.K.** (1949). *Human Behavior and the Principle of Least Effort*. Cambridge, MA: Addison-Wesley Press.

Appendix A: System Architecture

The virtual patient is deployed as a simple client/server architecture, with the client bearing responsibility for user interface functions, including rendering the animated image of the patient to the user, displaying responses, and passing the user's queries to the back-end server. The server, on the other hand, houses the apparatus for actually interpreting the user's query, formulating a response, tracking the state of the conversation to the extent that such tracking is done, storing data for later analysis, and automatic assessment of student performance. This separation of concerns may be obvious to a software engineer, but we note that it offers relatively modular handling of anticipated future work, including a spoken user interface. An overview diagram is shown in Figure A1.

The client software is developed in Unity 3D,^e a multi-platform 3D game development engine. For this experiment, the virtual patient client application was built and deployed in a WebGL format^f and hosted on an Apache web server running on a university computer. This enabled students to access the virtual patient anywhere that they had access to a web browser and a keyboard. Unity makes it easy to deploy the same app for different execution environments, for example, tablet computers, so here again we have flexibility to meet anticipated future needs. 3D models were acquired from the Unity Asset Store^g and animated using Autodesk Maya.^h The patient can express a range of emotions, and these are controlled with messages from the back-end service that are sent concurrently with linguistic replies to user queries. After providing their name for academic assessment purposes, students can interact with the patient by simply typing a question. When they are finished with the interview, they can click a button on the screen to receive a transcript of their session, including feedback about questions that should have been asked. This transcript comes in the form of a PDF file that is generated dynamically in the browser with JavaScript, using information provided from the back-end server. Students also have access to an instructional PDF, hosted statically on the same server as the WebGL application.

As stated above, while the client application bears responsibility for displaying content to the user, the back-end service is generally responsible for deciding what that content is. Noting that the typewritten chatbot paradigm imposes a regular synchronous request/response usage pattern, this back end is designed as a typical REST-like web application (Fielding 2000), communicating over HTTPS. We note that the web service is not actually RESTful, because it maintains a fair amount of state information about each user's conversations with the virtual patient, and therefore, as currently implemented, does not properly scale beyond installation on a single server. Nonetheless, the single server has proven to have adequate capacity for its intended educational use; that is, up to 200 users over the course of a week, with a deadline-driven ramp up in concurrent usage. Despite not being truly RESTful, we made an effort to maintain correct usage of HTTP verbs, return codes, etc. with respect to state changes when designing the API.

At the time of the experiment described in Section 5.2, the web service implemented the API shown in Table A1. Only the specified verbs were implemented.

With limited personnel developing the back-end service, many design decisions were made with an eye toward minimizing development time. The stacked CNN and the hybrid system described in Section 4 were implemented in PyTorch (Paszke *et al.* 2017) and SciKit-learn (Pedregosa *et al.* 2011), respectively. Therefore, when selecting a framework for deployment of the back-end web service, a premium was placed on those with a Python programming interface. Among these, Flaskⁱ was selected for having a low programming overhead and being fairly lightweight. The Flask application was deployed within a gEvent^j container for efficiently handling

^e<https://unity.com/>

^f<https://www.khronos.org/webgl/>

^g<https://assetstore.unity.com/>

^h<https://www.autodesk.com/products/maya/overview>

ⁱ<https://palletsprojects.com/p/flask/>

^j<http://www.gevent.org/>

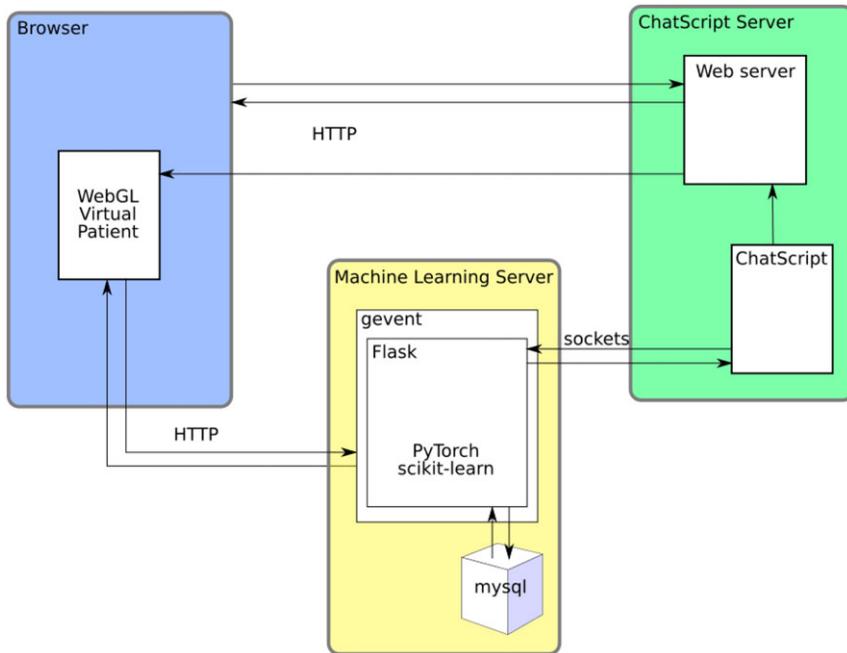


Figure A1. Overview of client-server architecture.

concurrent traffic with an event-based implementation. It was selected as a lighter-weight alternative to a more full-featured web server such as Apache httpd^k with installed modules, which was expected to require more maintenance effort. For purely logistical reasons, this Flask web service application and associated gEvent container were hosted on a separate university computer from the one hosting the Virtual Patient client WebGL application. This separate hosting did not introduce noteworthy latency effects but did require a Flask extension to enable cross-origin resource sharing. The server hosting the web service uses a six-core 2.4GHz AMD Opteron processor with 32GB of memory.

An important auxiliary function of the back-end service, beyond supporting the educational objectives of the virtual patient, is to collect data to support the present research and further improvements to the system. This function is fulfilled with a simple MySQL database instance (specifically, MariaDB^l), using only two tables. The web service interfaces with this database through the Flask-MySQLdb extension.

The web service must accept an input query from the user, decide between the ChatScript interpretation and the stacked CNN interpretation, and return the correct response, as coded by the expert content author within the ChatScript instance. The process for this decision making is explained in more detail below, but here we continue the overview of the web service by noting the organizational structure of the components of the process. The trained stacked CNN and chooser are simply embedded in the web service and are loaded into memory as soon as the service is started. The stacked CNN, despite being trained using a GPU, is configured within the service to perform computations using only the CPU. This is partially due to hardware availability but also based on the assumption that the overhead of loading data into the GPU memory would be greater than the benefit of parallel computation for single examples. Note, though, that we never

^k<https://httpd.apache.org/>

^l<https://mariadb.org/>

Table A1. The implemented REST-like API.

Endpoint	Description
/conversations/	POST: Create a new conversation resource with parameters provided in JSON post form. Return JSON containing opening greeting and URL for created resource.
/conversations/<number>/	GET: Show the metadata for the conversation specified by <number>
/conversations/<number>/query/	POST: Create a new query resource within the conversation specified by <number>, with the query text specified in JSON post form. Return JSON containing reply.

performed a rigorous evaluation of this assumption. The ChatScript instance runs in server mode in an independent process from the web service, and during this experiment, ran on the separate machine hosting the virtual patient client application. The back-end service makes calls to the ChatScript instance over raw socket connections.

The process for replying to a user query is as follows. When the user types a question into the client application, the text of the query, along with some metadata, is packaged into a JSON object and sent to the web service via an HTTP request. The web service unpacks the JSON object and forwards the query text to the ChatScript engine, which replies with what it believes to be the correct response. By design, ChatScript does not automatically provide any metadata along with this response text. Since responses are not unique to an interpretation of a query (e.g., yes/no questions), and because the interpretation is needed for the *CS_prob* feature for the chooser, we then send the debugging command `:why` to the ChatScript instance, which reveals information about the response, including the name of the template and the pattern that matched the input. Due to time constraints and the label collapse issue (see Section 3), during the experiment this metadata could only be used to determine whether or not ChatScript found a match for the query. This was confirmed to only have a minimal negative effect on accuracy with the core dataset (< 0.1%), since *CS_No_ans* is the most important feature for the chooser.

With the ChatScript response to the query adequately characterized, the same input is sent through the stacked CNN, and the output of both systems is sent to the chooser, as described in Section 4.5. If the two systems differ, and the chooser determines that the stacked CNN is the better choice, the canonical label sentence for the stacked CNN's chosen class is sent to the ChatScript instance. Thus, whenever the stacked CNN's response is chosen, ChatScript sees two volleys. With default settings, ChatScript is usually quite stateful, removing discussion topics that have already been seen, to prevent unnaturally repetitive responses; we configured the engine to repeat most answers an effectively unlimited number of times, so this double volley strategy is acceptable for those answers. For the few answers that do not accept unlimited repeats, we do not distinguish different replies in our evaluations.

Once the chosen response has been obtained, the synchronous HTTP response is returned to the client. Subjectively, the latency is noticeable compared to a system using only ChatScript but is not worse than the normal pace of a spoken conversation. Some of the latency is due to the double volley and network connection to the ChatScript server, but most of the wait is due to the computations of the stacked CNN.

One of the goals of the virtual patient is to provide rapid feedback to students about their interviewing technique, which we call *scoring*, although it might more accurately be called summary report generation. This is done by classifying queries by subject according to rules, and reporting the coverage of important subjects. In the present setup, this function is coded in the ChatScript

instance. Due to the way that scoring is implemented and the possibility of double volleys, the web service maintains a count of user queries, which must be passed to the ChatScript instance in parallel with the actual queries. Score summary reports are generated using JavaScript code hosted on the same machine hosting the virtual patient client. For readers wishing to build similar systems to the one we describe, our experience suggests that ChatScript may be best suited for dialogue management, and that such scoring analysis may be better suited to a different programming environment.

Finally, we note that the experiment detailed in Section 5.2 requires some users to converse with an agent implemented using ChatScript only; we include a per-conversation switch that allows the web service to bypass the stacked CNN and chooser when producing its response.

Appendix B: Question Classes

any changes in your diet
 • any chest pain • any family history of depression • any family history of hypertension
 • any family history of illness • any heart problems • any heart problems in your family • any otc medication
 • any other hospitalizations • any other past sexual partners • any other problems • any other sexual partners • any prescription medication • any previous heart problems • any prior surgeries • appetite • are you able to drive • are you able to sit • are you able to stand • are you abused • are you active • are you a new patient • are you currently working • are you current on vaccinations • are you dating anyone • are you depressed • are you exposed to secondhand smoke • are you frustrated • are you happy • are you having any other pain • are you healthy • are you in a relationship • are you married • are you mr. wilkins • are you nervous • are your grandparents living • are your muscles sore • are your parents healthy • are your parents living • are your relatives healthy • are your siblings healthy • are you sexually active • are you sick • are you stressed • are you

suicidal • are you sure • are you taking any medication • are you taking any medication for the pain • are you taking any other medication • are you taking any other medication for pain • are you taking supplements • can you care for yourself • can you do normal activities • can you eat • can you move around with the pain • can you move your back • can you point to the pain • can you rate the pain • can you run • describe the frequent urination • describe the furniture you were lifting • describe the pain • did anything happen to cause the pain • did i miss anything • did it happen at work • did the ibuprofen help • did the pain start immediately • did you fall • did you feel a sharp pain • did you find a parking spot • did you have any trouble finding us • did you hurt your back • did your doctor prescribe the medicine • did you take ibuprofen today • discussion is confidential • do an exam now • do any positions make the pain worse or better • does anyone in your family have back pain • does any position make the pain worse • does any position relieve the pain • does anything else help

the pain • does it hurt to move • does it hurt when not moving • does it hurt when you bend • does moving increase the pain • does the frequent urination keep you up at night • does the pain improve with exercise • does the pain increase when you stand • does the pain interfere with work • does the pain keep you up at night • does the pain radiate • does this bother you mentally • does your back hurt when you work • does your hip hurt • do you do any heavy lifting at work • do you drink • do you drink coffee • do you drink enough fluid • do you eat fast food • do you eat fruit • do you eat healthy food • do you exercise • do you feel anxious • do you feel hopeless • do you feel safe at home • do you feel well • do you have a cause for the pain • do you have a doctor • do you have a family • do you have a history of depression • do you have annual exams • do you have any allergies • do you have any bladder problems • do you have any bowel problems • do you have any children • do you have any chronic illnesses • do you have any hip pain • do you have any hobbies • do you have any medical

problems • do you have any pain • do you have any pain in your hip • do you have any pets • do you have any problems urinating • do you have any problems with your hip • do you have any questions • do you have any trouble sleeping • do you have any weakness • do you have a sexually transmitted infection • do you have a significant other • do you have a weapon • do you have exposure to chemicals • do you have frequent urination • do you have headaches • do you have loss of bowel or bladder control • do you have numbness or tingling • do you have pain anywhere else • do you have pain when you are at rest • do you have problems working • do you have relatives • do you have siblings • do you have support • do you have vision changes • do you live alone • do you mind if i call you jim • do you mind if I call you jim • do you need a note for work • do you prefer men or women • do your body-parts hurt • do you smoke • do you use any contraception • do you use illegal drugs • family history • good • goodbye • has the frequent urination become worse or better • has the frequent urination stopped you from doing anything • has the inability to work caused problems • has the intensity of the pain changed • has the pain affected your activity • has the pain become worse or better • has the pain stopped you from doing anything • has your weight changed • have you been able to work • have you been eating • have you been incontinent • have you been nauseous • have you been

off work • have you been resting • have you ever been dizzy • have you ever been in an accident • have you ever been in the hospital • have you ever been in the military • have you ever been pregnant before • have you ever had any chronic illnesses • have you ever had any serious illnesses • have you ever had a sexually transmitted infection • have you ever smoked • have you ever taken any other medication • have you had a colonoscopy • have you had an accident • have you had any past bladder problems • have you had a prostate exam • have you had back injury before • have you had back pain before • have you had blood glucose checked • have you had frequent urination before • have you had joint pain before • have you had physical therapy • have you had unprotected sex • have you had your blood glucose checked • have you noticed a discharge • have you noticed an itch • have you noticed an odor in your urine • have you noticed any blood in your urine • have you noticed any physical changes • have you noticed pain while urinating • have you seen a doctor • have you seen anyone else • have you tried anything else for the pain • have you tried anything for the frequent urination • have you tried any treatment • have you tried heat or ice • have you tried ice • having mood changes • hello • hopefully we can help you • how about your bladder • how about your hip • how are you • how are your children • how did the pain start • how did you get here today • how did

your grandparents die • how else is the frequent urination affecting you • how else is the pain affecting you • how far did you get in school • how frequent is the frequent urination • how frequent is the pain • how has the pain changed over time • how has this affected you • how has your day been going • how have you been handling this • how heavy was the couch • how intense is the pain • how is home • how is work • how is your blood pressure • how is your cholesterol • how is your diet • how is your family • how is your hip • how is your mood • how is your pain now • how is your social life • how long does the pain last • how long have you been a worker • how long have you been married • how long have you been taking the aspirin • how long have you been taking the ibuprofen • how long have you had the pain • how many sexual partners • how much do you drink • how much do you work • how much ibuprofen have you taken • how much sleep • how often do you take the ibuprofen • how often do you take the saw palmetto • how often do you urinate • how old are you • how old are your grandparents • how old are your parents • how painful is it • how were you lifting • i am doctor winslow • i am medical student • i am sorry • i have all the information i need • i have more questions • i'll go report to the doctor • is pain better when you lie down • is pain worse when you lie down • is the frequent urination constant • is the frequent urination improving • is the frequent urination new •

is the frequent urination worse in the morning or at night • is the pain constant • is the pain dull • is the pain improving • is the pain in your upper or lower back • is the pain new • is the pain on one side • is the pain sharp • is the pain worse in the morning or at night • is there anything else i can help you with • is this the first time you have had this pain • is urination always painful • is your job physically demanding • is your job pleasurable • is your job stressful • is your partner male • i understand completely • i will do my best • i would like to get to know more • let me wash my hands • make a follow up appointment • mr. • mr. wilkins • my name is bob • name and date of birth • negative symptoms • nice talking with you • nice to meet you • nice to see you • no answer • ok then • please repeat that • should I call you by a different name • social history • sounds like • tell me about your children • tell me about your grandparents • tell me about yourself • tell me about your work • tell me more about the saw palmetto • tell me more about your back pain • thanks • that must be awful

• that must be hard • that was nice • the doctor will be in • today we have 15 minutes • to summarize • want to ask about • was the onset of the frequent urination sudden • was the onset of the pain sudden • were you doing anything when the pain started • were you healthy • were you lifting anything • what are your allergy reactions • what bothers you the most • what brings you in today • what can't you do at work • what caused the past back pain • what concerns you about the pain • what do you do for a living • what do you do for fun • what do you eat each day • what do you think about the buckeyes • what do you think about this weather • what do you think is the problem • what else have you tried for the frequent urination • what else have you tried for the pain • what have you tried for the pain • what is your emotional state • what is your goal for this visit • what is your name • what is your past medical history • what is your religious preference • what kind of medicine is that • what makes the frequent urination better • what makes the frequent urination worse • what makes the pain better •

what makes the pain worse • what pills did you take • what should i call you • what should we do • what was the dose • what was the dose of aspirin • what were the pills called • what would you like me to do • when did the frequent urination start • when did the pain start • when did you take the ibuprofen • when do you have the frequent urination • when do you have the pain • when have you had back pain like this before • when is the pain most severe • when was the last time you had intercourse • when was the last time you saw a doctor • when was your last bowel movement • when was your last period • when was your last tetanus shot • when were you born • where are your children • where do you live • where do you work • where is the pain • who are your sexual partners • who do you live with • who prescribed the medicine • who supports you • why do you take the medication • why do you take the supplements • why supplements • would you like some pain medication • you are welcome • you seem uncomfortable.

Appendix C: Example Conversation

Selections from a single conversation with the patient are presented below. Some less interesting turns and reply contents are omitted for space. The chosen interpretation for each turn is **bolded**.

Table C1. Selected turns from a single conversation.

Query	CS	CNN	correct?	Reply
What brings you in today?	what brings you in today	what brings you in today	1	I was hoping you could help me with my back pain, it really hurts! [. . .]
I'm sorry to hear that. How long has this been going on?	how long have you had the pain	how long have you had the pain	1	Almost a week I think.
Have you experienced anything like this before?	have you had pain before	any other problems	1	I had some back pain about 4 or 5 years ago, but that was fine after one day and one tylenol.
Has anything made it better this time?	what makes the pain better	what makes the pain better	1	The only thing that seems to help some is ibuprofen, and that doesn't help all that much.
Have you noticed a time of day in which it is worse?	when did the pain start	is the pain worse in the morning or at night	0	The pain started about four or five days ago.
Could you describe the pain?	describe the pain	describe the pain	1	It's a dull ache right in the middle of my lower back.
Could you rate on a scale of 1-10?	no answer	can you rate the pain	1	Right now, I would call my pain a 7. When I take the ibuprofen, it goes down to about a 5 or 6.
Does anything make it worse?	how intense is the pain	what makes the pain worse	0	It is pretty bad. My back hurts all the time and I can't even work.
Does the spread at all?	no answer	does the pain radiate	1	Not that I have noticed. It pretty much just stays in my back.
Is there anything else you'd like to mention to the doctor?	any other problems	any other problems	1	The occasional need to urinate more frequently.
Is the urination painful? Is there blood?	negative symptoms	have you noticed pain while urinating	0	No.
I would like ask a few questions about background. First, have you had any major illnesses in the past?	have you ever had any chronic illnesses	i have more questions	1	I have no medical issues.
Have you had a colonoscopy?	have you had a colonoscopy	have you had back pain before	1	Oh, I think I had that a couple years ago. Everything came back fine and I don't have to go back for a few more years.
Have you had any surgeries in the past?	any prior surgeries	any prior surgeries	1	I have never had any surgeries. I have gotten stitches a few times [. . .]

Table C1. Continued.

Query	CS	CNN	correct?	Reply
Moving on to your family. Are you parents alive?	are your parents living	are your parents living	1	Both of my parents are alive and well.
How about your grandparents.	tell me about your grandparents	tell me about your grandparents	1	All of my grandparents are deceased. I didn't really know them, they died when I was really little or before I was born.
Any family history of back pain?	does anyone in your family have back pain	does anyone in your family have back pain	1	Not that I am aware of.
Lastly, some questions about your home life. Who do you live with?	who do you live with	who do you live with	1	I live alone.
How about work?	no answer	how is work	1	Work has been good. [. . .] I have been working on a vw beetle restoration [. . .]
What do you do for entertainment?	what would you like me to do	what do you do for fun	0	I'd really appreciate it if you could make my back feel better!
How is your activity level?	are you active	how is your hip	1	I keep active. My job keeps me moving most of the day.
Do you smoke?	do you smoke	do you smoke	1	No. I always hated the smell.
Are you sexually active?	are you sexually active	are you sexually active	1	I am not currently sexually active.
Do you feel safe at home?	do you feel safe at home	do you feel safe at home	1	I feel safe. My place is in an ok area, nothing that makes me worried.

Cite this article: Stiff A, White M, Fosler-Lussier E, Jin L, Jaffe E and Danforth D (2024). A randomized prospective study of a hybrid rule- and data-driven virtual patient. *Natural Language Engineering* 30, 31–72. <https://doi.org/10.1017/S1351324922000420>