Interpreting Questions with a Log-Linear Ranking Model in a Virtual Patient Dialogue System

Evan Jaffe and **Michael White** and **William Schuler**
The Ohio State University
Department of Linguistics
jaffe.59@buckeyemail.osu.edu
{mwhite,schuler}@ling.osu.edu

Eric Fosler-Lussier
The Ohio State University
Dept. of Computer Science and Engineering
fosler@cse.ohio-state.edu

Alex Rosenfeld
University of Texas at Austin
Department of Linguistics
alexbrosenfeld@gmail.com

Douglas Danforth
The Ohio State University
Department of Family Medicine
doug.danforth@osumc.edu

Abstract

We present a log-linear ranking model for interpreting questions in a virtual patient dialogue system and demonstrate that it substantially outperforms a more typical multiclass classifier model using the same information. The full model makes use of weighted and concept-based matching features that together yield a 15% error reduction over a strong lexical overlap baseline. The accuracy of the ranking model approaches that of an extensively handcrafted pattern matching system, promising to reduce the authoring burden and make it possible to use confidence estimation in choosing dialogue acts; at the same time, the effectiveness of the concept-based features indicates that manual development resources can be productively employed with the approach in developing concept hierarchies.

1 Introduction

In this paper, we present a log-linear ranking model for interpreting questions in a virtual patient dialogue system, along with initial experiments to determine effective sets of features with this model.

Learning to take a medical history is fundamental to becoming a successful physician. Most methods for assessing history taking skills involve interaction with Standardized Patients (SP) who are actors portraying real patients. SP interviews are effective, but they require significant faculty effort and institutional support. As an alternative, virtual standardized patients (VSPs) can be valuable tools that offer a practical and accessible means of simulating standardized patient encounters. VSP simulations have the potential to allow students to practice their communication and history taking skills before working with Standardized Patients. Students can rehearse interviewing skills in a risk-free environment, providing additional opportunities for practice prior to standardized or real-world patient encounters.

Our VSP system closely models the interaction between doctors and patients. Our virtual patients are avatars representing standardized patients that students can interview and communicate with using natural language. Students take a medical history and develop a differential diagnosis of the virtual standardized patient, much as they would a standardized or actual patient. As shown in Figure 1, the dialogue system is embedded in an immersive learning environment designed to provide student doctors with a sense of presence, allowing them to "suspend disbelief" and behave as if the virtual patient is a real patient. The virtual world platform can be run in a variety of environments; here we focus on text-based interaction for laptops and mobile devices.

The current task is a question matching paradigm where user input is mapped to a set of predefined questions, which have scripted answers created by content authors, as in much previous work on question answering systems (Leuski and Traum, 2011). This approach allows for easier authoring than, for example, systems that use deep natural language understanding (Dzikovska et al., 2012; Dzikovska et al., 2013) or semantic parsing (Artzi and Zettlemoyer, 2013; Berant and Liang, 2014), and yet still achieves the desired learning objectives of the virtual patient system.

To date, the VSP system has been based on the

Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications, 2015, pages 86–96,
Denver, Colorado, June 4, 2015.

ChatScript[1] pattern matching engine, which offers a low cost and straightforward approach for initial dialogue system development. In an evaluation where a group of third-year medical students were asked to complete a focused history of present illness of a patient with back pain and develop a differential diagnosis, the VSP system answered 83% of the questions correctly. This level of accuracy sufficed for all students to correctly identify the appropriate differential diagnosis, confirming that the virtual patient can effectively communicate and answer complaint-specific questions in a simulated encounter between a doctor and a patient (Danforth et al., 2009; Danforth et al., 2013).

A limitation of rule-based pattern matching approaches, however, is the need to create all patterns manually and extensively test and refine the system to allow it to answer questions correctly, with no ability to use confidence estimation in making dialogue act decisions. With our log-linear ranking model, we aim to substantially reduce the burden of designing new virtual patients, as well as to make it possible to use confidence estimation to decide when the system should ask the user to clarify or restate his or her question.

To create a corpus for developing our statistical interpretation model, the ChatScript patterns were refined to correct errors found during the evaluation and then run on a set of 32 representative dialogues, with the interpretation of all questions hand-verified for correctness.[2]

The rest of the paper is organized as follows. In Section 2, we discuss related work. In Section 3, we present the log-linear ranking model formally, comparing it to more typical multiclass classification models. In Section 4, we describe the features we investigate in detail, with experimental results and analysis appearing in Section 5. Finally, in Section 6 we conclude with a summary and discussion of avenues for future investigation.

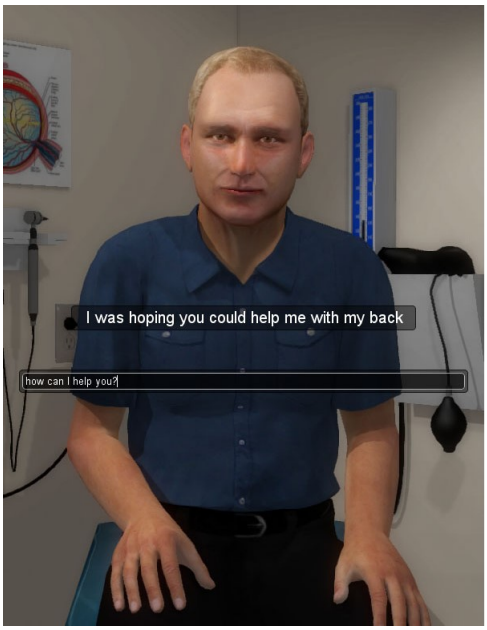

Figure 1: Example exam room and virtual patient avatar. The avatars are programmed to display emotions and movements that are appropriate for the nature of the question, interaction, or condition of the patient.

2 Background and Related Work

In a dialogue system where user utterances are expected to have one of a fixed set of expected interpretations, a straightforward way to implement the natural language understanding component is to map utterances to their interpretations using a multiclass classifier. DeVault et al. (2011) have pursued this approach with an interactive training system designed to enable users to practice multi-party negotiation skills by engaging with virtual humans. They employ a maximum entropy classification model with unigrams, bigrams, skip bigrams and length as features, reporting 87% accuracy in interpretation on transcribed user input (they then go on to show how acceptable accuracy can also be achieved incrementally with noisy ASR output). However, in our domain we find that a similar baseline model—using essentially the same information as the lexical

[1] http://chatscript.sourceforge.net/

[2] While the method by which we derived our corpus unfortunately precludes a direct comparison with the ChatScript patterns, since accuracy on the exact set of 32 dialogues in the corpus was not calculated before the patterns were corrected, we note that it is difficult in any case to fairly compare a pattern matching system with a statistical one, as the performance of the former is highly dependent on the time and effort spent refining the patterns. We consider the qualitative differences between the approaches to be of much greater importance, in particular that the machine-learned system can output a useful confidence measure and can be automatically improved with more training data, as discussed below and in Section 5. We are currently gathering a larger corpus of hand-corrected dialogues that will enable a direct comparison of accuracy in future work.

What kind of medicine is that

u: (what kind of medicine is that)	#must match exactly
u: ([kind type] * ~medicines)	#'kind' or 'type', then any word(s), then a ~medicines concept
u: (what * ~medicines * that)	#'what', then any word(s), then a ~medicines concept, then any word(s), then 'that'

Table 1: Example ChatScript patterns to match the canonical question, *What kind of medicine is that?* Brackets indicate disjunctions of terms, asterisks match zero or more words, and ~prefixes mark concepts, which are themselves disjunctions of terms or other concepts. *u* indicates that the pattern will match a question or statement. See Figure 4 for an example of a ChatScript concept.

overlap baseline discussed below—only achieves a mediocre 67% accuracy; presumably, this discrepancy results from many of the questions the virtual patient is expected to answer being more superficially similar to each other than is the case with DeVault et al.'s training system, thereby making the interpretation task more challenging.

Another way to approach the interpretation task is to view it as one of paraphrase identification, comparing user questions for the virtual patient to a set of expected questions. Since the introduction of the Microsoft Research Paraphrase Corpus (Dolan et al., 2004), or MSRP, there has grown a considerable body of research on paraphrase identification reporting results on this corpus. We draw on this research here, in particular for our baseline feature sets. In adapting these paraphrase identification methods to our setting, however, the question arises as to how to generalize beyond pairwise classification: with the MSRP corpus, the task is to take a pair of superficially similar sentences and classify it as a paraphrase or not a paraphrase, while here the goal is to identify which member of the set of expected questions provides the best match with the user's question. One way to find the best match would be to continue to make use of a binary classifier, selecting the best matching question as the one with the highest probability for the true paraphrase class. Alternatively, one can train a model to rank the competing alternatives, directly selecting the top-ranked option. In the context of question answering, Ravichandran et al. (2003) compared these two methods on the task of answer pinpointing and found that the ranking approach significantly improved upon the pairwise classification approach even using the same features, suggesting that with ranking models the alternatives compete more effectively in training than with binary classifiers, where the pairs are treated in piecemeal fashion. Subsequently, Denis & Baldridge (2007; 2008) also demonstrated a substantial performance improvement using a ranking model for coreference, in comparison to a pairwise classification model. Consequently, in this paper we have adopted the ranking approach.[3]

A perhaps surprising lesson from the paraphrase identification research based on the MSRP corpus is the strong performance of lexical overlap baselines. In particular, Das and Smith (2009) develop a lexical overlap baseline using 1- to 3-gram precision/recall/F-score features over words and stems, reporting 75.4% accuracy on the MSRP corpus. This lexical overlap baseline substantially exceeds many (and perhaps even most) published results on the task, as well as the performance of their own soft alignment model based on quasi-synchronous grammar; moreover, using this much fancier alignment model together with the lexical overlap baseline, they are only able to achieve a 0.7% improvement to 76.1%. Das & Smith's strong results with a lexical overlap baseline echo Wan et al.'s (2006) earlier results using features inspired by the BLEU MT evaluation metric (Papineni et al., 2002). More recently, Madnani et al. (2012) have shown that BLEU can be combined with a variety of newer MT evaluation metrics in classifier obtaining 77.4% accuracy, until recently the best result on the MSRP corpus. In particular, they showed

[3]Note that in general, ranking models allow for a variable number of alternatives, as may be familiar from log-linear parsing models; while allowing for a variable set of prediction options is not necessary in our setting, and thus our ranking model is technically also a multiclass classification model, its feature set is more like those found in typical ranking models than typical classification models, as explained further in Section 3.

that just using BLEU (and two other base metrics using only words, not stems, namely NIST and TER) together with Meteor (Denkowski and Lavie, 2011)—which goes beyond BLEU in employing stems, WordNet synonyms and a database of paraphrases acquired using the pivot method (Bannard and Callison-Burch, 2005)—yields 76.6% accuracy, already one of the best results on this corpus.

Given the strong performance of Das & Smith's lexical overlap baseline, we use these features as a starting point for our log-linear ranking model, and we also combine them with Meteor to yield two competitive baselines. On our corpus, the baselines deliver 75–76% accuracy, much higher than the 67% accuracy of the DeVault et al. multiclass classifier approach. We then add weighted variants of the Das & Smith baseline features, using information content estimated from the Gigaword corpus and a task-specific measure of inverse document frequency, yielding a nearly 3% absolute improvement.

The remaining features we investigate are inspired by our success to date in using handcrafted ChatScript patterns for interpreting user questions. Note that unlike with the MSRP corpus, where the task is to identify unrelated, open domain paraphrases, in our setting the task is to interpret related questions in a constrained domain. As such, it is not overly onerous to arrange relevant words and phrases into a domain-specific concept hierarchy to enhance ChatScript pattern matching. Using the concept hierarchy already developed for use with ChatScript, we are able to achieve a greater than 3% absolute improvement in accuracy over the lexical overlap baseline, indicating that developing such hierarchies may be the most productive way to employ manual development resources. ChatScript additionally makes use of a notion of topic to organize the dialogue, which we incorporate into our model using topic transition features. Finally, to fine tune patterns, ChatScript allows words that should not be matched to be easily specified; as such, we investigate a general method of discovering useful lexically specific features. Unfortunately, however, the topic and lexical features do not yield appreciable gains.

Other approaches to paraphrase identification with the MSRP corpus have been investigated. In particular, vector space models of word meaning have been employed to assess text similarity, representing a rather different angle on the problem in comparison to the methods investigated here, which we plan to explore in future work in combination with our current methods. For example, Rus et al. (2011) make use of Latent Semantic Analysis, a technique they have found effective in their work on interpreting user input in intelligent tutoring systems; however, their results on MSRP corpus lag several percentage points behind the Das & Smith lexical overlap baseline. Socher et al. (2011) present another vector space method making use of recursive autoencoders, enabling vectors for phrases in syntactic trees to be learned. Their method yielded the best published result at the time, though perhaps surprisingly their accuracy is nearly identical to using Meteor together with baseline MT metrics, trailing Madnani et al.'s (2012) best MT metrics combination by half a percentage point. More recently, Ji and Eisenstein (2013) have obtained the best published result on the MSRP corpus by refining earlier distributional methods using supervised information, in particular by discriminatively reweighting individual distributional features and learning the relative importance of the latent dimensions. Xu et al. (2014) have also shown that an approach based on latent alignments can improve upon Ji and Eisenstein's methon on a corpus of Twitter paraphrases.

Finally, Leuski and Traum (2011) present a method inspired by research on cross-language information retrieval that ranks the most appropriate system responses by measuring the similarity between the user's question and the system's potential answers. We have chosen to keep the formulation of the virtual patient's responses separate from question interpretation, though that remains a potential avenue for exploration in future research.

3 Log-Linear Ranking Model

In designing a virtual patient, the content author devises a set of expected questions that the virtual patient can answer. Each expected question has a canonical form, and may additionally have variant forms that have been collected during initial interactions with the virtual patient[4]. Thus, considering the

[4]Variants are identified automatically from training data any time two asked questions are annotated with the same canonical question.

canonical form of the question to be one of its variants, the task of the interpretation model is to predict the correct canonical question for an input question based on one or more known variants of each canonical question.

Formally, we define the likelihood of a canonical question c given an input question x using a log-linear model that marginalizes over the observed variants v of c:

$$P(c|x) = \frac{1}{Z(x)} \sum_{v \in c} \exp\Big(\sum_j w_j f_j(x, v) \Big) \quad (1)$$

Here, the features $f_j(x, v)$ are intended to indicate how well the input question x matches a variant v, and $Z(x)$ normalizes across the variants:

$$Z(x) = \sum_v \exp\Big(\sum_j w_j f_j(x, v) \Big) \quad (2)$$

In training, the objective is to choose weights that maximize the regularized log likelihood of the correct canonical questions c_i for each input x_i:

$$\sum_i \log P(c_i|x_i) - \lambda \sum_j w_j^2 \quad (3)$$

The model is implemented with MegaM,[5] using a default value of $\lambda = 1$ for the Gaussian prior regularization parameter.[6] We also experimented with a linear ranking SVM (Joachims, 2002; Joachims, 2006), but did not observe a performance improvement.

At test time, we approximate[7] the most likely canonical question c^* for input question x as the canonical question $c(v^*)$ for the best matching question variant v^*, i.e. the one with the highest score:

$$\begin{aligned} c^* &= c(v^*), \text{where} \\ v^* &= \text{argmax}_v \textstyle\sum_j w_j f_j(x, v) \end{aligned} \quad (4)$$

[5] `http://www.umiacs.umd.edu/~hal/megam/`

[6] We used MegaM's `-explicit` format option to implement the ranking model, where each question variant is considered a class, along with the `-multilabel` option to give a cost of zero to all variants of the correct canonical question and a cost of one to all other variants.

[7] A testing objective that more closely following the training objective was also attempted. This testing method summed over likelihoods of variants for a given canonical question, and then took the argmax over canonical questions. This method did not perform as well as the approximation.

In our ranking model, features can be defined that are shared across all question variants. For example, in the next section we make use of an unweighted unigram recall feature, whose value is the percentage of words in v that also appear in x:

$$f_1(x, v) = \text{unigram\_recall}(x, v)$$

In training, a single weight is learned for this feature (rather than one per class), indicating the relative contribution of unigram recall for predicting the correct interpretation. We expect that the trained weights for general features such as this one will carry over reasonably well to new virtual patients, aiding in the process of bootstrapping the collection of training data specific to the new virtual patient.

It is also possible to define lexical- and class-specific features. For example, the following feature indicates a recall miss for a specific word (*ever*) and canonical question (c_{27}):

$$f_2(x, v) = \begin{cases} 1, \text{ if } \textit{ever} \text{ in } v \text{ but not } x \text{ and} \\ \quad c(v) = c_{27} \\ 0, \text{ otherwise} \end{cases}$$

Sparse features such as this one are intended to fine-tune the predictions that can be made with the more general, dense features like the one above. Note, however, that class-specific features cannot generally be expected to carry over to predictions for new virtual patients (except where the patients are designed to answer some of the same questions).

While our ranking model allows us to make use of features that are defined in terms of the words in both the input question x and a variant question v, it is worth pointing out that most implementations of log-linear classification models require features to be defined only in terms of the input x, with the class implicitly conjoined, and thus with no features shared across classes. For example, Devault et al.'s (2011) maximum entropy classification model—as well as our multiclass baseline model below—makes use of class-specific features indicating n-grams found in the input, such as

$$f_3(x, c) = \begin{cases} 1, \text{ if } \textit{have you} \text{ in } x \text{ and} \\ \quad c = c_{27} \\ 0, \text{ otherwise} \end{cases}$$

Here, the weight learned in training is indicative of the relative importance of the bigram *have you* for

predicting a specific class, i.e. the one for canonical question c_{27}. As noted above, such class-specific features cannot generally be expected to carry over to predictions for new virtual patients, and thus a model consisting of only such features will be of little value for new virtual patients.

4 Features

The features described below are used to create feature subsets evaluated as models. Precision and recall features are defined as being relative to either the asked question or the compared question, respectively. Precision n-gram features, for example, are the ratio of matched n-grams to total n-grams in the asked question. Matching can happen at the exact, stem, concept, or Meteor alignment level.

AlignScore the overall Meteor alignment score

LexOverlap 1- to 3-gram exact/stem unweighted precision/recall/F-score features inspired by Das and Smith

Weighting 1- and 2-gram exact and stem lexical overlap features weighted by IDF and InfoContent

Meteor 1- and 2-gram IDF/InfoContent weighted precision/recall, matched on Meteor alignments

Concept paraphrase-type features based on stem n-gram overlap, but using the concept hierarchy to add further equivalences. Includes 1- and 2-gram precision/recall, weighted and unweighted.

Lex lexical exact match features, as well as precision/recall miss and canonical question-specific precision/recall misses

Topic topic start and transition features

Inverse document frequency weighting is implemented by taking the canonical question and its variants as a document. A gram is weighted based on its frequency in documents, where a gram that only occurs in one or a few documents is more informative than a word that occurs in many documents.

$$\mathrm{IDF}(w) = \log((N+1)/(\mathrm{count}(w)+1))$$

concept: ~medicines [~drugs_legal analgesia **antibiotics** antidote claritin drug drugs hormone hormonal loratidine medication medications medicine meds narcotic 'pain killer' 'pain killers' painkiller pill prescription 'prescription medication' 'prescription medications' remedy steroid tablet tums]

Figure 2: An example ChatScript concept. The *~medicines* concept is defined in the figure, where *antibiotics* is an instance of *medicines*, and *~drugs_legal* is a subconcept of *~medicines*. Each concept is defined as a disjunction of terms, and can include subconcepts.

Asked: what kind of medicine is that
Compared: what type of tablet would that be

Asked: what ~anon of ~medicines is that
Compared: what ~anon of ~medicines would that be

Figure 3: Example sentence pair and derived concept n-gram sequence. The words *kind* and *type* match in an anonymous concept (indicated here as *~anon*) derived from a ChatScript pattern, while the words *medicine* and *tablet* match under the *~medicines* concept.

N is the total number of documents and $\mathrm{count}(w)$ is the number of documents the gram w appears in.

InfoContent weighting uses negative log probabilities of the Gigaword corpus. For bigrams, weighting is calculated as the product of probabilities of a unigram with the conditional probability of the subsequent gram, using Katz backoff.

Concept features are lexical overlap features that use domain-specific knowledge to allow for matching on more words than the exact or stem level. Concept matches occur when a stem matches another stem in a ChatScript concept hierarchy, defined by content authors as labeled classes of equivalent words or phrases.

See Figure 4 for an example. Concepts are used in Chatscript to increase generalizability of the match patterns and reduce authoring burden. To calculate concept features, stems are replaced with the concept name if the stem in the question is listed under a concept in the hierarchy.

Figure 3 shows an example sentence pair and its resulting concept n-gram sequence, given concepts that include *kind* and *type* in an anonymous concept (i.e., an unlabeled disjunction) in one of the ChatScript patterns, along with the words *medicine*

```
Lex:::what
Lex:::of
Lex:::that
LexMissPrec:::kind
LexMissPrec:::medicine
LexMissRec:::type
LexMissRec:::tablet
LexMissRec:::would
LexMissRec:::be
LexMissRecClass:::what_kind_of_tablet_would_that_be:::tablet
LexMissRecClass:::what_kind_of_tablet_would_that_be:::would
LexMissRecClass:::what_kind_of_tablet_would_that_be:::be
```

Figure 4: Example lexical features. These binary features fire in the presence (or absence, in the case of a *Miss*) of a specific word. *Prec* and *Rec* miss features fire when a word appears in one question, but not the other, and is defined in both directions. Here, *LexMissPrec:::kind* fires because *kind* appears in the asked question, but not the compared question. *Class* miss features define lexical misses that are specific to a canonical question, and are similarly defined with *Prec* and *Rec* to refer to the asked and compared question, respectively.

and *tablet* being included under the *medicines* topic. Lexical overlap features are then computed on this concept-level n-gram sequence.

Lexical features are binary features that include an exact match or miss. A canonical question-specific miss feature is implemented for precision and recall. See Figure 4 for example lexical features and descriptions, using the running example sentence pair from the concept features.

Topic features keep track of the topic at each point in the dialogue. They include binary transition features that track the current and previous topic, or else the current start topic in the case of the first line of a dialogue. For example, Figure 5 shows the features generated from three example training data. The previous topics are taken from the gold annotation during training and testing. If automatically classified values were used instead of this oracle setting, performance would likely not suffer greatly, given that these features were not found to be very informative and low weights were learned during training.

5 Experiments

The corpus consists of 32 dialogues, which include 918 user turns, with a mean dialogue length of 29 turns. For each turn, the asked question, canonical question, current topic and a question response are

Asked	Correct	Topic
1. hello, mr. covington	hello	~openingdialogue
2. What brings you in today	what brings you in today	~chiefcomplaint
3. Any health problems	what brings you in today	~chiefcomplaint

```
1. TOPICSTART:::~openingdialogue
2. TOPICTRANS:::~openingdialogue:::~chiefcomplaint
3. SAMETOPIC
```

Figure 5: Example topic features

annotated. 193 total canonical questions were created by content authors as the fixed set of classes. Correct canonical questions were obtained by running ChatScript, then hand-correcting the output. Any asked questions annotated with the same canonical question are considered variants of that canonical question. There are 787 variants, with a mean of 4.1 variants (standard deviation 4.7) per canonical question. The median number of variants is 2.0, and the maximum number is 34.0.

System accuracy is measured by outputting the correct canonical question, given an input question. Cross-fold validation is run on a per-dialogue basis.Total system accuracy is measured as the mean over all individual cross-fold accuracies.

Results of system accuracy by model are shown in Table 2. The weighted, concept-based, topic-based, and lexical features model (Full-no-meteor) shows a significant improvement over the LexOverlap baseline model, using a McNemar paired chi-square test (chi-square=16.5, p=4.86e-05). At an overall accuracy of 78.6%, this represents an error reduction of 15% over the baseline and approaches the performance of the handcrafted patterns. Of interest, the LexOverlap+concept shows a significant improvement over LexOverlap alone (chi-square=18.3, p=1.95e-05). Meteor features do not show a significant difference when comparing the Full vs. Full-no-meteor model (chi-square=3.2, p=.073), indicating that the concept-based features largely suffice to supply the information provided by WordNet synsets and pivot-method paraphrases in Meteor.

Training with variants as acceptable matches is a useful strategy for this domain, reducing error by 47%, as compared to training without variants. This allows for comparison at test time to not only the

Model Name	Features Included	% Accuracy
Align	Meteor AlignScore feature alone	75.3
LexOverlap	Das and Smith-style lexical overlap baseline	74.9
LexOverlap+lex	adds lexical features	74.1
LexOverlap+topic	adds topic features	75.1
LexOverlap+align	adds Meteor AlignScore	75.8
LexOverlap+weighting	adds weighting features	77.8
LexOverlap+concept	adds concept features	78.1
LexOverlap+concept+weighting	adds weighting and concept features	78.5
Full	all features	77.0
Full-no-meteor	full minus AlignScore and Meteor features	78.6

Table 2: Model results, with a description of their included features

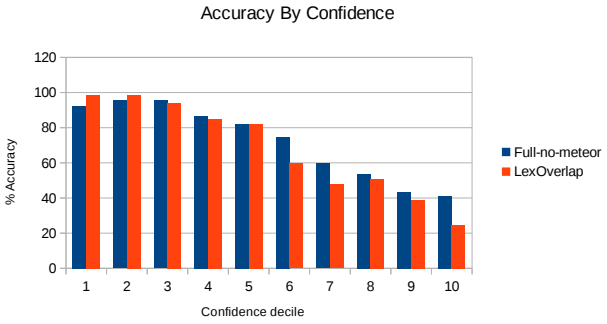

Figure 6: Percent accuracy shown by deciles of decreasing confidence. The most confident deciles have the highest accuracy.

canonical version of a question, but also each correct variant of the canonical version. Matching the correct canonical question or any of its variants results in a correct system response.

In addition, accuracy is higher in cases where the model is most confident, suggesting that confidence can be successfully employed to trigger useful clarification requests, and that training with question variants acquired in previous dialogues yields a large reduction in error. Lastly, an error analysis reveals that many question interpretation errors yield matches that are close enough for the purposes of the dialogue, though some errors remain that reflect misleading lexical overlap, lack of world knowledge or the lack of a dedicated anaphora resolution component.

A measure of system confidence can be obtained from test items' probabilities, and can be compared to accuracy to show that higher confidence system responses are more accurate. Confidence is defined as follows:

$$P(v|x) = \frac{\exp \sum_j w_j f_j(x,v)}{\sum_v \exp \sum_j w_j f_j(x,v)} \quad (5)$$

In Figure 6, test items' answer probability is binned by decile. Mean response accuracy is then calculated for each bin of test items. Future work will use confidence to make discourse management decisions, such as when to answer a question, ask for clarification between close candidates, or give a generic response. Additionally, higher system accuracy is possible if the system is limited to answering higher confidence quantiles.

As an alternative to the log-linear ranking model employed here, a baseline multiclass classifier[8] trained on 1- to 3-gram word and stem indicator features obtains an accuracy of 67%. The ranking system performs better when trained on essentially the same information (LexOverlap), with 75% accuracy.

A ranking model using SVMRank (Joachims, 2002; Joachims, 2006) was also tried, but performance (not shown) was similar to the log-linear model. Future work might explore other machine learning models such as neural networks.

System errors largely fall into a few categories. First, some responses are actually acceptable, but reported as incorrect due to a topic mismatch. For example, the same question *have you ever had this type of pain before* could be labeled as *have you ever had this pain before* or *have you ever had back pain before*, depending on the topic. If the topic was *currentbackpain* or *currentpain*, the gold label could differ. Topics, therefore, exist at varying levels of

[8] http://scikit-learn.org/

specificity. Including nearly identical questions in multiple topics promotes question reuse across virtual patients but can be a source of error if the topic is not tracked well.

A second class of errors comes from superficially similar questions, where the most meaningful word or words in the question are not matched. For example *does the pain ever go away* vs. *does rest make the pain go away* would have high lexical overlap, but this does not reflect the fact that the most informative words do not match. Interestingly, we expect that questions that match primarily on common n-grams and not on rarer n-grams have relatively low confidence scores, since the common n-grams would match multiple other questions. Using confidence scoring could help mitigate this error class.

For the previous example, the correct question is actually, *is the pain constant*, which highlights a third kind of error, where some inference or world-knowledge is necessary. Understanding that things that *go away* are not *constant* is an entailment involving negation and is more complicated to capture than using a paraphrase resource.

While room exists for absolute improvement in accuracy, the results are encouraging, given the relatively small dataset and fact that the full model approaches ChatScript pattern-matching system performance (83%). Larger datasets will likely improve accuracy, but given the expense and limited availability of large corpora, we focus on exploring features that maximize limited training data. Annotation is in progress for a larger corpus of 100 dialogues with approximately 5500 user turns.

Qualitatively, the ranking system is less labor-intensive than ChatScript and can use confidence values to drive dialogue act decisions, such as asking the user to rephrase, or to choose between multiple candidate question interpretations. Additionally, the ranking system could potentially be combined with ChatScript to provide ranking when multiple ChatScript patterns match, or to provide a question when no existing ChatScript pattern matches the input.

Better anaphor resolution could help address errors from uninformative pronouns that might not match the canonical question form. Zero-anaphors are missed by the current features and could occur in a dialogue setting such as: *What medications are you taking*, followed by *ok, how often*.

6 Conclusion

In this paper, we have presented a log-linear ranking model for interpreting questions in a virtual patient dialogue system that substantially outperforms a vanilla multiclass classifier model using the same information. In the full model, the most effective features turned out to be the concept-based matching features, which make use of an existing concept hierarchy developed for an extensively handcrafted pattern matching system, and play a similar (but less error-prone) role as WordNet synsets and pivot-based paraphrases in tools such as Meteor. Together with weighted matching features, these features led to a 15% error reduction over a strong lexical overlap baseline, approaching the accuracy of the handcrafted pattern matching system, while promising to reduce the authoring burden and make it possible to use confidence estimation in choosing dialogue acts. At the same time, the effectiveness of the concept-based features indicates that manual development resources can be productively employed in the ranking model by developing domain-specific concept hierarchies.

The student-VSP interaction creates a comprehensive record of questions and the order in which they are asked, which allows for student assessment as well as the opportunity for focused practice and improvement. Indeed, the primary goal of our current research is to leverage the advantages of the VSP system to provide for deliberate practice with immediate feedback.

To better support student practice and assessment, we plan to investigate in future work the impact of more advanced methods for anaphora resolution, as our error analysis suggests that questions containing anaphors are a frequent source of errors. In a dialogue system that uses speech input, we expect automatic speech recognition errors to hurt performance. The exact impact is left as an empirical question for future work. Finally, we also plan to investigate incorporating syntactically-informed vector space models of word meaning into our system, which may help to boost accuracy, especially when acquiring patient-specific training data during the early phase of developing a new virtual patient.

Acknowledgments

We would like to acknowledge Kellen Maicher who created the virtual environment and Bruce Wilcox who authored ChatScript and customized the software for this project. We also acknowledge the expert technical assistance of Laura Zimmerman who managed the laboratory and organized student involvement in this project.

This project was supported by funding from the Department of Health and Human Services Health Resources and Services Administration (HRSA D56HP020687) and the National Board of Medical Examiners Edward J. Stemmler Education Research Fund (NBME 1112-064).

References

Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1(1):49–62.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of ACL*, ACL '05, pages 597–604.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Baltimore, Maryland, June. Association for Computational Linguistics.

D. R. Danforth, M. Procter, R. Heller, R. Chen, and M. Johnson. 2009. Development of virtual patient simulations for medical education. *Journal of Virtual Worlds Research*, 2(2):4–11.

D. R. Danforth, A. Price, K. Maicher, D. Post, B. Liston, D. Clinchot, C. Ledford, D. Way, and H. Cronau. 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.

Dipanjan Das and Noah A. Smith. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 468–476, Suntec, Singapore, August. Association for Computational Linguistics.

Pascal Denis and Jason Baldridge. 2007. A ranking approach to pronoun resolution. In *In Proceedings of IJCAI-2007*, Hyderabad, India.

Pascal Denis and Jason Baldridge. 2008. Specialized models and ranking for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 660–669, Honolulu, Hawaii, October. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*.

David DeVault, Kenji Sagae, and David Traum. 2011. Incremental interpretation and prediction of utterance meaning for interactive dialogue. *Dialogue and Discourse*, 2(1):143–170.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of Coling 2004*, pages 350–356, Geneva, Switzerland, Aug 23–Aug 27. COLING.

Myroslava O. Dzikovska, Peter Bell, Amy Isard, and Johanna D. Moore. 2012. Evaluating language understanding accuracy with respect to objective outcomes in a dialogue system. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 471–481, Avignon, France, April. Association for Computational Linguistics.

Myroslava Dzikovska, Elaine Farrow, and Johanna Moore. 2013. Improving interpretation robustness in a tutorial dialogue system. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 293–299, Atlanta, Georgia, June. Association for Computational Linguistics.

Yangfeng Ji and Jacob Eisenstein. 2013. Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 891–896, Seattle, Washington, USA, October. Association for Computational Linguistics.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proc. KDD*.

Thorsten Joachims. 2006. Training linear svms in linear time. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*.

Anton Leuski and David Traum. 2011. Npceditor: Creating virtual human dialogue using information retrieval techniques. *AI Magazine*, 32(2):42–56.

Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the*

Association for Computational Linguistics: Human Language Technologies, pages 182–190, Montréal, Canada, June. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA.

Deepak Ravichandran, Eduard Hovy, and Franz Josef Och. 2003. Statistical QA — classifier vs. re-ranker: What's the difference? In *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering*, pages 69–75, Sapporo, Japan, July. Association for Computational Linguistics.

Richard Socher and Eric H. Huang and Jeffrey Pennington and Andrew Y. Ng and Christopher D. Manning. 2011. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In *Advances in Neural Information Processing Systems 24*.

Vasile Rus, Mihai Lintean, Arthur C. Graesser, and Danielle S. McNamara. 2011. Text-to-text similarity of sentences. In Phillip McCarthy and Chutima Boonthum-Denecke, editors, *Applied Natural Language Processing*. IGI Global.

Stephen Wan, Mark Dras, Robert Dale, and Cecile Paris. 2006. Using dependency-based features to take the 'para-farce' out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop 2006*, pages 131–138, Sydney, Australia, November.

Wei Xu, Alan Ritter, Chris Callison-Burch, William B. Dolan, and Yangfeng Ji. 2014. Extracting lexically divergent paraphrases from Twitter. *Transactions of the Association for Computational Linguistics (TACL)*, 2(1).