Practical Language Processing for Virtual Humans

Anton Leuski and **David Traum**
Institute for Creative Technologies
13274 Fiji Way
Marina del Rey, CA 90292

Abstract

NPCEditor is a system for building a natural language processing component for virtual humans capable of engaging a user in spoken dialog on a limited domain. It uses a statistical language classification technology for mapping from user's text input to system responses. NPCEditor provides a user-friendly editor for creating effective virtual humans quickly. It has been deployed as a part of various virtual human systems in several applications.

1 Virtual Humans

Imagine talking to a computer system that looks and acts almost human—it converses, it understands, it can reason, and exhibit emotion. As an example, recall some of such computer characters created by Hollywood moviemakers: the librarian in *Time Machine*, the holographic professor in *I Robot*, and of course, the holodeck characters in numerous *Star Trek* episodes. Once the realm of science fiction, limited, domain-specific versions of these kinds of characters are now achievable, using AI and computer graphics technology. Such simulations, called *virtual humans* (VH), open up whole new horizons for entertainment, teaching, and learning. Virtual humans can serve as colleagues or adversaries in training simulations helping a student to study language and culture (Johnson, Vilhjalmsson, and Marsella 2005) or hone her negotiation skills (Traum et al. 2005). They can assist physicians in treating psychological disorders (Rizzo et al. 2006) or to train the physicians themselves (Kenny, Parsons, and Rizzo 2009). They work as virtual guides (Jan et al. 2009), museum docents (Kopp et al. 2005), or engage the user in a gunfight (Hartholt et al. 2009).

A typical virtual human system is rather complex and may consist of several components including with automatic speech recognition (ASR), gesture recognition, language understanding, dialogue management, emotion reasoning, planning, inference, verbal and non-verbal output, body simulation, realistic graphics, and mixed reality displays (Gratch et al. 2002). For virtual human systems to become mainstream there are two main requirements. First, the advances in technology must reach the level of reliability and efficiency that make the interactions with virtual humans seamless and realistic. Second, this technology has to be implemented and packaged in software that is accessible to the training or entertainment system designers who are not technical experts in the underlying AI technology. In this paper, we concentrate on the former, as well as examples of use that serve as evidence that we have been successful in the latter. A companion paper includes mode details on the user interface for the technology and how it is used to build new characters (Leuski and Traum 2010).

As with most complex software, there are different types of users, using it in different ways. Here we distinguish among four categories of users of virtual human technology:

Designers *author* the VH system using available tools. They create scenarios, develop the look and behavior of the agents, including the interactive dialogue behavior.

Administrators *deploy* the system, maintain it in the working order so that others can interact and view the VHs.

Interactors *talk to* the VH. There are really two types of interactors, *demoers* who work with the Administrators and are familiar with the system, and *players* who are not. Players are the primary target of the interaction. In the case of Demoers, it is the audience that is the primary target of interaction and demoers are presenting the system to the audience.

Audience members *observe* others interact with the virtual human. As said above, when interactors are demoers then the audience is the primary target of the interaction, however there may be an audience member acting as secondary target even when the interactors are players.

In this paper we describe NPCEditor[1]—a system used to support all of the above user classes with the natural language processing (NLP) parts of a virtual human system, including natural language understanding and generation and dialogue management. At the core of the system is a statistical text classification algorithm developed specifically for the task of understanding the interactor's language. NPCEditor packages the text classifier in a GUI-based application that allows creation of useful VH systems with minimal training. NPCEditor has been used extensively in a number of projects both internally in our group and externally by

[1] *NPC* stands for Non-Player Character

other teams at the institute and outside organizations. VH systems that use NPCEditor have been deployed at training locations, shows, virtual worlds, and museums.

In the rest of the paper we outline the NPCEditor design, describe its role in constructing a VH system, describe the statistical classification technology at the core of the system, and summarize some experimental results that show the effectiveness of the classification approach. We also describe several current and past projects that used NPCEditor.

2 Scientific Contribution: Cross-Language Retrieval for Dialogue Response Selection

There are many NLP technologies that might be applied to virtual human language interaction. The choice depends in large part on the required capabilities: Does the VH have a firm agenda or is it more flexible? Does it lead the interaction or react? Does it need to perform deep inference on the meaning of what is said, or can it stay close to the surface? Will the responses need to be computed on the fly based on current context, or can they be pre-computed or authored?

NPCEditor has been used primarily to construct *question-answering characters*. These characters play the role of interviewees and respond to questions in character. There are many kinds of interviews (doctor-patient, police-suspect, reporter-witness, information seeker-expert, and so forth) and thus question-answering characters have broad applicability. For example, imagine that you are playing the role of a detective in the game of "Clue."[2] An owner of a large house has been murdered and you interrogate the guests of the house. The house guests and witnesses are played by virtual humans. Each character should be capable of answering a number of questions on a limited set of topics that are potentially relevant to the event and it should be able to deflect all other questions.

A question answering virtual human is characterized by a collection of responses relevant to a particular topic. This approach gives complete control over the virtual persona's knowledge and expressions to the scriptwriter who creates the responses. It allows the writer to specify the character of the virtual persona, what information it can deliver and the form of that delivery. When an interactor comes up to the virtual character and asks it a question, the system driving the character analyzes the interactor's question and selects the appropriate response from the collection.

This approach also simplifies the overall VH system: a bare-bones system would consist of an ASR module for speech processing, the NPCEditor system to process the interactor's utterances and select the character response, and a rendering engine, capable of presenting the animated character on the screen and playing back prerecorded responses.

Automatic question answering has been studied extensively in recent years. It is generally defined as an information retrieval (IR) problem where a user places her request in a form of a question and expects a relevant and succinct response, e.g. "How tall is mount Everest?"—"Mount Everest is 29029 feet tall." One example of such a system is START from MIT (Katz 1988). It uses well-defined informational databases and carefully crafted question parsing rules to find the required answer. Web-based question answering systems and systems studied in the context of the question-answering track at the Text REtrieval Conference (TREC) attempt to answer user's questions by finding and parsing relevant paragraphs in large text collections (Voorhees 2003).

In contrast to the fact-based question answering scenario where the goal is to provide the most *relevant* answer, we focus on the answer's *appropriateness*. In our example about an investigation, an evasive, misleading, or an "honestly" wrong answer from a witness character would be appropriate but might not be relevant. Alternatively, different characters may have different knowledge about the event and respond differently to the same question. We try to highlight that distinction by talking about question-answering *characters* as opposed to question-answering systems or agents. Another difference is that question-answering systems rely on the question text to be lexically and grammatically correct and well-formed. Our system operates with transcriptions of spoken language, which is often different from the written language forms available in collections of written English. A third difference is that the input for our system comes from an automatic speech recognition module that sometimes introduces errors into the transcription. These errors can affect the interpretation performance significantly. A virtual human should be robust to both disfluencies in conversational English and to the errors introduced by the ASR.

Similar requirements exist for automatic phone reservation and call routing systems (Gorin, Riccardi, and Wright 1997). For example, Chu-Carroll and Carpenter describe a system that picks up a phone, asks a caller some questions, and routes the call to the appropriate destination (Chu-Carroll and Carpenter 1999). The system uses vector-based text classification approach to analyze the caller's responses and map them to the destinations in the organization. Our NPCEditor system maps text of the question directly to texts of the answers and uses a novel text classification approach based on statistical language modeling that significantly outperforms vector-based approaches (Leuski et al. 2006).

Text classification has been studied for several decades, and numerous approaches exist (Lewis et al. 1996). It is the task of assigning pieces of text to one or more classes based on the training data. The traditional text classification approach for our task is to define each answer as a class and define the corresponding questions as the training text pieces. When a new question arrives, it is compared to the known questions and the answer corresponding to the best matching group of questions is returned. The disadvantage of this approach is that it completely ignores the content of an answer. Two answers might be very similar and convey exactly the same information, but unless the same training questions are linked to both answers, only one of them will be returned by the classifier.

The main difference between our text classification and a traditional text classification approach is that we do not compare a user's question to known questions—we compare it to known answers. This approach relies heavily on results from cross-language information retrieval.

[2] "Clue" is official trademark of Hasbro Inc.

The key achievement of IR is an ability to match two strings of text based on content similarity. That is how a search system works—a text representation is computed for documents and a query, a matching algorithm is applied, and the best match is returned to the person who entered the query. One technique for text content representation that has recently gained wide usage in IR (Ponte and Croft 1997) uses a statistical language model: a probability distribution $P(W)$ over all possible word strings $W = w_1, ..., w_n$. A topic can be described by a set of text sentences. The probability of observing a particular sentence describing the topic will vary from topic to topic. Thus, a language model can be used as a technique to represent the topic content.

Before we describe the details of the method, we have to make an observation: We cannot compare question and answer language models directly because the former is the probability over questions and the latter is the probability over answers. These probabilities are not the same, as some words (e.g. "wh" words) are much more likely to appear in questions, while others are more likely to appear in answers. Moreover questions and answers are "generated" by different entities—the interactor and the character (or, the scriptwriter). In this paper we speak about questions and answers as samples from two different languages.

We can, however, compare a conditional probability of an answer given an observed question $P(A|Q)$ with the language models of known answers. One can interpret this value as a "translation" of the question Q into the language of answers. Here we use the character database as the "parallel corpora" that maps questions to the corresponding strings in the answer language. The translation rules are implicitly derived from that mapping. This problem is similar to the cross-language information retrieval task, e.g., where a search system has to find Chinese documents in response to an English query (Grefenstette 1998).

There are different ways to compare two probability distributions. NPCEditor uses the Kullback-Leibler (KL) divergence $D(P(F|W)||P(F))$ defined as

$$D(P(A|Q)||P(A)) = \int_A P(A|Q) \log \frac{P(A|Q)}{P(A)} \quad (1)$$

which can be interpreted as the relative entropy between two distributions.

Normally a topic is represented by a single text string. It is impossible to determine the language model from such a sample explicitly. The goal is to estimate the $P(Q)$ as accurately as possible. The problem of estimating the joint probability $P(q_1, ..., q_n)$ of several words occurring together to form a string of text Q has received a lot of attention in recent years among researchers in the IR community. The main challenge is to take into account of the interdependencies that exist among the individual words while still making the computation feasible. Several different methods were suggested starting from the most trivial technique where all words are assumed to be distributed identically and independently from each other—the unigram model in Equation (2).

$$P(q_1, ..., q_n) = \prod_{i=1}^{n} P(q_i) \quad (2)$$

Other approaches include Probabilistic Latent Semantic Indexing (PLSI) (Hofmann 1999) and Latent Dirichlet Allocation (LDA) (Blei et al. 2003), where the authors model text collections by a finite set of k topics and the overall text probability is viewed as a mixture of the individual topic language models.

Lavrenko (Lavrenko 2004) suggests a more general approach where the word interdependencies are defined by an unknown parameter vector θ and the words are taken as conditionally independent. This allows relaxing the independence assumption of the unigram model so that the probability distribution depends on the co-occurrence of words.

An approximation for the joint distribution is shown in Equation (3), where S is all questions from the character database, $|S|$ is the size of the training set and $p_s(q_i)$ is the probability distribution of words in string s.

$$P(q_1, ..., q_n) = \frac{1}{|S|} \sum_{s \in S} \prod_{i=1}^{m} p_s(q_i) \quad (3)$$

There are several ways of estimating the latter value. We use Maximum Likelihood Estimation (MLE) with Jelinek-Mercer smoothing (Bahl, Jelinek, and Mercer 1990):

$$p_s(q) \cong \pi_s(q) = \lambda_\pi \frac{\#_s(q)}{|s|} + (1 - \lambda_\pi) \frac{\sum_s \#_s(q)}{\sum_s |s|} \quad (4)$$

$\#_s(q)$ is the number of times word q appears in string s, $|s|$ is the length of string s, and λ_π is a tunable parameter that can be determined from the training data.

Equation (3) assumes that all words q_i come from the same vocabulary. We can show that in the case of two different vocabularies, the conditional probability of observing a word in answer language a given an interactor's utterance Q $P(a|Q)$ can be estimated as:

$$\begin{aligned} P(a|Q) &= \frac{P(a, q_1, ..., q_n)}{P(q_1, ..., q_n)} \\ &= \frac{\sum_s \pi_{A_s}(a) \prod_{i=1}^{m} \pi_{Q_s}(q_i)}{\sum_s \prod_{i=1}^{m} \pi_{Q_s}(q_i)} \end{aligned} \quad (5)$$

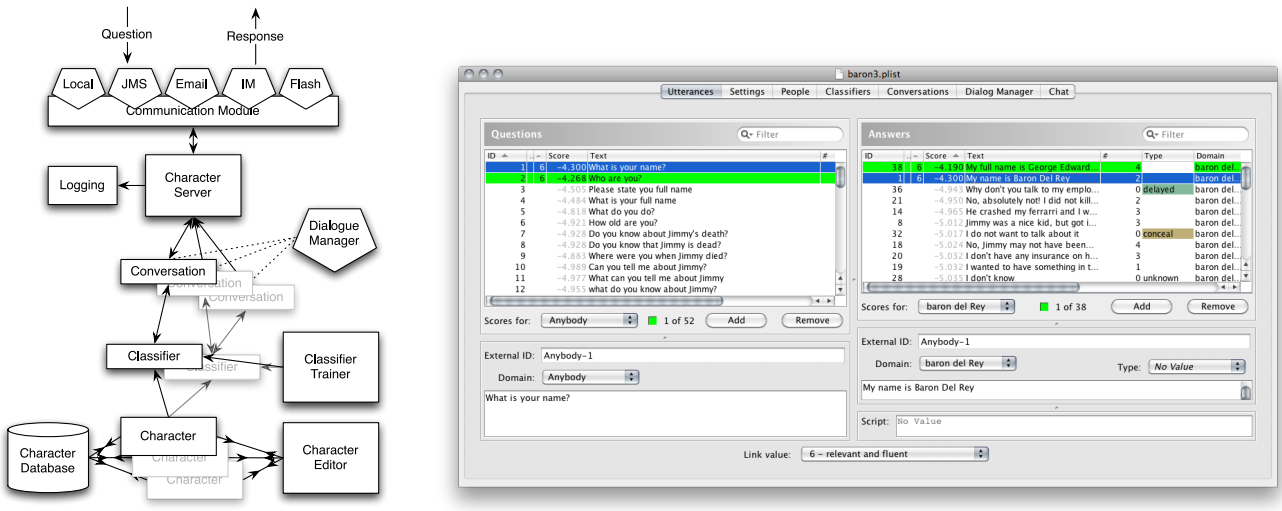
The matching criteria in Equation 1 can be written as

$$D(P(Q)||P(A)) = \sum_a P(a|Q) \log \frac{P(a|Q)}{\pi_A(a)} \quad (6)$$

In summary, given a character database $\{A_s, Q_s\}$ and a question Q, we use Equations 3, 4, and 5 to compute Equation 6 for each answer A in the database and return the answer with the highest value $-D(P(Q)||P(A))$. See (Leuski et al. 2006; Leuski and Traum 2008) for more details.

Non-lexical Features

So far we have described how a textual answer is selected in response to a textual question. There are several other cases in which the NPCEditor uses the same classification algorithm to go beyond this scenario. First, in some applications we may use the cross-language information retrieval approach to convert between text and a semantic language.



(a) (b)

Figure 1: a) NPCEditor system design. b) Character editor screen.

In some systems, we use the NPCEditor to recognize features such as speech acts or impact on interpersonal variables (Roque and Traum 2007), while in other systems, the NPCEditor can be used interpret the meaning of an utterance in a semantic representation, rather than selecting the answer to respond (Gandhe et al. 2008). Likewise, the NPCEditor can be used to translate a semantic representation of a response into text (Leuski and Traum 2008).

In some applications additional context information might be available as well as text. For example, in the Gunslinger system (Hartholt et al. 2009) the interactor meets with three different virtual humans. The system uses NPCEditor, an ASR module, and a vision component, which (among other things) detects where the interactor is looking. NPCEditor annotates the ASR output with a token corresponding to the interactor's gaze target. The classifier treats such annotations as words in a piece of text associated with the question but separate from the actual question text. This way a question becomes a multi-field data structure. One of these fields contains the original text, the other fields contain label tokens. These label tokens have special vocabulary different from the question text vocabulary, so a separate language model is estimated for each field. The question-answer similarity score becomes a weighted sum of similarities between the answer language model and the language models for each field in the question data structure:

$$D(Q||A) = \sum_i \alpha_i \sum_{w \in V_i} P_i(w|Q) \log \frac{P_i(w|Q)}{P(w|A)} \quad (7)$$

here the outer summation goes over every field of interest, while the inner summation iterates over vocabulary for the ith field. The parameters α_i allow us to vary the importance of different fields and can be determined from the training data. Thus NPCEditor can be trained to respond differently to the same question,—e.g., "What is your name?",—depending on who is the interactor is looking at. NPCEditor's user interface allows the designer to define arbitrary annotation classes or categories and specify some of these categories as annotations to be used in classification.

3 Practical Contribution: System for Character Development and Deployment

While the cross-language information retrieval models described in the previous section have been shown to be effective (see Section 4), it can still be daunting for a system developer to master the equations, create the requisite training data, and train a classifier. It may also be a challenge for a system administrator to connect this module to other parts of a VH system, so that interactors can successfully communicate with the virtual human. To this end, NPCEditor creates a unified development and run-time interface, that allows easy character authoring and run-time usage. NPCEditor is written in Java and should run on any platform that supports Java 6. It has been extensively tested on Microsoft Windows and Mac OS X. The initial version was developed part-time in 2004-2005 and subsequently maintained and extended by the first author. There are two main parts in NPCEditor: the *character database* that stores the virtual human information and language data and a *character server* that monitors the network, accepts incoming messages, processes the requests, and sends out character responses.

Figure 1a shows the block diagram of the NPCEditor system. The character database stores information about the virtual characters. A character designer can store multiple characters in the same database so the interactor(s) may have a conversation with several virtual humans at the same time. Each virtual human character is associated with a set of responses it can produce. The designer enters sample

questions and links them to the responses. The *classifier trainer* component generates *classifiers* using the methods described in Section 2 that map from the interactor's questions to the character's responses. The designer also selects one of the provided *dialogue manager* components. A dialogue manager is a rule-based subsystem that uses the classification results and the dialogue history to select the actual response. Finally, the character designer sets up the character server by registering network identities for each character with the *communication module* and enabling the conversation logging. Multiple people can interact with the virtual human at the same time. For this purpose the server maintains a list of *conversations*.

NPCEditor provides monitoring and control functionality over the individual system components using a GUI *character editor*. An NPCEditor window consists of several tabbed panels each corresponding to a particular function. These panels are listed below, along with the how they are used by the different user classes described in Section 1.

1. Utterances: a character designer specifies answers and sample questions and link them to each other.
2. Settings: a designer creates and modifies annotation categories and labels, assigns colors to labels, and specifies whether a category should be used as a non-lexical feature for classification.
3. People: an administrator uses to select available characters, edit the character properties and specify their network identities or accounts.
4. Classifier: a designer can select text classifiers for every unique questioner-responder character pair and train the classifier parameters.
5. Conversations: administrator can monitor conversations.
6. Chat: an interactor can pose arbitrary questions to the characters in the database and observe how the classifier ranks the available responses. Also, the audience can see details of the system performance. An administrator and designer can debug the characters.

Figure 1b shows an NPCEditor window with the utterance editor panel selected. There are two main areas here: the question editor is on left and the answer editor is on the right. Both the question and the answer editors follow the master-detail interface pattern: each lists all the utterances in a table and provides controls for editing the selected utterance. Specifically, a developer can define the utterance text, speaker, assign a text-based identifier and annotation labels. To link a question to an answer, the developer selects the question and the answer in the corresponding lists and assigns the link value using the popup menu at the bottom of the window. More details about the interface and uses may be found in (Leuski and Traum 2010).

4 Evaluation

We have evaluated NPCEditor in a number of off-line and on-line experiments. We have tested the classification accuracy, robustness to the errors in the classifier input, and user engagement in interactions with a virtual human. In this section we summarize some of these experiments. More details about the experimental setup and the results can be found elsewhere (Leuski et al. 2006; Artstein et al. 2009; Leuski and Traum 2008; Kenny, Parsons, and Rizzo 2009). Evaluations of the performance of characters built using the NPCEditor are briefly described in Section 5.

Classification Accuracy

In the first set of experiments we have evaluated the classification accuracy or how often the first answer returned by the system was appropriate. As the baseline we used a text classification approach based on Support Vector Machines (SVM). We represented questions as vectors of term features and the linked answers defined the question classes. We tokenized the questions and stemmed the tokens using the KStem algorithm (Krovetz 1993) in exactly the same way as we tokenize the text to compute language models. We used a $tf \times idf$ weighting scheme to assign values to the individual term features (Allan et al. 1998). Finally, we trained a multi-class SVM (SVM^{struct}) classifier with an exponential kernel (Tsochantaridis et al. 2004). We also experimented with a linear kernel function, various parameter values for the exponential kernel, and different term weighting schemes. The reported combination of the kernel and weighting scheme showed the best classification performance. Such an approach is well-known in the community and has been shown to work well in numerous applications (Joachims 1998). We believe it provides us with a strong baseline.

As the second baseline we used the language model approach described in Section 2, but we compared questions to questions instead of comparing them to answers. This is equivalent to making a single-language retrieval without the translation effect of the question-answer mapping. Specifically, in Equation 5 we use the likelihood over question terms and sum over all sample questions. Given an input question, this technique retrieves the most similar sample question, and we return the answer linked to that question.

To evaluate the systems we used the language database from the SGT Blackwell virtual human (see Section 5). The database contains 1,261 questions and 60 answer classes. We divided the collection of questions into training and testing subsets following the 10-fold cross-validation schema and calculated the effectiveness of each approach.

Table 1 shows the accuracy numbers for the two baselines (we call them "SVM" and "SLM") and the NPCEditor classification ("CLM"). The NPCEditor classification approach is 17% more accurate than the SVM baseline. The differences shown are statistical significant by t-test ($p < 0.05$).

Both SLM and CLM methods may return several candidate answers ranked by their scores. That way an interactor may get a different response if she repeats the question (if there is more than one good answer). We want to measure the quality of the ranked list of candidate answers or the proportion of appropriate answers among all the candidate answers, but we should also prefer the candidate sets that list all the correct answers before all the incorrect ones. A well-known IR technique is to compute average precision—for each position in the ranked list compute the proportion of correct answers among all preceding answers and average

SVM	SLM			CLM		
accuracy	accuracy	impr. over SVM	avg. prec.	accuracy	impr. over SVM	avg. prec.
53.13	57.80	8.78	63.88	61.99	16.67	65.24

Table 1: Comparison of three different algorithms for answer selection on SGT Blackwell data. Each performance number is given in percentages.

those values. We show the average precision numbers for the SLM and CLM runs. Both average precision and accuracy scores are significantly higher for the cross-language approach. We have repeated the experiment on 7 other virtual characters with smaller language databases. We observed that our system is more effective on problems with more answer classes.

Classifier Robustness

In the second set of experiments we evaluated the classifier robustness to input errors. Recall that in a typical VH system the text input to the classifier comes from an ASR module. The automatic speech recognition process can introduce errors into transcription. We were interested in the effect of the ASR quality on the answer quality. We recruited 20 participants to interview the SGT Blackwell character. Each participant asked 20 questions. We recorded and transcribed the questions using both ASR and human transcribers. We computed the Word Error Rate (WER) for each ASR transcription. The average WER score was 37.33%.

We applied the NPCEditor classification approach to both ASR and human transcribed data and recorded the selected answers. We asked three human raters to judge the appropriateness of the selected responses using a 1-6 scale (Gandhe et al. 2004). The Cronbach's alpha value measuring the inter-rater agreement was above 0.91 indicating high consistency among the judges.

For both datasets we computed the cumulative average appropriateness score (CAA) as a function of WER: for each WER value t we average the appropriateness scores for all questions-answer pairs with WER score less than or equal to t. This gives us the expected appropriateness score given that the ASR WER does not exceed the value t. We calculated the differences between CAA score for both datasets at different values of WER. These differences are small and not statistically significant until WER reaches 60%. After that point the CAA score is significantly lower (by t-test with $p < 0.05$) on the ASR transcribed data.

Interaction Quality

Kenny and his colleagues (Kenny, Parsons, and Rizzo 2009) study virtual humans for clinician training. They have built a virtual human using NPCEditor that plays a role of a patient with a psychiatric problem and they wanted to assess if a virtual patient would respond to the clinician interview as a real patient would. They wanted to see if 1) clinicians could elicit proper responses from questions relevant for an interview from a virtual patient and 2) to evaluate psychological variables such as openness and immersion of the participant and believability of the character as a patient. They have engaged 15 test subjects from a medical school including medical students, psychiatry residents and fellows. Each subject conducted a 15 minutes interview with the virtual patient trying to diagnose her condition and filled out a set of questionnaires before and after the interview. The researchers analyzed the data from the interview transcripts and from the questionnaires and found that the subjects were generally immersed in the interviews, they described the virtual patient character as believable and engaging, and they did ask and received responses covering all aspects of a typical patient interview. The study showed a feasibility of using virtual patients for training.

5 Application Development, Use and Payoff

NPCEditor has been used as the language processing component for over a dozen virtual humans at ICT (some with multiple versions), and several dozen elsewhere. Over a dozen different developers have so far used the system to create or extend characters. The system has been deployed and administered in museums, in virtual worlds, at trade shows and conferences, and in mobile vans by people not involved in their development. Thousands of people have interacted with these systems, and even more have seen them as audience to live interactions. In this section we describe some of the installations, highlighting their unique features.

SGT Blackwell Originally created as a showcase of VH technology, SGT Blackwell was introduced at the 2004 Army Science Conference. He is a life-size 3D US Army soldier projected onto a transparent screen in a mixed-reality environment. Conference attendees acted as audience with ICT demoers who spoke to SGT Blackwell. The original domain had 83 responses on different topics covering his identity, origin, language and animation technology, design goals, our university, the exhibition setup, and some miscellaneous topics, such as "what time is it?" and "where can I get my coffee?" After a lot of positive feedback from the attendees, several subsequent versions were built, using the NPCEditor. An extended version with additional domain items was used for demos both at ICT and by the office of the Director for Research and Laboratory Management at US Army, with external administrators and demoers. It was also selected as a part of the Smithsonian's National Design Triennial, Design Life Now exhibit in 2006. The system was installed at the Cooper-Hewitt Museum in New York from December 2006 to July 2007 (and later at two other museums), where SGT Blackwell was administrated by Museum staff and interacted directly with over 100,000 visitors. Limited versions of SGT Blackwell were also created specially for the Director for Research and Laboratory Management to interact with at the opening and closing of the 2006 Army Science conference, as well as a cameo appearance with SGT Star at the 2008 conference. An example of di-

alogue with SGT Blackwell can be found in (Leuski et al. 2006). Preliminary evaluation of Blackwell in the Cooper-Hewitt can be found in (Robinson et al. 2008).

SGT Star Interactive SGT Star was funded by the US Army Accessions Command, who wanted a mobile, life-sized, face to face version of their web character from goarmy.com[3]. SGT Star is like SGT Blackwell a life-size rendering of an Army soldier that answers questions on topics including Army careers, training, education and money for college. He can also handle queries about the technology behind his development and explain how his creation fits in with plans for future Army training environments. There are approximately 320 answers in his repertoire. The original version was used by Accessions command at trade shows and has since been ported to several "Army Adventure Vans" in which Army educational personnel interact with SGT Star about Science and Army careers. The character database was constructed by a linguist in our lab, with consultation from scriptwriters and Army SMES and it's administered and interacted with by the vans' Army staff. More about SGT Star, including a longitudinal evaluation at several conventions can be found in (Artstein et al. 2009).

Virtual Patients for Clinical Training Since 2006, the NPCEditor has been used to create virtual characters exhibiting psychological conditions who can interact verbally and non-verbally with a clinician in an effort to teach the clinician interpersonal skills such as interviewing and diagnosis. Three virtual patient characters were developed by a separate team at the institute without the direct involvement of the NPCEditor creators. Each character database contained up to 200 responses. Users were medical and psychology students (Kenny, Parsons, and Rizzo 2009) .

Army Communication Skills Training Since 2006 NPCEditor has been successfully used by the Program Executive Office (PEO) Simulation, Training, and Instrumentation (STRI), US Army, as a natural language understanding and processing component in a number of interactive training systems that teach soldiers communication and culture-specific conversational skills. We have received very positive feedback about NPCEditor from designers and developers of the training systems. These systems have been fielded in 19 training installations. As of this writing, more than 3,000 soldiers (commissioned and noncommissioned) have received training using the system. An independent analysis has shown that the US Army has achieved significant savings (as much as $30 million) in training systems research and development costs by reusing this existing system and have realized greater flexibility in the ability to respond to theater driven changing training requirements[4]. The designers and administrators are Army or contracted personnel outside ICT, and the interactors are soldiers, using the systems for training.

Virtual World Guides Since 2008, NPCEditor has been used to develop several AI avatars in online virtual worlds such as Second Life and Active Worlds. These characters are used for aides in educational settings as well as guides of the virtual space. These characters have been designed and administrated at ICT, but the interactors were people in the virtual worlds who came to visit the areas and interact with the characters. In contrast to the other characters described in this section, the online virtual world characters do not use speech recognition but the native virtual world chat and IM facilities. Probably the most advanced of these is LT Moleno, a staff duty officer, who patrols the US Army Welcome island in Second Life. He can answer questions about the island and conduct interactive tours of the island facilities. At the time of this writing approximately 4,000 visitors have interacted with officer Moleno. More details on LT Moleno can be found in (Jan et al. 2009).

Virtual Human Toolkit & Others NPCEditor is being used to create virtual humans in more and more diverse applications. NPCEditor is now part of a Virtual Human Toolkit that is a collection of modules, tools and libraries that allow developers to create their own virtual humans. The toolkit is available without cost for academic research purposes at our web site[5]. In September 2008 we have conducted a 3 day workshop, where approximately 30 attendees, mostly graduate students from universities across the country, designed and built 6 different characters for a game of "Clue" over two afternoons. Each character had approximately 30 to 40 responses. This illustrates how quickly a novice character designer can develop a useful virtual human. There are currently several other projects being developed or deployed that use NPCEditor. **Gunslinger** is an interactive-entertainment application of virtual humans that places the user inside a Wild West setting (Hartholt et al. 2009). The **InterFaces**[6] exhibit in the Boston Museum of Science contains a pair of virtual docents who can answer questions about computers, robots, and communications, as well as themselves and exhibits in the museum's Cahners Computer Place exhibit hall.

6 Conclusions

In this paper we presented NPCEditor, a system for building and deploying virtual characters capable of engaging a user in spoken dialog on a limited domain. NPCEditor has been used mainly for question answering characters where a interactor asks questions and the character responds. However other types of dialogue have also been successfully implemented, for example, the Gunslinger system, in which characters take the initiative and question the user. The dialog may have other forms as long as the character responses can be fully specified a priori.

NPCEditor contains a state of the art cross-language information retrieval-based classifier that is robust to noisy input from speech recognition results. It contains a development environment that includes a user-friendly GUI to support several classes of user, from developer to interactor and audience. NPCEditor has been successfully evaluated in the laboratory and field-tested and proved to be an effective and versatile system in a number of different applications.

[3]The website version was developed by NextIT and does not use the NPCEDitor or any shared technology with the ICT version.

[4]Personal communication. The report is not publicly available.

[5]http://vhtoolkit.ict.usc.edu/

[6]http://www.mos.org/interfaces/

Acknowledgments

We would like to thank the users of NPCEditor for many helpful suggestions that led to specific improvements in usability and functionality. We also would like to thank the reviewers for their invaluable suggestions and comments helping us to improve the paper. The effort described here has been sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

References

Allan, J.; Callan, J.; Croft, W. B.; Ballesteros, L.; Byrd, D.; Swan, R.; and Xu, J. 1998. Inquery does battle with TREC-6. In *Proceedings of the 6th Text REtrieval Conference*, 169–206.

Artstein, R.; Gandhe, S.; Gerten, J.; Leuski, A.; and Traum, D. R. 2009. Semi-formal evaluation of conversational characters. In Grumberg, O.; Kaminski, M.; Katz, S.; and Wintner, S., eds., *Languages: From Formal to Natural*, volume 5533 of *Lecture Notes in Computer Science*, 22–35. Springer.

Bahl, L. R.; Jelinek, F.; and Mercer, R. L. 1990. A maximum likelihood approach to continuous speech recognition. In *Readings in speech recognition*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. 308–319.

Blei, D. M.; Ng, A. Y.; Jordan, M. I.; and Lafferty, J. 2003. Latent dirichlet allocation. *Machine Learning Research* 3:993–1022.

Chu-Carroll, J., and Carpenter, B. 1999. Vector-based natural language call routing. *Computational Linguistics* 25(30):361–388.

Gandhe, S.; Gordon, A.; Leuski, A.; Traum, D.; and Oard, D. W. 2004. First steps toward linking dialogues: Mediating between free-text questions and pre-recorded video answers. In *Proceedings of the 24th Army Science Conference*.

Gandhe, S.; DeVault, D.; Roque, A.; Martinovski, B.; Artstein, R.; Leuski, A.; Gerten, J.; and Traum, D. 2008. From domain specification to virtual humans: An integrated approach to authoring tactical questioning characters. In *Proceedings of Interspeech Conference*.

Gorin, A.; Riccardi, G.; and Wright, J. 1997. How may i help you? *Speech Communication* 23:113–127.

Gratch, J.; Rickel, J.; Andre, E.; Cassell, J.; Petajan, E.; and Badler, N. 2002. Creating interactive virtual humans: Some assembly required. *IEEE Intelligent Systems* 54–63.

Grefenstette, G. 1998. *Cross-Language Information Retrieval*. Norwell, MA, USA: Kluwer Academic Publishers.

Hartholt, A.; Gratch, J.; Weiss, L.; Leuski, A.; Morency, L.-P.; Marsella, S.; Liewer, M.; Thiebaux, M.; Doraiswamy, P.; and Tsiartas, A. 2009. At the virtual frontier: Introducing Gunslinger, a multi-character, mixed-reality, story-driven experience. In *Proceedings of the 9th International Conference on Intelligent Virtual Agents*, 500–501.

Hofmann, T. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd International ACM SIGIR Conference*, 50–57.

Jan, D.; Roque, A.; Leuski, A.; Morie, J.; and Traum, D. R. 2009. A virtual tour guide for virtual worlds. In *Proceedings of the 9th International Conference on Intelligent Virtual Agents*, 372–378.

Joachims, T. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Machine Learning: ECML-98*. chapter 19, 137–142.

Johnson, W. L.; Vilhjalmsson, H.; and Marsella, M. 2005. Serious games for language learning: How much game, how much AI? In *Proceedings of the 12th International Conference on Artificial Intelligence in Education*, 306–313.

Katz, B. 1988. Using english for indexing and retrieving. In *Proceedings of the 1st RIAO Conference*.

Kenny, P. G.; Parsons, T. D.; and Rizzo, A. A. 2009. Human computer interaction in virtual standardized patient systems. In *Proceedings of the 13th International Conference on Human-Computer Interaction. Part IV*, 514–523.

Kopp, S.; Gesellensetter, L.; Krämer, N. C.; and Wachsmuth, I. 2005. A conversational agent as museum guide - design and evaluation of a real-world application. In *Proceedings of the 5th Intelligent Virtual Agents Conference*, 329–343.

Krovetz, R. 1993. Viewing morphology as an inference process. In *Proceedings of the 16th International ACM SIGIR Conference*, 191–202.

Lavrenko, V. 2004. *A Generative Theory of Relevance*. Ph.D. Dissertation, University of Massachusetts at Amherst.

Leuski, A., and Traum, D. 2008. A statistical approach for text processing in virtual humans. In *Proceedings of the 26th Army Science Conference*.

Leuski, A., and Traum, D. 2010. NPCEditor: A tool for building question-answering characters. In *Proceedings of The 7th International Conference on Language Resources and Evaluation*.

Leuski, A.; Patel, R.; Traum, D.; and Kennedy, B. 2006. Building effective question answering characters. In *Proceedings of the 7th SIGdial Workshop*.

Lewis, D. D.; Schapire, R. E.; Callan, J. P.; and Papka, R. 1996. Training algorithms for linear text classifiers. In *Proceedings of the 19th International ACM SIGIR Conference*, 298–306.

Ponte, J. M., and Croft, W. B. 1997. Text segmentation by topic. In *Proceedings of the ECDL Conference*, 120–129.

Rizzo, A.; Pair, J.; Graap, K.; Manson, B.; McNerney, P.; Wiederhold, B.; Wiederhold, M.; and Spira., J. 2006. A virtual reality exposure therapy application for iraq war military personnel with post traumatic stress disorder: From training to toy to treatment. *Nato Security Through Science Series E Human And Societal Dynamics* 6:235.

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 5th International Conference on Language Resources and Evaluation*.

Roque, A., and Traum, D. 2007. A model of compliance and emotion for potentially adversarial dialogue agents. In *Proceedings of the 8th SIGdial Workshop*.

Traum, D.; Swartout, W.; Gratch, J.; Marsella, S.; Kenney, P.; Hovy, E.; Narayanan, S.; Fast, E.; Martinovski, B.; Bhagat, R.; Robinson, S.; Marshall, A.; Wang, D.; Gandhe, S.; and Leuski, A. 2005. Dealing with doctors: Virtual humans for non-team interaction training. In *Proceedings of the 6th SIGdial Workshop*.

Tsochantaridis, I.; Hofmann, T.; Joachims, T.; and Altun, Y. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21st international conference on Machine Learning*.

Voorhees, E. M. 2003. Overview of the TREC 2003 question answering track. In *Proceedings of the 12th Text REtrieval Conference*, 54–69.