Quantitative Evaluation of User Simulation Techniques for Spoken Dialogue Systems

Jost Schatzmann
Engineering Department
University of Cambridge
Trumpington Street
Cambridge, England
js532@eng.cam.ac.uk

Kallirroi Georgila
School of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh, Scotland
kgeorgil@inf.ed.ac.uk

Steve Young
Engineering Department
University of Cambridge
Trumpington Street
Cambridge, England
sjy@eng.cam.ac.uk

Abstract

The lack of suitable training and testing data is currently a major roadblock in applying machine-learning techniques to dialogue management. Stochastic modelling of real users has been suggested as a solution to this problem, but to date few of the proposed models have been quantitatively evaluated on real data. Indeed, there are no established criteria for such an evaluation. This paper presents a systematic approach to testing user simulations and assesses the most prominent domain-independent techniques using a large DARPA Communicator corpus of human-computer dialogues. We show that while recent advances have led to significant improvements in simulation quality, simple statistical metrics are still sufficient to discern synthetic from real dialogues.

1 Introduction

Within the broad field of research on spoken dialogue systems (SDS), the application of machine-learning approaches to dialogue management is currently attracting interest (Levin et al., 2000) (Young, 2002). The major motivation driving research in this area is the hope of learning optimal strategies from data. Yet, it is rarely the case that enough training data is available to sufficiently explore the vast space of possible dialogue states and strategies. Ironically, the best strategy may often not even be present in the given dataset. It may thus be argued that an optimal strategy cannot be learned from a fixed corpus, regardless of the size of the training corpus.

An interesting approach to solving this problem is to use small corpora to train stochastic models for simulating real user behavior. Once such a model is available, any number of dialogues can be generated through interaction between the simulated user and the dialogue system. The simulated user also enables us to explore dialogue strategies that are not present in the given corpus. This way the learning dialogue manager can deviate from the known strategies and learn new and potentially better ones. Figure 1 illustrates the learning setup.

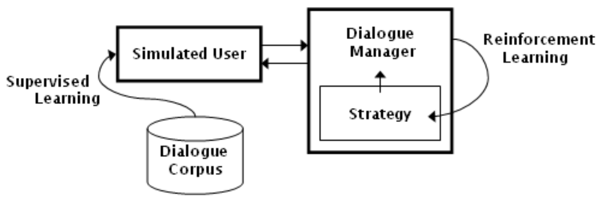


Figure 1: Strategy learning using a simulated user

Previous research has demonstrated the success of the learning setup (Levin et al., 2000), (Scheffler, 2002) and also examined the use of user simulation for system evaluation (Eckert et al., 1997). The quality of the user model, however, has not been thoroughly investigated. It is indeed unclear, how we can quantitatively evaluate whether the simulated user responses are realistic, generalise well to unseen dialogue situations and resemble the variety of the user population.

This paper assesses the most prominent domain-independent simulation techniques using a large DARPA Communicator corpus of human-computer dialogues. We describe what modifications are necessary to train and test the models presented in the literature on real data. We further present a systematic approach to evaluating user simulations. Our analysis shows that none of the currently available techniques can realistically reproduce the variety of human user behaviour and that simple statistical measures are sufficient to distinguish synthetic from real dialogues. We investigate these shortcomings and outline suggestions for future research.

This paper is supported by the EU FP6 TALK Project.

2 User Simulation in SDS

2.1 Intention-based dialogue

Approaches to user simulation can be classified in a number of ways. Most commonly, one distinguishes systems with regard to the level of abstraction at which they model dialogue. This can be at either the acoustic-, word-, or intention-level. The latter is a particularly useful representation of the interaction, since it avoids the need to reproduce the enormous variety of human language on the level of speech signals or word sequences.

Hence, simulation on the intention level has been most popular in recent years. This approach was first taken by (Eckert et al., 1997) and has been adopted in later work by most other research groups (Levin et al., 2000), (Scheffler, 2002), (Pietquin, 2004), (Georgila et al., 2005a). Examples of user simulation on the word or acoustic level are rare, but can be found in (Watanabe et al., 1998) and (Lopez-Cozar et al., 2003). Naturally, their portability and scalability is limited.

2.2 Probabilistic vs. Deterministic Simulation

One may further distinguish between probabilistic and deterministic user models. Whereas probabilistic models can be trained on data and allow for some "lifelike" randomness in user behaviour, deterministic models are driven by handcrafted rules. For a given dialogue state and system action, a deterministic user model will always produce the same user response.

Deterministic models have been used to evaluate which dialogue strategies work well for different types of user response pattern (Lin and Lee, 2001). While they may be suitable for observing general correlations between dialogue strategy, user behaviour and system performance, a probabilistic model is clearly preferable for modelling realistic user behaviour. The following sections will review some of the most prominent work in this area. Very recent work by (Georgila et al., 2005a) is not covered here.

2.3 The Bigram Model

Stochastic modelling of users on the intention level is first suggested as a means of SDS evaluation by Eckert, Levin and Pieraccini (Eckert et al., 1997). Their work introduces a Bigram model for predicting the user action a_u in response to a given system action a_s

$$p = P(a_u|a_s). \tag{1}$$

The Bigram model has the advantage of being purely probabilistic and fully domain-independent. Its weakness is that it does not place enough constraints on the user to simulate realistic behaviour. The generated responses may correspond well to the previous system action, but they often do not make sense in the wider context of the dialogue. The authors note that the model can be extended to a general n-gram model but due to data sparsity, it is usually impossible to train n-grams with $n > 2$.

Eckert et al. do not train the Bigram model on real data or evaluate the quality of the simulated output.

2.4 The Levin Model

Levin, Eckert and Pieraccini describe how the pure Bigram model can be modified to limit the number of model parameters and to account for some degree of conventional structure in dialogues (Eckert et al., 1997), (Levin et al., 2000). Instead of allowing any user response, only the probabilities for anticipated types of user responses are calculated for each type of system action. A system request for attribute A_x, for instance, is parameterised using the probability that the user actually specifies A_x and that he specifies n additional attributes

$$P(provide\ A_x|request\ A_x) \tag{2}$$

$$P(n|request\ A_x). \tag{3}$$

This set of probabilities implicitly characterises the level of cooperativeness and the degree of initiative taken by the user model. The Levin model places stronger constraints on the user actions than the pure Bigram model, but it also makes assumptions concerning the format of the dialogue. If the dialogue manager or the anticipated dialogue format changes, a new set of parameters is needed.

Like the Bigram model, the Levin model does not ensure consistency between different user actions over the course of a dialogue. The assumption that every user response depends only on the previous system turn is flawed. The user actions can violate logical constraints and the synthetic dialogues often continue for a long time, with the user continuously changing his goal or repeating information.

Levin et al. use the ATIS corpus to train a small subset of their model parameters, all other probabilities are handcrafted using common sense. The authors also do not evaluate how realistically simulated the responses are. However, the authors demonstrate that the simulated user can be used to reveal errors in the dialogue management strategy (Eckert et al., 1997) and that it can be used for reinforcement-learning of strategies (Levin et al., 2000).

2.5 The Scheffler Model

Scheffler and Young (Scheffler and Young, 2001) (Scheffler, 2002) attempt to overcome the lack of goal consistency that the Levin model suffers from. Their approach uses deterministic rules for goal-dependent actions and probabilistic modelling to cover conversational behaviour.

To model the user goal, Scheffler and Young introduce fixed goal structures. These consist of attribute-value pairs with associated status variables. All of the possible "paths" that a user may take during a dialogue are mapped out in advance in the form of a network. The probability of each route through the network is learned from training data and the explicit representation of the user goal ensures that the simulated user always selects routes in accordance with his goal. Scheffler and Young's approach produces promising results, but it is highly task-dependent and ideally requires an existing prototype system.

The authors address the problem of evaluating the simulated user by comparing statistical properties of the simulated dialogues with those of the training data dialogues. More precisely, they show that the goal-completion time and goal-achievement rate for different tasks are comparable in the simulated and real dialogues.

2.6 The Pietquin Model

Pietquin (Pietquin, 2004) combines features from Scheffler and Young's work with the Levin model. The core idea is to condition the probabilities used by Levin et al on an explicit representation of the user goal

$$P(provide\ A_x | request\ A_x,\ \ goal). \qquad (4)$$

This enables Pietquin to explicitly model the dependencies between a user's actions and his goal. Pietquin hand-selects the probability values so as to ensure that the user acts are in accordance with his goal throughout the dialogue.

Like Scheffler and Young, Pietquin represents the user goal using a simple table of attribute-value pairs. The appropriate values are randomly selected from a database.

Pietquin introduces interesting dependencies between the user's goal and his conversational behaviour. This is done by adding new status variables to each attribute-value pair. The priority variable for instance, governs how likely the user is to drop the corresponding attribute-value pair from his goal. This enables Pietquin to model how likely a user is to relax a certain constraint, such as "Preferred airline is British Airways". Pietquin also attaches a simple counter to each attribute-value pair to record how often a piece of information has been transmitted to the system. The likelihood of the user hanging up before completing the task can be modelled as a function of this variable.

While these models of user goal, memory and satisfaction are rather coarse, they illustrate the various aspects of the user state which influence behaviour. It is also important to note that Pietquin's model is domain-independent - a definite advantage over Scheffler and Young's work.

A major weakness of Pietquin's work however is that it is not trained or tested on any real dialogue data. All the probabilities in his model are hand-selected using common sense, and no attempt is made to evaluate how realistic the user simulation is. Pietquin shows that an equivalent representation of his user model can be found in the form of a Bayesian Network, but the parameter values for this network are also copied from the original model rather than learned.

3 Evaluating User Simulation

3.1 Overview

The previous section has reviewed a number of different user simulation techniques. To date, few of these have been evaluated on real data. In part this is due to the lack of a suitable evaluation methodology. It is indeed not clear what constitutes a "realistic" simulation.

In our view, evaluation must cover two aspects. First, we need to assess if the user model can generate human-like output. Does it produce responses that a real user might have given in the same dialogue context? Secondly, we need to assess if the simulation can reproduce the variety of real user behaviour. This ensures that the model represents the whole user population - not just an average user.

3.2 Comparing Simulated and Real User Responses

For the first part of the evaluation, the dataset is split into a training and a test set. The dialogues are assumed to be annotated as a sequence of turns t, with each turn consisting of a variable number of actions a, as shown in the sample dialogue in Table 1.

Evaluation is done on a turn by turn basis. Each of the system turns in the test set is separately fed into the simulation, together with the corresponding dialogue history and the current user goal. The response turn generated by the simulated user is then compared to the real response given by the user in the test set.

We propose the use of Precision and Recall to quantify how closely the synthetic turn resembles the real user turn. These metrics have not yet been used for user model evaluation in SDS development but they are a common measure of goodness in user modelling outside SDS. (Zukerman and Albrecht, 2001). Recall (R) measures how many of the actions in the real reponse are predicted correctly. Precision (P) measures the proportion of correct actions among all the predicted actions. An action is considered correct if it matches at least one of the actions in the real user response.

$$P = 100 * \frac{Correctly\ predicted\ actions}{All\ actions\ in\ simulated\ response} \qquad (5)$$

$$R = 100 * \frac{Correctly\ predicted\ actions}{All\ actions\ in\ real\ response} \qquad (6)$$

It is of course not possible to specify what levels of Precision and Recall need to be reached in order to claim that a simulated user is realistic. Nevertheless, Precision and Recall offer a reliable method for comparing simulated and real user responses.

3.3 Comparing Simulated and Real Datasets

Precision and Recall deliver a rough indication of how realistic the *best* response is that the simulated user can generate. On its own however, this form of evaluation is not sufficient. Our goal is not to build a simulated user for producing the single most likely response to a given system action. A dialogue strategy must perform well for all kinds of possible user response, not just the one with the highest probability. Hence we need to produce a large number of dialogues with a variety of user behaviour. We then need to assess if the synthetic dataset has the same statistical properties as the training data set.

The difficult question is: "What statistical properties are reliable indicators of realistic dialogues?". In previous research, dialogue length, goal achievement rate and goal completion length have been used (Scheffler and Young, 2001). These metrics can only be considered rough indicators of how realistic the dialogues are. It would be possible to optimise a user model according to these criteria and still produce non-sense dialogues. For instance, given that the average dialogue length found in the training data was n turns, the simulated user could be forced to hang up after exactly n turns, thus achieving a perfect evaluation score.

We argue that a large set of measures is needed to cover a variety of dialogue properties. For our evaluation, we divide these into three groups:

1. The first group of experiments investigates high-level features of the dialogue. How long do the dialogues last and how much information is transmitted in individual turns? How active are the dialogue participants?

2. The second group of experiments analyses the style of the dialogue. This aims to produce a more fine grained picture of the system and user behaviour. We investigate the frequency of different speech acts and analyse what proportion of actions is goal-directed, what part is taken up by dialogue formalities etc. We also examine the user's degree of cooperativeness.

3. The third and last group of experiments investigates the success rate and efficiency of the dialogues. In particular, we look at goal achievement rates and goal completion times. This helps us to evaluate if misunderstandings are modelled well.

In closing this section, it should be remarked that all of the statistical measures suggested here are only indicators of how good a simulation technique is. It is not possible to specify what range of values a synthetic corpus needs to satisfy in order to be sufficiently realistic. Moreover, no guarantee can be given that a simulated dialogue is realistic even if all of its properties are identical to the training data.

Yet, the set of measures forms a helpful toolkit for comparing simulation techniques and identifying possible weaknesses. The tests cover dialogue length, style and efficiency. In addition, the variety of measures is sufficiently large to ensure that a user model cannot be easily trained so as to achieve perfect scores on each of them.

4 Experimental Setup

4.1 Training and Testing Data

Data from the DARPA Communicator project is used for all of of the experiments presented in this paper. The full corpus consists of 4 datasets, recorded using systems from ATT, BBN, CMU and SRI. The 4 sets add up to a total of 697 dialogues. Each of the four sets is split into training and testing data, with a ratio of 90:10. Further information regarding the content and annotation of data can be found in (Georgila et al., 2005b).

All of the datasets contain slot-filling dialogues from the travel booking domain, covering flight-, hotel- and rental car-reservations. The dialogue systems differ slightly in the wording of their prompts and in their choice of dialogue strategy and the language understanding components are not equally powerful. On the intention level however, the general structure of the dialogues is very similar. The systems cover roughly the same booking details and they are all almost entirely driven by system-initiative.

User model training is done on the recognised user output rather than the reference transcriptions. The simulation thus effectively combines the user and the communication channel. No separate error modelling is performed.

4.2 Dialogue Annotation

The dialogue data is automatically converted to the following format: Each dialogue is a sequence of alternating user and system turns. Each turn t contains one or more actions. Each action a consists of a speech act (compulsory), an attribute (optional) and a value (optional). A snippet of a sample dialogue is shown in Table 1.

We also add "hangup" actions to the end of each dialogue. Considering the act of "hanging up" as an action, helps us to train user model parameters concerning the likelihood of a user hanging up in a given dialogue situation.

4.3 Predicting Attribute Values

For the purpose of our evaluation, we implement and train the Bigram, Levin and Pietquin models. None of these

Turn	Spkr.	Actn.	Speech act, Attribute, Value
t_1	Sys	a_1	greeting
		a_2	request_info orig_city
t_2	User	a_3	provide_info orig_city boston
t_3	Sys	a_4	implicit_confirm orig_city oslo
		a_5	request_info dest_city
t_4	User	a_6	no_answer
		a_7	provide_info orig_city boston
t_5	Sys	a_8	apology
		a_9	explicit_confirm orig_city boston
t_6	User	a_{10}	yes_answer

Table 1: Sample dialogue

models has been fully applied to real data before and we found that a number of modifications were necessary to be able to actually train the models. The Bigram model and the Levin model, for example, include the prediction of attribute values in the user model. This means that different probabilities are estimated for user actions with different values, say, $provide\_info\ \ orig\_city\ \ london$ and $provide\_info\ \ orig\_city\ \ boston$.

We found that this approach led to severe data sparsity problems when applied to a real corpus. Datasets such as the Communicator corpus contain a large number of possible values for each attribute. The number of possible combinations of user action, attribute and value prohibits us from reliably estimating a probability for each one.

It is thus not possible to implement the Bigram model and the Levin model in their original form. For this evaluation we choose to adapt both models in the following way: The speech act and attribute are modelled probabilistically, as suggested by the respective authors. The attribute-value is determined by the user goal, as suggested by Scheffler and Young. This ensures that sufficient training data is available to train all model parameters. It further improves the model as it ensures that the same value is provided if the user is asked multiple times for the same attribute. See Figure 2 for an illustration.

To allow for "lifelike" randomness in the user goal, we use a probabilistic domain model. At the start of each dialogue, a user goal is randomly constructed according to the probability distribution over all the attributes and values found in the training data. We use this domain model for all of our simulated users. Testing is not done on the attribute-value, only on the speech act and the attribute.

4.4 Bigram Model Implementation

The original Bigram model as described by Eckert et al. assumes that dialogue is a sequence of alternating user and system actions. Under this assumption, the next user action is predicted based on the previous system action.

In real dialogues, dialogue turns can include several

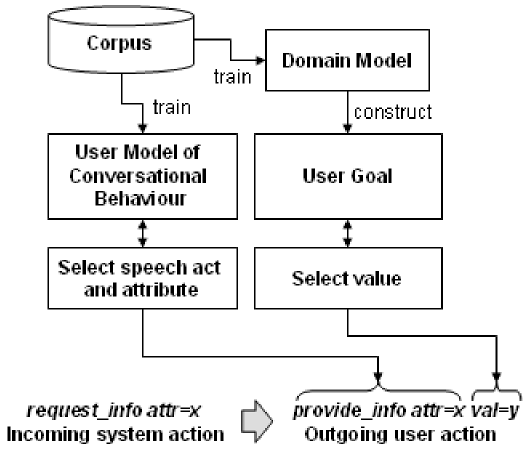


Figure 2: Response generation

actions. The assumption that action a_i can be predicted from a_{i-1} is hence no longer valid. The sample dialogue in Table 1 illustrates this well: action a_4 triggers action a_6, which in turn triggers action a_8.

However, it is also not possible to estimate "turn bigrams", i.e. estimate $P(t_i|t_{i-1})$ instead of $P(a_i|a_{i-1})$. Since the number of actions per turn is variable, the number of possible turn combinations will inevitably cause data sparsity problems.

For our implementation, we choose the following workaround: Bigrams are still estimated on an "action" basis, but the probability $P(a_u|a_s)$ is interpreted as the probability that the user response *contains* a_u when the previous system turn *contains* action a_s.

We further implement a simple back-off mechanism to account for system actions that appear in the test data but have not appeared in the training data. For these actions, no bigram is trained during parameter estimation. In these cases, we back off to the unigram probability of each user action.

4.5 Levin Model Implementation

The Levin model has to be adapted to the dialogue format present in the Communicator data. Relaxing questions ("Would you also consider another airline?"), for example, were anticipated by Levin et al. but do not exist in the Communicator data. Instead the dialogue managers spend a considerable amount of time on grounding (implicitly or explicitly confirming pieces of information).

To account for these differences in the system action set, we parameterise the Levin user model using a slightly modified set of probabilities. A positive response to an explicit confirmation of attribute A_x, for instance, is parameterised as

$$P(yes\_answer|explicit\_confirm\ \ A_x). \qquad (7)$$

Similar modifications are made for the other user and system actions that were not present in the dialogue data available to Levin et al.

4.6 Pietquin Model Implementation

As described in Section 2.6, the Pietquin model is an extension of the Levin model. The core idea of Pietquin's work is to condition the user model parameters on the user goal. In a real dataset, however, it is not possible to estimate a probability for every conceivable configuration of user goal. The number of possible combinations of user actions and user goals is far too large to obtain reliable probability estimates.

Our workaround for this problem is to condition the probabilities on selected properties of the user goal, rather than its full state. For instance, we check if an attribute is present in the goal or not, or if it has been provided before or not. This geatly reduces the number of parameters and avoids data sparsity problems.

4.7 User Goal Inference

The data available to us did not contain annotations regarding the specifics of the user goals. We were able to automatically infer these by scanning the parsed reference transcriptions of the user utterances. For every $provide\_info$ action, the corresponding attribute and value were added to the user goal. When two actions contradicted each other (i.e. same attribute, but different value) the later one was assumed to overwrite the earlier one. Counts were recorded to track how often each piece of information had been transmitted to the system over the course of the dialogue.

As explained by (Scheffler and Young, 2001), the automatic inference of user goals from dialogues is not unproblematic. The true user goal can never be known since the achieved goal may not be the one that the user started out with. It is impossible to ascertain which goals are indeed completed correctly and which are flawed by recognition errors. User goals may also change as users become aware of system limitations.

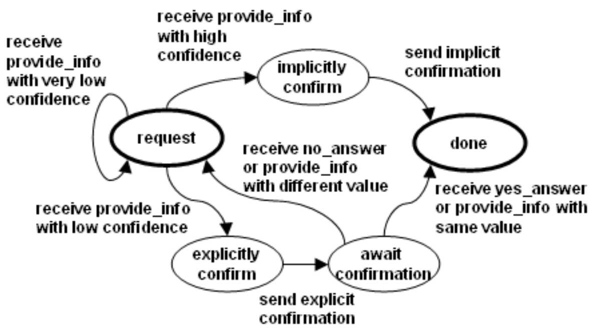
4.8 Dialogue Manager Implementation

To generate dialogues, the simulated user needs "a dialogue partner" to interact with. The straightforward strategy would be to take one of the original dialogue managers from ATT, BBN, CMU or SRI. Since none of these was available to us, the only alternative was to implement a new dialogue manager (DM). To make full use of the 4 training datasets, we chose to build a DM which is "an average" of the four original dialogue managers.

The new DM includes the features which are common to all of the original dialogue managers and it structures the dialogue in a similar way. Like all the original managers, the new DM covers flight bookings (origin and destination city, departing date and time, return flight) and ground arrangements (hotel location, hotel chain, car rental).

The new DM can process any of the user speech acts present in the data. This includes yes_answer and no_answer actions, which need to be correctly resolved according to the dialogue context. The DM can also handle user-initiative, i.e. process multiple pieces of incoming information. To resemble the dialogue managers in the training data, however, it does not encourage user initiative. Each DM turn contains at most one $request\_info$ action.

For each slot, the DM uses a simple state machine. The state of the slot informs the dialogue manager what action to take next to fill and confirm the slot. As can be seen in Figure 3, the DM can reject, implicitly confirm or explicitly confirm incoming information, based on the confidence score of the incoming action. Confidence scores for each user action are randomly selected from a flat distribution. The threshold levels for rejection, implicit or explicit confirmation can be set so that their relative proportion resembles that found in the training data. The behaviour of the dialogue manager involves no actual access to flight, hotel or car booking systems. Since all interaction occurs on the intention level, no database retrieval needs to be implemented.



receive provide_info with high confidence
receive provide_info with very low confidence
implicitly confirm
send implicit confirmation
request
receive no_answer or provide_info with different value
done
receive provide_info with low confidence
receive yes_answer or provide_info with same value
explicitly confirm
await confirmation
send explicit confirmation

Figure 3: Dialogue manager agenda

With regards to evaluation, it is difficult to quantify the effect of using a new DM on the quality of the simulated dialogues. Quite clearly, if the new DM behaves very differently from the original DM that was used for collecting the training data for the simulated user, then the synthetic data can never match the real data exactly - no matter how good the simulated user is. Since the training data is recorded with many different dialogue managers, it is also questionable if a single DM can generate the same variety of dialogues.

The fact that the training data is recorded using 4 different DMs is a great advantage for us. It enables us to quantify how much user behaviour can vary due to differences in experimental setup and dialogue strategy. By comparing the four original DMs, we can sketch out a target range for our simulated dialogues.

5 Evaluation Results

As explained in section 3, the evaluation is split into two main parts. The first part compares simulated user responses to real user responses in an unseen test set. This assesses how realistic the best response is that the simulated user can predict. The second part compares corpora of simulated dialogues to real corpora. This evaluates how well the simulation covers the variety of user behaviour in the training data.

5.1 Evaluation of the Best Response

As described in Section 3.2, we use Precision and Recall to measure the similarity between simulated and real user responses. The results (Table 2) show that the scores significantly improve from the Bigram to the Levin model. It is interesting to note that the jump in precision clearly exceeds the jump in recall. This is due to the fact that the Bigram model outputs a much greater number of user actions than the Levin model. We will confirm and discuss this problem in more detail later. The Pietquin model

	Train		Test	
	Precision	Recall	Precision	Recall
BIG	19.74	24.11	17.83	21.66
LEV	43.11	35.07	37.98	31.57
PTQ	45.00	36.35	40.16	33.38

Table 2: Precision and recall scores

outperforms both the Bigram and the Levin model. Its improvement over the Levin model is notable, but not as dramatic as the gap between the Bigram and the Levin model. This is natural, considering that the Pietquin model may be viewed as an extension of the Levin model.

The relative ranking of the three models is as expected: As the level of sophistication rises, the performance improves. Also as expected, the training data performance is slightly better than the test data performance.

5.2 Evaluation of the Generated Corpus

In the second part of the evaluation, we test how well the user models cover the variety of the user population in the training data. A corpus of 150 dialogues is generated with each of the user models through interaction with the dialogue manager (DM). The statistical distribution of the synthetic corpus is then compared to the training data, as described in Section 3.3.

As described in Section 4.8, our evaluation experiments are run with a DM that is different from the one used to collect the training data for the simulated user. It is interesting to investigate what effect the dialogue manager has on user behaviour. We therefore show individual measurements for each of the four datasets as well as the full training corpus (denoted by "ALL"). The range of values spanned by the four DMs is the *target range* for the simulated dialogues. Variations within this range can be attributed to the dialogue manager and the experimental setup.

5.2.1 High-level Dialogue Features

The first group of experiments covers the following statistical dialogue properties:

- Dialogue length, measured in the number of turns per task: mean, variance and shape of distribution
- Turn length, measured in the number of actions per turn: mean, variance and shape of distribution
- Participant activity as a ratio of system and user actions per dialogue

Figure 4 shows the mean values for dialogue length (= task length) and turn length. The Pietquin model achieves a very good result for dialogue length, missing the mean length of the training data by less than 2 turns. The Levin model is further away from the training data result, but it is still within the target range. The Bigram model performs very badly - the dialogues finish far too early. Analysis of the simulated dialogues shows that the user is very uncooperative, causing the system to finish the dialogue before completing any booking. This will also be confirmed by very low goal completion rates later in the evaluation.

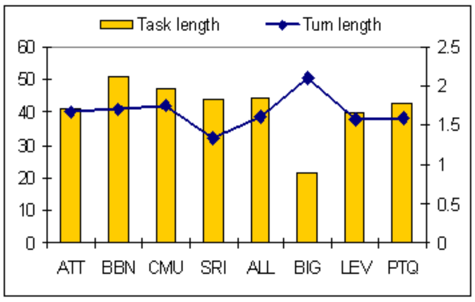


Figure 4: Mean task and turn length

We found that the standard deviation of the task length is too small in all of the simulated datasets. The shape of the distributions (Figure 5) confirms this. The curves for the Levin model and the Pietquin model look better than for the Bigram model, but their tails are still too flat.

Interestingly, the results for turn length look better. As shown in Figure 4 the means of the simulated datasets and the training data are nicely aligned. Only the Bigram model produces far too many actions per turn. The flaw

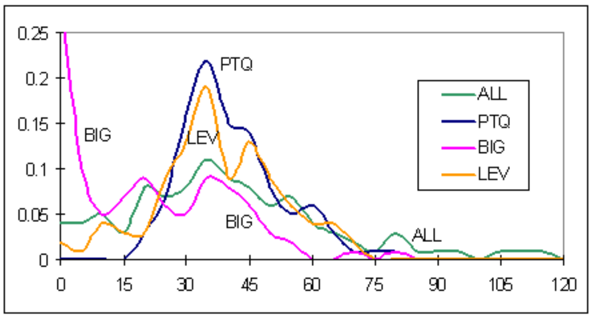


Figure 5: Task length distribution

leading to this problem is the assumption that each system action triggers exactly one user action. In real dialogues the relationship is not necessarily 1 to 1. An open question such as ”How may I help you?”, for instance, can lead the user to respond with several pieces of information. An implicit confirmation or an apology, on the other hand, may trigger no user response at all. The latter case is very common in real dialogues, leading to a lower average number of actions per turn.

The Levin model and the Pietquin model achieve almost perfect results for the standard deviation of the turn length. Looking at the shape of their distributions (Figure 6), we can see that they closely resemble the shape of the training data distribution.

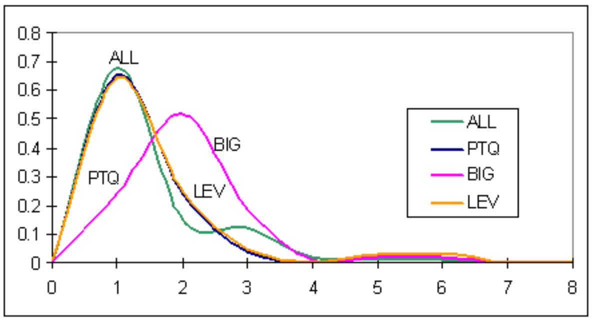


Figure 6: Turn length distribution

The next experiment investigates dialogue participant activity. Figure 7 shows the ratio of user vs. system actions. The lower part of the bar indicates the percentage of user actions while the upper part represents system actions.

Once again, the Bigram model is far outside the target range. As confirmed by the previous experiment, the user is ”talking too much”. The Levin and the Pietquin model achieve almost identical scores. Both models are inside the target range and not far from the training data result.

5.2.2 Dialogue Style and Cooperativeness

The next group of experiments covers the following statistical properties:

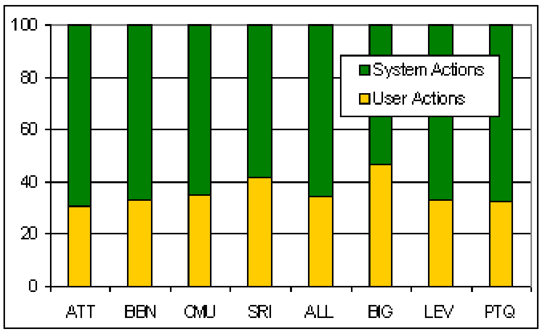


Figure 7: Ratio of user vs. system actions

- Frequency of different user and system speech-acts (average number of occurrences per dialogue)
- Proportion of goal-directed actions (request and provide information) vs. grounding actions (explicit and implicit confirmations) vs. dialogue formalities (greetings, apologies, instructions) vs. unrecognised actions (unknown).
- Number of times a piece of information is requested, provided, re-requested and re-provided in each dialogue

The histogram in Figure 8 shows the frequency of the most dominant user and system speech acts. The first three bins cover user actions: ”provide_info”, ”yes/no_answer” and ”unknown”. The last two bins are for system actions: ”request_info” and ”explicit/implicit_confirm”.

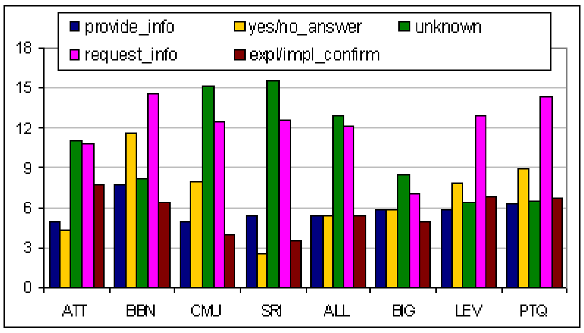


Figure 8: Histogram of speech acts

Secondly, we group all user and system actions into categories, as shown in Figure 9. This allows us to investigate what proportion of the dialogue is spent on goal-directed actions, grounding actions and dialogue formalities. Since the number of unrecognised actions is high, a separate category is created for these actions. Hang-up actions are not included in this analysis, since every dialogue contains exactly one hang-up action.

Our analysis shows that the relative ordering of actions is fairly similar for the four real systems. In the simulated datasets, the number of "unknown" user actions is clearly too low. This indicates that misunderstandings are not simulated well. At the same time, the proportion of goal-directed actions is too low compared to grounding actions and dialogue formalities.

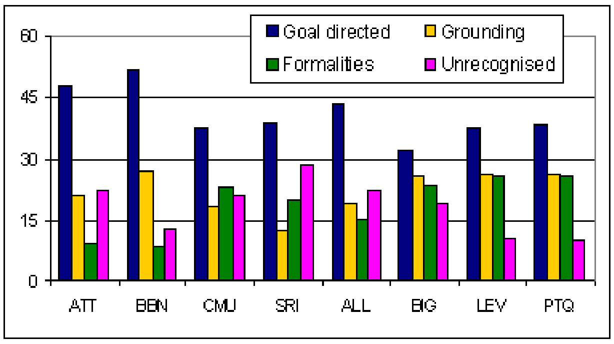


Figure 9: Proportion of dialogue spent on goal-directed actions, grounding actions, dialogue formalities, unrecognised actions. All bars show the percentage of actions in the corresponding class, i.e. the four bars add up to 100%

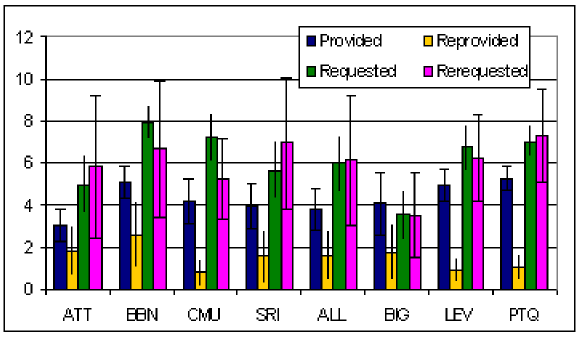


Figure 10: Dialogue efficiency (thin lines show std. deviation)

To evaluate the cooperativeness of the simulated user, we examine how often attributes are requested, provided, re-requested and re-provided per dialogue (Figure 10). The results confirm that the simulated user in the Bigram model is too active: The ratio between $provide\_info$ and $request\_info$ actions is tilted towards the user actions. The Levin and the Pietquin models show a rather large number of $provide\_info$ and $request\_info$ actions. The ratio between system requests and corresponding user responses, however, is very similar to the training data. This shows that the degree of user cooperativeness is modelled fairly well.

5.2.3 Dialogue Success Rate and Efficiency

The final group of experiments covers the following statistical properties:

- average goal / subgoal achievement rate
- mean and variance of the goal completion time

Figure 11 shows the goal achievement rates and goal completion times for the four real systems and the three simulated systems. We are only showing the results for flight-bookings here, but a similar analysis can be done for hotel-reservations and rental-car bookings. We have assumed that a subgoal is completed when the system acknowledges the corresponding booking.

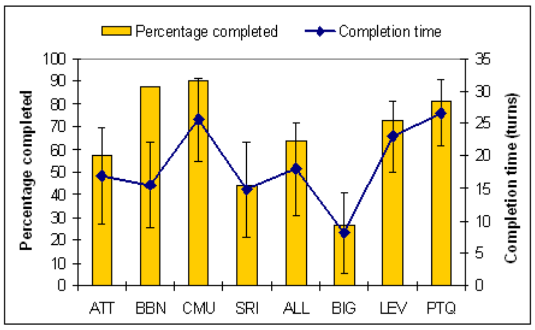


Figure 11: Flight goal completion rates (percentage of dialogues with successfully completed subgoal) and completion times (in dialogue turns, thin lines indicate standard deviation).

As expected, the Bigram model produces very poor results. The performance of the Levin and the Pietquin model is more interesting. Our analysis shows that the simulated users more frequently achieve their goals, but that the average completion time is longer. A possible explanation for this may be that the user's level of persistence and patience is not modelled well. Real user's seem to be more likely to hangup if the dialogue progress is slow.

Another plausible explanation is that real users can be roughly divided into a large group of novices and a small group of experts. The latter group is aware of system limitations and completes the dialogue successfully and quickly. The novices, on the other hand, tend to engage in long, error-prone dialogues that do not lead to successful completion. This dependency between user expertise and user behaviour is not accounted for in our implementations.

Analysis of the dialogue transcriptions shows that many real users produce special requests, such as "a window seat on the plane" or "rental car-insurance". This group of novice users appears to be underrepresented in the simulated datasets. Our experiments confirm this: "special requests" are usually parsed as "unknown" and this type of action is significantly less frequent in simulated datasets.

Interestingly, the Pietquin model performs worse than the Levin model for the goal completion metrics, although it explicitly takes the user goal into consideration. It appears that the model in its current form is too constraining. The assumption that the user goal stays fixed over the course of a dialogue is not correct. Secondly, the Pietquin model encourages the user not to mention attributes which are not part of his goal. While this is conceptually correct, it seems to have negative effects on user behaviour when the goal representation cannot capture the complexity of real user goals.

Manual analysis of the simulated dialogues also shows that the first phase of the dialogue (greeting, instruction, exchange of flight booking details) is fairly realistic, presumably because it follows dialogue conventions which are modelled well by Levin and Pietquin. The second phase (modification of booking details, re-retrieval of suitable flights, etc.) is less realistic, possibly because it is more strongly driven by the user goal.

6 Discussion and Future Work

This paper has presented a detailed evaluation of the most prominent domain-independent approaches to stochastic user simulation based on a large corpus of real human-computer dialogues. The nature of the simulation problem is such that no single measure of goodness exists, but we have demonstrated that a set of metrics can be used to identify the strength and weaknesses of each method.

Our results show that the works of Levin and Pietquin have led to good improvements in user simulation quality. Both approaches clearly outperform the Bigram baseline. However, the results also show that the simulated datasets can still be distinguished from real datasets using simple statistical metrics. Our analysis indicates that it may beneficial to distinguish between different user groups, for instance by training multiple user models with (say) different levels of expertise. Further research is also needed on modelling user goals and on modelling dialogue misunderstandings. We hope to address these problems in future work.

We believe that it may be particularly beneficial to develop a better representation of the user goal. To cover realistic dialogues, we must acknowledge that user goals can have hierarchical structures - and that these structures can evolve over time. The Hidden Vector State Model (Young, 2002) has recently been introduced as a method for learning hierarchical dependencies and we intend to investigate its use for user modelling.

References

W. Eckert, E. Levin, and R. Pieraccini. 1997. User modelling for spoken dialogue system evaluation. In *Proc. of ASRU '97*, pages 80–87.

K. Georgila, J. Henderson, and O. Lemon. 2005a. Learning user simulations for information state update dialogue systems. *Submitted to Eurospeech '05.*

K. Georgila, O. Lemon, and J. Henderson. 2005b. Automatic annotation of COMMUNICATOR dialogue data for learning dialogue strategies and user simulations. In *Proc. of DIALOR '05 (to appear).*

E. Levin, R. Pieraccini, and W. Eckert. 2000. A stochastic model of human-machine interaction for learning dialog strategies. *IEEE Trans. on Speech and Audio Processing*, 8(1):11–23.

B.-S. Lin and L.-S. Lee. 2001. Computer-aided analysis and design for spoken dialogue systems based on quantitative simulations. *IEEE Transactions on Speech and Audio Processing*, 9(5):534–548.

R. Lopez-Cozar, A. de la Torre, J. Segura, and A. Rubio. 2003. Assessment of dialogue systems by means of a new simulation technique. *Speech Communication*, 40:387–407.

O. Pietquin. 2004. *A Framework for Unsupervised Learning of Dialogue Strategies*. Ph.D. thesis, Faculte Polytechnique de Mons.

K. Scheffler and S. J. Young. 2001. Corpus-based dialogue simulation for automatic strategy learning and evaluation. In *Proc. NAACL Workshop on Adaptation in Dialogue Systems*, pages 64–70.

K. Scheffler. 2002. *Automatic design of spoken dialogue systems*. Ph.D. thesis, Cambridge University.

T. Watanabe, M. Araki, and S. Doshita. 1998. Evaluating dialogue strategies under communication errors using computer-to-computer simulation. *Trans. of IEICE, Info Syst.*, E81-D(9):1025–1033.

S. Young. 2002. Talking to machines (statistically speaking). In *Proc. of ICSLP '02*. Denver, Colorado.

I. Zukerman and D. Albrecht. 2001. Predictive statistical models for user modeling. *User Modeling and User-Adapted Interaction*, 11:129–158.