

Evaluating user simulations with the Cramér-von Mises divergence

Jason D. Williams

*AT&T Labs – Research
180 Park Avenue, Florham Park, NJ 07932, USA*

Abstract

User simulations are increasingly employed in the development and evaluation of spoken dialog systems. However, there is no accepted method for evaluating user simulations, which is problematic because the performance of new dialog management techniques is often evaluated on user simulations alone, not on real people. In this paper, we propose a novel method of evaluating user simulations. We view a user simulation as a predictor of the performance of a dialog system, where per-dialog performance is measured with a domain-specific scoring function. The divergence between the distribution of dialog scores in the real and simulated corpora provides a measure of the quality of the user simulation, and we argue that the Cramér-von Mises divergence is well-suited to this task. To demonstrate this technique, we study a corpus of callers with real information needs and show that Cramér-von Mises divergence conforms to expectations. Finally, we present simple tools which enable practitioners to interpret the statistical significance of comparisons between user simulations.

Key words: user modeling, user simulation, dialog simulation, dialog management

1 Introduction

Traditionally, spoken dialog systems have been hand-built, which is problematic because a human designer needs to consider innumerable dialog situations, many of which can be difficult to foresee. To address this, researchers have begun incorporating machine learning techniques into spoken dialog systems. The idea is for a (human) designer to provide the high-level objectives, and

Email address: jdw@research.att.com (Jason D. Williams).

for the machine learning algorithm to determine what to do in each dialog situation.

Machine learning algorithms for dialogs usually operate by exploring different dialog strategies and making incremental improvements. This process, called *training*, often requires thousands or millions of dialogs to complete, which is clearly infeasible with real users. As a result, machine learning algorithms are usually trained with a *user simulation*, which is a computer program or model that is intended to be a realistic substitute for a population of real users.

Ultimately, the success of a machine learning approach depends on the quality of the user simulation used to train it. Yet, despite this, there is no accepted method to evaluate user simulations. This is especially problematic because machine learning-based dialog systems are often trained *and evaluated* on user simulations alone, not on real users. Without some quantification of user simulation reliability, it is hard to judge claims about machine learning approaches not evaluated on real users.

In this paper, we suggest a quality measure for user simulations. Our quality measure is designed to fill a similar role as a metric like *word error rate* (WER) provides for speech recognition accuracy. WER serves a valuable role by enabling speech recognizers to be rank-ordered, by quantifying improvements in a recognition algorithm, and by providing a measurement of the gap between observed and perfect performance. In the same way, the evaluation metric presented here enables user simulations to be rank-ordered, allows an improvement in a user simulation to be quantified, and provides a measurement of the gap between the observed and perfect user simulation.

Our evaluation method operates as follows. First, since different factors are important in different domains, our method relies on a domain-specific *scoring function*, which assigns a real-valued score to each dialog. Scores from real and simulated dialogs are aggregated to estimate two *distributions*, and the user simulation is evaluated by determining the *similarity* of these distributions using a *normalized Cramér-von Mises* divergence (Anderson, 1962).

The normalized Cramér-von Mises divergence has a host of desirable properties for this task. First, it is designed to handle small *samples* from one or both distributions, which is significant because there may be only 50 or 100 real (human-machine) dialogs available in a given domain. In addition, the Cramér-von Mises divergence makes no assumption about the parametric form of the distributions – such as assuming a normal or uniform distribution – which is important because the parametric form of the score distributions will not be known. Moreover, the Cramér-von Mises divergence accounts for the notion of samples from a “true” distribution and a “modeled” distribution in a principled way. Finally, the normalization enables practitioners to report

user simulation performance on an intuitive, standardized scale.

This paper is organized as follows. First, section 2 reviews background and related work. Next, section 3 states our assumptions, presents the evaluation procedure, and discusses its strengths and limitations. Then, section 4 provides an illustration using real dialog data and confirms that the evaluation procedure agrees with common-sense intuition. Finally, recognizing that there may be a small number of real dialogs available, section 5 tackles the important problem of data sparsity, developing a concise guide for practitioners to easily interpret the reliability of an evaluation. Section 6 then concludes.

2 Background and motivation

A spoken dialog system helps a user to accomplish some goal through spoken language, such as booking an airline reservation, restoring service to an internet connection, or selecting music in an automobile. Figure 1 shows the logical components of a spoken dialog system. A *dialog manager* decides what to say to a user and passes a text string to a *text-to-speech* engine which renders this text string as audio for the user to hear. The *user* speaks in response, and this audio is processed by a *speech recognition* engine which converts the audio into a (possibly erroneous) text string. This text string is passed to a *language understanding* component which attempts to extract the caller's underlying meaning or intent from the text string. This intent is passed back to the dialog manager, which maintains an internal state and updates this state based on the result from the language understanding component. The cycle then repeats until either the goal is achieved or the user (or system) abandons the conversation.

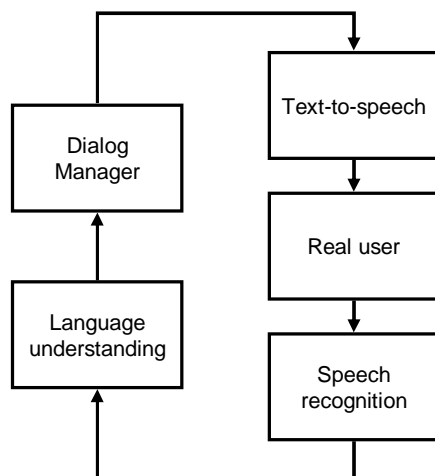


Fig. 1. Logical diagram showing a spoken dialog system. The components are described in the text.

In practice, spoken dialog systems cope with significant uncertainty: speech recognition is error-prone (with error rates for state-of-the-art systems commonly around 30% (Walker et al., 2002; Thomson et al., 2007)), and users' behavior is highly unpredictable. This uncertainty makes it difficult for a dialog designer to anticipate all of the situations a dialog system may encounter, and as a result dialog designers usually resort to sub-optimal heuristics (such as frequent use of confirmation questions) to keep the design process tractable.

Recently, researchers have begun applying machine learning techniques to the problem of dialog design. The essential idea is that the human designer provides high-level objectives, and an optimization or planning algorithm determines the detailed plan. In particular, *Markov decision processes* (MDPs) have been extensively studied, and this research has given rise to sophisticated and novel optimization schemes (Levin and Pieraccini, 1997; Walker et al., 1998; Levin et al., 2000; Goddeau and Pineau, 2000; Singh et al., 2002; Pietquin and Renals, 2002; Scheffler and Young, 2002; Pietquin, 2004; Denecke et al., 2004; Henderson et al., 2005; Frampton and Lemon, 2006; Heeman, 2007). Researchers have also begun to apply *partially observable Markov decision processes* (POMDPs), which extend MDPs by maintaining a distribution over many possible dialog states. This distribution is more robust to speech recognition errors and POMDPs have been shown to yield additional gains over MDPs and hand-built systems, albeit for simpler domains (Roy et al., 2000; Zhang et al., 2001; Williams and Young, 2007b,a; Young et al., 2007; Thomson et al., 2007; Bui et al., 2007; Williams, 2007).

In one way or another, each of these techniques relies on a *user simulation*. A user simulation is a computer program or model that is intended to be a realistic substitute for a population of real users. A user simulation consists of a *user behavior model* which maintains an internal state and generates synthetic user responses, and a *speech recognition model* which simulates the speech recognition process, possibly introducing errors. For example, MDP optimization typically operates by conducting thousands or millions of simulated dialogs, making gradual, periodic improvements (for example, see Scheffler (2002)). In addition, POMDPs also employ a user simulation to infer the likelihood of various user actions (for example, see Williams (2006)). Figure 2 shows a logical diagram, where the dotted box indicates the two elements of the user simulation.

Creating user simulations is a challenging modelling task. Unlike pattern classification, where the goal is to assign the most likely output for a given input, user simulations are generative and attempt to replicate the *variety* of outputs. Moreover, it is not obvious what data structures and parameters are appropriate for modelling human behavior, and there is often very little training data – perhaps 50 or 100 dialogs. Finally, whereas learning algorithms usually assume that the distributions observed in training are identical to distribu-

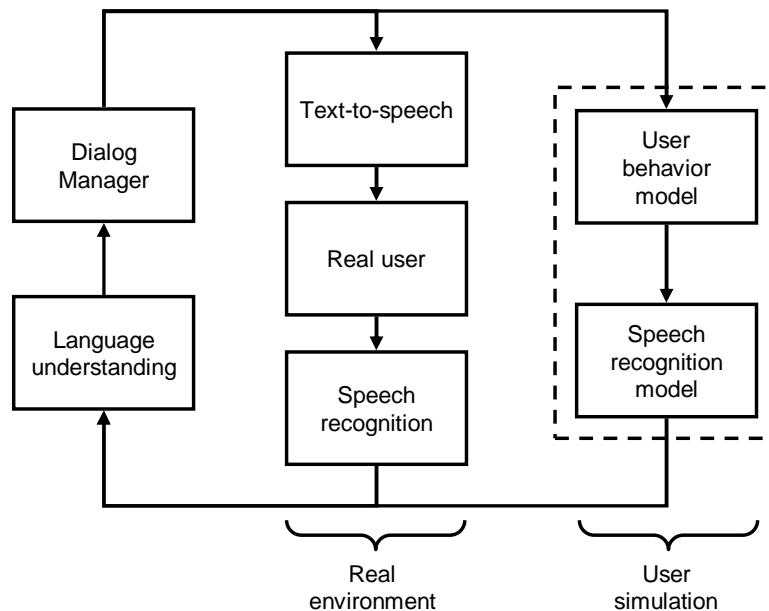


Fig. 2. Logical diagram showing how a user simulation can be used to simulate dialogs. In this paper, “user simulation” refers to both the model of user behavior *and* the model of the speech recognition process.

tions observed at run-time, the aim of a user simulation is to interact with some *new* dialog system, likely to explore many dialog situations unseen in the training data.

A variety of methods have been suggested for implementing user simulations (Schatzmann et al., 2007b). For the user behavior model, the basic idea is to estimate the distribution of user actions given the dialog history. Approaches include expressing the dialog history as an N-Gram (Levin et al., 2000; Georgila et al., 2005) or N-Gram with contextual dialog features (Georgila et al., 2006), casting the user model as a recursive finite-state transition network (Scheffler and Young, 2001), training a maximum entropy model over features describing the dialog history (Georgila et al., 2005), creating a Bayesian network over elements of the dialog history and hidden user state (Pietquin, 2004), and using an agenda or stack-like structure to represent the user’s state (Schatzmann et al., 2007b).

For the speech recognition model, the basic idea is to estimate the distribution of (possibly erroneous) ASR outputs given an input user action. Approaches include expressing the confusion process as a series of finite-state transduction operations (Stuttle et al., 2004), building generative models based on fragment alignment statistics (Schatzmann et al., 2007a), or performing (real) speech recognition on audio composed of concatenated speech segments (Filisko and Seneff, 2005).

Ultimately these user simulation techniques are used to improve or evaluate

dialog systems, and in trials with real users, the resulting dialog systems have outperformed reasonable baselines (Singh et al., 2002; Lemon et al., 2006; Thomson et al., 2007; Young et al., 2007). Unfortunately, evaluations with real users are expensive and time-consuming and as a result are rarely conducted. Instead, it is more common for machine learning applications to be evaluated *exclusively* with a user simulation (Levin and Pieraccini, 1997; Roy et al., 2000; Levin et al., 2000; Goddeau and Pineau, 2000; Zhang et al., 2001; Scheffler and Young, 2002; Pietquin and Renals, 2002; Pietquin, 2004; Henderson et al., 2005; Frampton and Lemon, 2006; Williams and Young, 2007b,a; Heeman, 2007; Bui et al., 2007; Young et al., 2007; Williams, 2007). In all of these studies, no measurement of the accuracy or reliability of the user simulation is reported. As such, it is hard to judge whether performance improvements will hold once systems are deployed to real users.

It is thus appealing to evaluate user simulations *directly*: the hope is that an appropriate measure of goodness would show that a user simulation is a faithful replication of real users, and that its predictions about a dialog system can be trusted. Researchers have suggested a variety of methods. First, Schatzmann et al. (2005) examines whether a user simulation produces responses that a real user would have generated in the same context, and whether a user simulation generates the same variety of responses observed in real dialogs. They propose a broad set of tests for comparing simulated and real dialogs, such as computing the precision and recall of simulated vs. real user responses, and comparing the distribution of turn lengths, dialog lengths, ratio of user to machine actions, and so on. Georgila et al. (2006) add “expected” accuracy, precision, and recall, which seek to measure how well the *variability* of real users has been modeled. Ai and Litman (2007) propose a similar set of measurements for the tutoring domain, adding the learning rate of the simulated user. Taken together, the proposed metrics by these researchers form a useful and powerful toolkit for identifying and investigating differences between user simulations. By applying these tests to user simulation techniques from the literature and several corpora of real dialog data, interesting and important strengths and weaknesses of each user simulation have been identified. However, a toolkit-based approach does not address the problem of a single quality measure for a user simulation. It is not clear how to combine the results of each of the tests in a toolkit to rank order two user simulations, and reporting on all of the tests would be cumbersome for practitioners.

Perplexity has also been suggested as a measure of goodness for a user simulation (Georgila et al., 2005). The perplexity of a model $q(x)$ with respect to a dataset $\{x_i\}$ is defined as $2^{H(q)}$ where $H(q) = -\sum_{i=1}^N \frac{1}{N} \log_2 q(x_i)$. The intuition is that the perplexity expresses how well $q(x)$ models the data $\{x_i\}$: a lower perplexity indicates that $q(x)$ allocates probability mass more closely to the distribution observed in the data. Georgila et al. (2005) propose evaluating user simulations by comparing their perplexity on real, held-out dialog

data. Perplexity is a principled and widely understood measurement in the speech and language community, and it has the desirable property of being a scalar value, which enables user simulations to be rank-ordered. However, while perplexity is useful for making comparisons, perplexity ranges from 1 to infinity, and it is unclear how to interpret the perplexity of a single system. Also, it is unclear how to assign domain-specific attributes, such as the relative importance of dialog length and accuracy. Finally, perplexity requires that the user simulation assigns probabilities to dialogs, but user simulations are usually implemented as computer programs with many layers of sampling, for which computing a well-formed probability may not be possible. In other words, the perplexity metric restricts the types of algorithms that may be used to implement a user simulation.

In other work, Cuayáhuítl et al. (2005) evaluate user simulations by computing the “dialog similarity” of a real and simulated corpus. Each of these two corpora are viewed as the output of a hidden Markov model (HMM), and the dialog similarity measure is defined as the divergence between the distributions comprising these two HMMs. This method has the desirable property of producing a scalar-valued distance which can be used to rank order different user simulations. However, casting corpora as the output of an HMM makes strong assumptions about the structure of dialog, and it is not clear how to determine how well the estimated HMMs match the corpora. In addition, it is unclear how to express the relative importance of different dialog elements, such as task completion and dialog length, in a given domain. Finally, when conveying results, many details of the HMMs such as their states, transition structures, parameterizations, estimation methods, etc. would need to be discussed. Reporting and understanding this level of detail in the course of developing dialog systems would be challenging for practitioners.

More broadly, all techniques for evaluating user simulations face an important, common problem. The ultimate aim of a user simulation is to improve some new dialog system, but currently it is not known what properties of a user simulation are desirable for training new dialog systems. At present, researchers rely on the intuition that a user simulation that is a good replication of user behavior on some *existing* dialog system ought to also be a good replication of user behavior on some *new* dialog system, provided the new dialog system is somehow not too different than the existing system. Whether this holds in practice is an important open research question.

Nonetheless, an important first step is developing a sound method for evaluating user simulations on the same dialog system, and none of the existing methods have all of the desirable properties for an evaluation metric in this domain – i.e., one which is easy to report and interpret, which need only operate in a generative mode, which supports rank-ordering, and which can be tailored to the properties of different domains. In this paper, we seek such a

metric. The next section explains our approach.

3 Method

We start by addressing the overall objective of the user simulation. Although past work has argued that the aim of a user simulation is to engage in “realistic” dialogs (Schatzmann et al., 2005), basing an evaluation measure on realism seems problematic. Indeed, Schatzmann et al. (2005) reports that “it is of course not possible to specify what levels of [evaluation metrics] need to be reached in order to claim that a user simulation is realistic.” Realism is a reasonable aim, but in practice it is unclear how it could be implemented as a quantitative metric.

Here we take a slightly different view. For the purposes of an evaluation metric, we believe that the role of a user simulation is to accurately predict the performance of a dialog system when it is deployed to a certain user population:

Statement 1: For a given dialog system \mathbb{D} and a given user population \mathbb{U}_0 , the goal of a user simulation \mathbb{U}_1 is to accurately predict the performance of \mathbb{D} when it is used by \mathbb{U}_0 .

In other words, we view a user simulation as a *predictive* tool, and our quality measure will assess the accuracy of that prediction. Note that this prediction is dependent on a particular dialog system \mathbb{D} and a particular user population \mathbb{U}_0 . The dialog system is assumed to be in a fixed point, and the user population is defined to include the variations expected across users and the variations expected for each individual user, including variations in experience levels, initiative levels, dialog act frequencies, patience, and so on. For a goal-oriented dialog system, the user population includes the variety and frequency of the tasks that users are trying to accomplish.

Further, it is assumed that the ASR model component of the user simulation \mathbb{U}_1 is non-trivially stochastic and produces semantically meaningful and random errors a realistic fraction of the time, conditioned only on inputs available at run-time to a real speech recognizer. In other words, like a real speech recognizer, at a given point in a dialog, the same semantic input to the ASR model will generate meaningfully different semantic outputs. Violating this assumption can produce undesirable results, and this is discussed in detail in Appendix 1.

The quantity being predicted is “performance” of a dialog system, which we must of course define concretely. We first address performance in a single

dialog:

Statement 2: The performance of a dialog system \mathbb{D} in a particular dialog $d_{(i)}$ can be expressed as a single real-valued score $x_{(i)}$, computed by a scoring function $\mathbb{Q}(d_{(i)}) = x_{(i)}$.

The scoring function itself is dependent on the dialog system and is created by its designer. The scoring function captures all of the factors that the designer believes are relevant – such as task completion, dialog length, and user satisfaction – and combines them in some way. The designer may base the scoring function on business requirements (such as those suggested by Levin and Pieraccini (2006)) or a weighted sum of factors intended to predict user satisfaction such as the PARADISE method (Walker et al., 2000). In any case, the main aim of the dialog design process is to make trade-offs between competing needs (for example, speed and accuracy) appropriately, and the scoring function codifies this in mathematical terms. Often, the scoring function is already available since it is required by many machine-learning algorithms, such as Markov decision processes and partially observable Markov decision processes, where it is called a *reward function* (See, for example, Levin et al. (2000) and Williams and Young (2007b)). In this paper, we will not define the scoring function explicitly since this scoring function will be very different in different domains: for example, a scoring function in the entertainment domain will likely be very different than a scoring function in the banking domain.

Next, these scores can be aggregated into sets:

Statement 3: A given user population \mathbb{U}_0 using dialog system \mathbb{D} will yield a set of scores $\mathcal{S}_0 = (x_{(1)}^0, \dots, x_{(N_0)}^0)$. Similarly, a user simulation \mathbb{U}_1 using dialog system \mathbb{D} will yield a set of scores $\mathcal{S}_1 = (x_{(1)}^1, \dots, x_{(N_1)}^1)$.

With these two sets, we can now state the basic intuition of our quality measure for a user simulation:

Statement 4: A user simulation \mathbb{U}_1 may be evaluated by computing a real-valued divergence $D(\mathcal{S}_0||\mathcal{S}_1)$.

In this paper we define a *divergence* $D(\mathcal{X}||\mathcal{Y})$ to be a non-negative, real-valued measurement of how well some set \mathcal{X} , which is taken to be samples from a “true” distribution, is matched by some other set \mathcal{Y} , taken to be a “model” of the truth. Here, a divergence expresses how well the scores produced by the user simulation \mathcal{S}_1 match the scores produced with real users \mathcal{S}_0 . Similar to a distance measurement, $D(\mathcal{X}||\mathcal{X}) = 0$; however, unlike a distance measurement, a divergence is not necessarily symmetric: $D(\mathcal{X}||\mathcal{Y})$ is not necessarily

equal to $D(\mathcal{Y}||\mathcal{X})$.

Because a divergence is a scalar (a real number), divergences to different user simulations can be rank-ordered, enabling direct comparisons to be made between different user simulations.

In the limit of an infinite number of dialogs, the sets \mathcal{S}_0 and \mathcal{S}_1 could be described by probability density functions $p_0(x)$ and $p_1(x)$. In practice, however, collecting real dialogs is expensive and time-consuming, and there may only be $N_0 = 50$ or 100 real dialogs. As a result, an estimate of the probability density function $p_0(x)$ is likely to be unreliable. Moreover, it seems unlikely that we will know the parametric form of $p_0(x)$ in advance. Thus, our divergence measurement should not make any assumption about parametric form.

Given these considerations, a natural choice of divergence measure is the *normalized Cramér-von Mises* divergence:

$$D(F_0||F_1) = \alpha \sqrt{\sum_{i=1}^{N_0} (F_0(x_{(i)}^0) - F_1(x_{(i)}^0))^2} \quad (1)$$

where F_j is the *empirical distribution function* (EDF) of the data $\mathcal{S}_j = (x_{(1)}^j, \dots, x_{(N_j)}^j)$:

$$F_j(x) = \frac{1}{N_j} \sum_{i=1}^{N_j} \begin{cases} 1 & \text{if } x_{(i)}^j < x \\ \frac{1}{2} & \text{if } x_{(i)}^j = x \\ 0 & \text{if } x_{(i)}^j > x \end{cases} \quad (2)$$

and α is a normalizing constant

$$\alpha = \sqrt{\frac{12N_0}{4N_0^2 - 1}} \quad (3)$$

which scales the upper bound of $D(F_0||F_1)$ to be 1. Appendix 2 provides a derivation of α .

The normalized Cramér-von Mises divergence is based on a family of statistical tests originally developed by Cramér (1928) and von Mises (1931) which measure agreement between observed sets of data. Equation 1 is based on a variant of the Cramér-von Mises criterion studied by Anderson (1962), augmented here with a normalization constant α .

Intuitively, an EDF $F(x)$ gives an estimate of the percent of the observed data which is less than x , and the Cramér-von Mises divergence is a sum of the squared differences between the EDFs of the real data \mathcal{S}_0 and modeled data \mathcal{S}_1 evaluated at the real data points \mathcal{S}_0 .

The normalized Cramér-von Mises divergence has the properties desired for evaluating user simulations. Because it operates on the empirical distribution function (EDF), it makes no assumptions about the parametric form of $p(x)$ and requires no tuning parameters. In addition, the Cramér-von Mises family of tests is regarded as having more statistical power than other non-parametric methods for comparing EDFs (Stephens, 1992), such as the Kolmogorov-Smirnov test (Kolmogorov, 1933). Moreover, if $F_0(x)$ and $F_1(x)$ are co-incident at the points in \mathcal{S}_0 , then $D(F_0||F_1) = 0$. In other words, if \mathcal{S}_0 and \mathcal{S}_1 have the same distribution of scores, then the user simulation \mathbb{U}_1 will be taken to be a perfect model of the real user \mathbb{U}_0 . Finally, the *normalized* Cramér-von Mises divergence provides a common scale of $[0, 1]$, independent of N_0 , which enables scores to be easily interpreted.

In the speech and language community, a common divergence measurement is the Kullback-Leibler (KL) divergence $D_{\text{KL}}(f_0||f_1) = \int_{-\infty}^{\infty} f_0(x) \log(f_0(x)/f_1(x))dx$ (Kullback and Leibler, 1951). The KL divergence is less desirable for this task for three main reasons. First, it requires estimating densities f_0 and f_1 from the data \mathcal{S}_0 and \mathcal{S}_1 , which in turn requires making assumptions that can introduce error. Second, interpreting a KL divergence is troublesome: KL divergence is (positively) unbounded which precludes measuring where on the spectrum of best to worst a user simulation lies. By contrast, the normalized Cramér-von Mises divergence ranges from 0 to 1, where 0 implies the best possible user simulation and 1 implies a completely mis-estimated user simulation. Finally, the KL divergence places a disproportionately high penalty on *under-estimating* variability. This is illustrated in Figure 3, which shows the KL and Cramér-von Mises divergences from a Gaussian with mean 0 and standard deviation 1 to other Gaussians with various means and variances.¹ Under-estimation of the standard deviation leads to explosive growth in the KL divergence, whereas the Cramér-von Mises divergence makes a more balanced trade-off between errors in the mean and standard deviation. In this task, it doesn't seem appropriate to encode a preference for over-estimating variances.

To summarize the Cramér-von Mises method, each real dialog and each simulated dialog is assigned a score by a scoring function created by the system designer. The aim of a user simulation is to interact with a dialog system and predict the distribution of scores that will be observed when the same dialog system is deployed to a population of real users. Since in practice there may be few real dialogs, and since the form of the score distributions is unlikely to be known, we choose the normalized Cramér-von Mises divergence which has properties well-suited to this application. Our statement of the Cramér-von Mises divergence includes a normalization constant and yields results on a

¹ These plots were generated using densities rather than samples, which is discussed below in section 5.

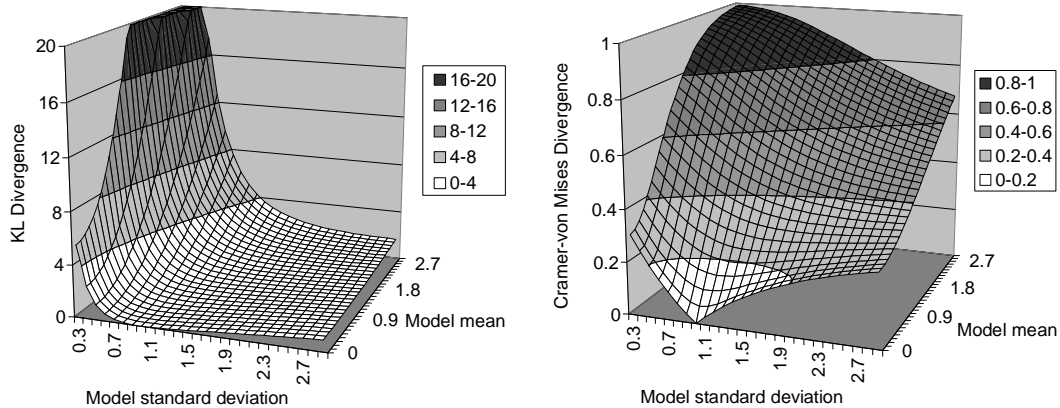


Fig. 3. Kullback-Leibler divergence $D_{KL}(f_0||f_1)$ (left) and the Cramér-von Mises divergence $D(F_0||F_1)$ (right), where f_0 is the density and F_0 is the cumulative distribution functions of a Gaussian with mean 0 and standard deviation 1. In these plots, the horizontal axes of the plots indicate the mean and standard deviation of F_1 and f_1 , and the height of the surface indicates the divergence.

standardized, intuitive scale.

Comparing two user simulations \mathcal{S}_1 and \mathcal{S}_2 simply requires computing $D(F_0||F_1)$ and $D(F_0||F_2)$, and the minimum of these divergences is the better user simulation. A logical diagram of this process is shown in Figure 4.

This metric enables user simulations to be compared to each other by quantifying the divergence between a user simulation and a population of real users.² In comparison to the HMM-based method of Cuayáhuatl et al. (2005) and perplexity-based measure of Georgila et al. (2005), this method has several advantages: the Cramér-von Mises method makes no assumptions about the structure of dialogs; its scoring function enables dialog designers to explicitly specify the relative importance of dialog elements such as speed and accuracy in a given domain; it does not require assignment of probabilities to dialogs; it operates directly on the data, making no parametric assumptions; and the scoring function and measurement are easy to report and interpret.

Even so, this method has several limitations. First, just as evaluation metrics like WER do not suggest how a speech recognizer could be improved, we do not expect that our metric will suggest how a user simulation could be improved. The toolkits of tests proposed by Schatzmann et al. (2005), Georgila et al. (2006), Ai and Litman (2007), and others seem more appropriate for this type of analysis. Also, even if the dialog scores in the sets \mathcal{S}_0 and \mathcal{S}_1 are co-incident, the true and modelled dialogs may still be quite different, for example in terms of sequences of dialog acts. This is inevitable with any scalar evaluation metric: for example, in speech recognition, identical word error rates may have

² The method might also be useful for comparing different populations of real users, but this is not explored in this paper.

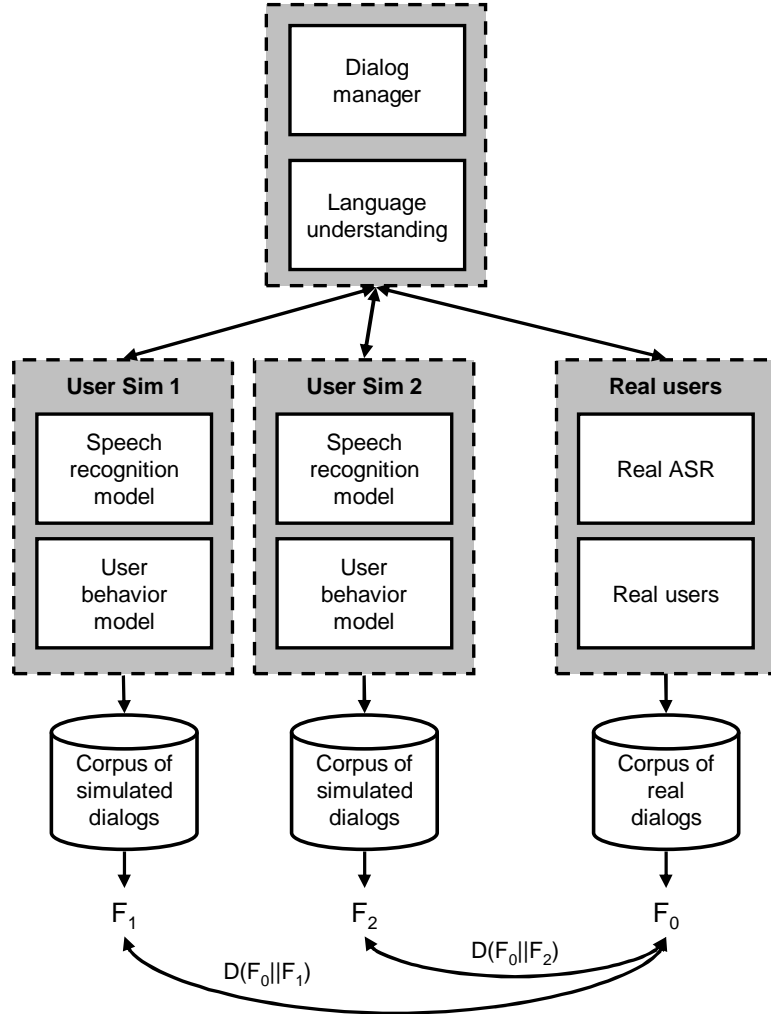


Fig. 4. Logical diagram summarizing the method for evaluating user simulations. A domain-specific scoring function \mathbb{Q} scores each of the real human-computer and simulated human-computer dialogs to form sets \mathcal{S}_0 , \mathcal{S}_1 , and \mathcal{S}_2 , which are in turn used to form the empirical distribution functions F_0 , F_1 , and F_2 , respectively.

different measurements of deletions, insertions, and substitution errors. Here, the intention is that the domain-specific scoring function weights the relevant factors of the dialog appropriately, such that any aliasing is by definition acceptable.

Finally, it is important to recall that the larger question of what sort of user simulation is best for training some *new* dialog system remains open. As a result, currently no evaluation measure can safely claim to support sound comparisons of user simulation techniques across domains. For example, a simple user simulation built for a simple dialog system might produce a low divergence, but that does not imply that the same user simulation technique will be a good predictor of the performance of a more sophisticated dialog system or more complex domain. A broader program of research is needed to

understand how these predictions can reliably be made, if at all.

Nevertheless, an important first step is understanding how to compare user simulations on a single dialog system. In the next section, a series of experiments with a real dialog system are presented, which strive to confirm that this evaluation procedure is consistent with common-sense intuitions and can be easily applied.

4 Example application to a real dialog system

In this section, we strive to show that the normalized Cramér-von Mises evaluation procedure agrees with common-sense intuition by studying a corpus of dialogs with a real dialog system. A series of user simulations are created, and it is shown that increasingly realistic user simulations yield decreasing Cramér-von Mises divergences. In other words, it is shown that the Cramér-von Mises divergence correlates well with the qualitative difference between the real environment and the user simulation.

4.1 Application and corpus

The dialog system presented here is a voice dialer application. This application is advertised and accessible within the AT&T research laboratory and receives daily calls from different members of the laboratory. In the simplest case, callers say the name of the person they want to call (the callee) and the system responds with an implicit confirmation and transfers the call. If a callee has an office and mobile (cell) phone registered, the system will ask which one to call. The dialer’s vocabulary consists of approximately 30,000 distinct callees across many business units (not just staff in the research laboratory) and as a result, a recognized name (such as “John Smith”) often matches multiple callees. In this case, the system can ask for a city and state.³ Overall the prompt language is rather terse, which is designed to encourage fast, repeat usage. Table 1 provides the transcript of an example real dialog with the voice dialer.

The dialer has been active for over a year, and in that time has received more than 468 calls from 40 distinct callers. These figures exclude calls from the individuals involved with the development of the system.

³ The dialog system can also read out phone numbers on request, but in practice this functionality is used very rarely.

Speaker	Transcript	Score
System	AT&T Dialer.	-1
Caller	[silent]	
System	First name and last name?	-1
Caller	Jeannie Smith [JENNY SCHMIDT U S ~ 85]	
System	Jenny Schmidt, USA	-1
Caller	No [NO ~ 95]	
System	First name and last name?	-1
Caller	Jeannie Smith [JEANNE SMITH ~ 98]	
System	Jeanne Smith. Office or cell?	-1
Caller	Office [OFFICE ~ 82]	
System	Office phone. Office or cell?	-1
Caller	[silent]	
System	Dialing.	+20
<i>Total score</i>		+14

Table 1. An example conversation with the AT&T dialer. Text in brackets shows ASR recognition result and confidence score, which ranges from 0 to 100. The scoring function is described in the text below and summarized in Table 2. Names have been changed for privacy.

We selected the dialer application for this study because it is used by people with real needs. Since our focus is on user behavior, it would be less desirable to use dialogs collected from paid subjects, who in effect role-play and do not really suffer the consequences of system failures.

To illustrate the Cramér-von Mises evaluation method, we began by defining a scoring function. The scoring function provides a large positive reward for transferring the caller to the correct callee, a large negative reward for transferring the caller to the incorrect callee, and smaller negative rewards if the caller hangs up without making a connection.⁴ In addition a per-turn penalty rewards faster dialogs. Table 2 explains the entire scoring function, which has

⁴ The small handful of dialogs in which the caller requested to hear the callee’s phone number (instead of being transferred) were scored in a similar manner.

Condition	Score
System transfers caller to the correct destination	20
System transfers caller to the incorrect destination	-20
System hangs up for any reason	-20
Caller hangs up at very first turn	0
Caller hangs up after very first turn	-5
Each system turn	-1

Table 2. The scoring function used for the voice dialer.

x	$ x \in \mathcal{S}^* $	$ x \in \mathcal{S} $	x	$ x \in \mathcal{S}^* $	$ x \in \mathcal{S} $	x	$ x \in \mathcal{S}^* $	$ x \in \mathcal{S} $
-32	0	1	-15	0	1	9	1	0
-31	2	0	-12	1	0	10	0	1
-29	1	1	-11	2	4	11	0	1
-27	3	0	-10	2	9	12	0	6
-26	3	2	-9	5	17	13	3	8
-25	4	7	-8	7	16	14	1	13
-24	4	5	-7	9	25	15	11	26
-23	5	9	-1	0	1	16	27	77
						17	58	90

Table 3. Frequencies of scores in the training and test sets. x denotes a score, \mathcal{S}^* denotes the test set, and \mathcal{S} denotes the training set.

also been applied to the dialog in Table 1.

We then divided the 468 logged dialogs into a training set (of 320 calls with 1265 caller turns) and a test set (of 148 calls with 581 caller turns), with disjoint sets of callers. In other words, all of the calls from a given caller were grouped into either the training set or the test set. Below, the training set will be used to develop a series of user simulations, and the test set will be used for comparison and assessment.

Table 3 shows the counts of each score for the training set \mathcal{S} and test set \mathcal{S}^* . There are effectively 3 clusters of scores: a cluster in the range -32 to -23 which accounts for incorrect transfers, a cluster in the range of -12 to -7 which accounts for user hang-ups, and a cluster in the range of 9 to 17 which accounts for correct transfers. The variation within each of these clusters is due to the variation in dialog length.

Figure 5 plots this data as an empirical distribution function (EDF). This plot

illustrates some of the basic properties of an EDF: for $x < \min(\mathcal{S})$, $F(x) = 0$, for $x > \max(\mathcal{S})$, $F(x) = 1$, and for $\min(\mathcal{S}) \leq x \leq \max(\mathcal{S})$, $F(x)$ increases monotonically, with an increment at each $x_{(i)} \in \mathcal{S}$. Regions with large increments indicate a higher density of data, and flat regions indicate an absence of data.

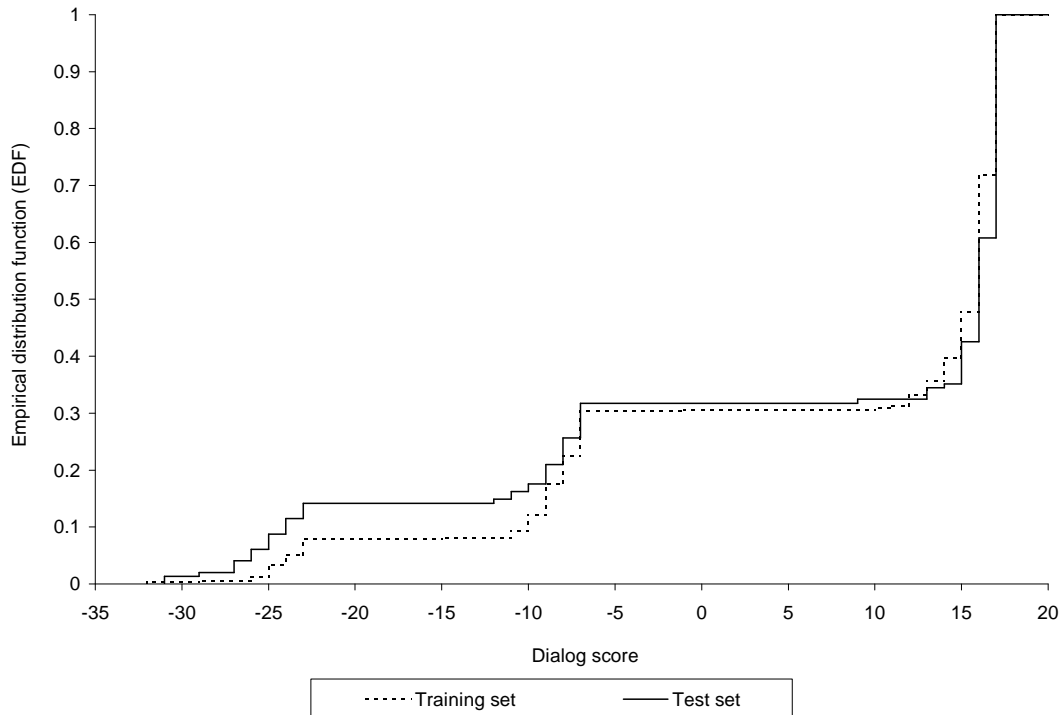


Fig. 5. Empirical distribution function of test set $F^*(x)$ and training set $F(x)$.

4.2 User simulations

Next, four user simulations were built. By design, the user simulations vary in how closely they mimic the behavior of a real user and real ASR. Our hypothesis is that user simulations which deviate significantly from reality will show a larger Cramér-von Mises divergence, and user simulations which more closely model real users will show a smaller Cramér-von Mises divergence.

First, two different user behavior models were created. A *handcrafted* user behavior model was designed which assumed that the user is cooperative and patient, always answering questions as requested, and never hanging up. Second, a *stochastic* user behavior model was estimated from the training dialog data. At each system prompt, categories of user responses were counted, including cooperative answers, out-of-grammar speech, silence, and hang-up. For cooperative answers, combinations of slots were counted – for example, if a user was observed to say “Jason Williams, Florham Park” in the training set of dialogs, this includes the *first-name*, *last-name*, and *city* slots. These

Situation (s)	User action (a)	$P(a s)$
System asks for name	<i>first-name,last-name</i>	0.692
	<i>first-name,last-name,city,state</i>	0.033
	out of grammar	0.147
	silent	0.039
	hang-up	0.089
System confirms correct name	<i>yes</i>	0.089
	<i>no</i>	0.097
	out of grammar	0.040
	silent	0.766
	hang-up	0.008
System confirms incorrect name	<i>yes</i>	0.047
	<i>no</i>	0.422
	out of grammar	0.063
	silent	0.313
	hang-up	0.156

Table 4. Portion of the stochastic user behavior model.

frequency counts were used to form a statistical model of user behavior using simple maximum-likelihood estimation. A portion of the resulting statistical model of user behavior is shown in Table 4.

Next, two speech recognition simulations were created. Each speech recognition simulation takes as input the text of the user’s speech, and produces as output a (possibly erroneous) text string and a confidence score, which indicates the reliability of the output text string. The confidence score is used by the dialog manager to decide whether to accept or discard the output.

The first speech recognition simulation made no errors: in-grammar speech was recognized accurately (with the maximum confidence score of 100), silence was correctly identified, and out-of-grammar speech was discarded (via a confidence score of zero). The second speech recognition simulation modelled the errors and confidence scores found in the training set. Error statistics were computed by examining each recognition attempt in the training set, and determining whether the user’s speech a was in-grammar, out-of-grammar, or empty, and also determining whether the recognition outcome \tilde{a} was correct, incorrect, or empty. Counts of each (a, \tilde{a}) pair were made and used to compute conditional probabilities $P(\tilde{a}|a)$, shown in Table 5.

User action type (a)	Recognition outcome (\tilde{a})	$P(\tilde{a} a)$
In-grammar speech	Recognized correctly	0.795
	Recognized incorrectly as other speech	0.190
	Recognized incorrectly as silence	0.015
Out-of-grammar speech	Recognized incorrectly as other speech	0.921
	Recognized incorrectly as silence	0.079
Silence	Recognized incorrectly as other speech	0.075
	Recognized correctly as silence	0.925

Table 5. ASR simulation’s model of recognition errors for the main name grammar.

For each (a, \tilde{a}) pair, confidence score frequencies in the training set were counted and used to construct empirical distribution functions (EDFs). Figure 6 shows two of these EDFs for in-grammar speech. As expected, these plots show that the confidence score for incorrect recognitions is more likely to be low (80% are less than 30), whereas the confidence score for correct recognitions is more likely to be high (80% are greater than 60).

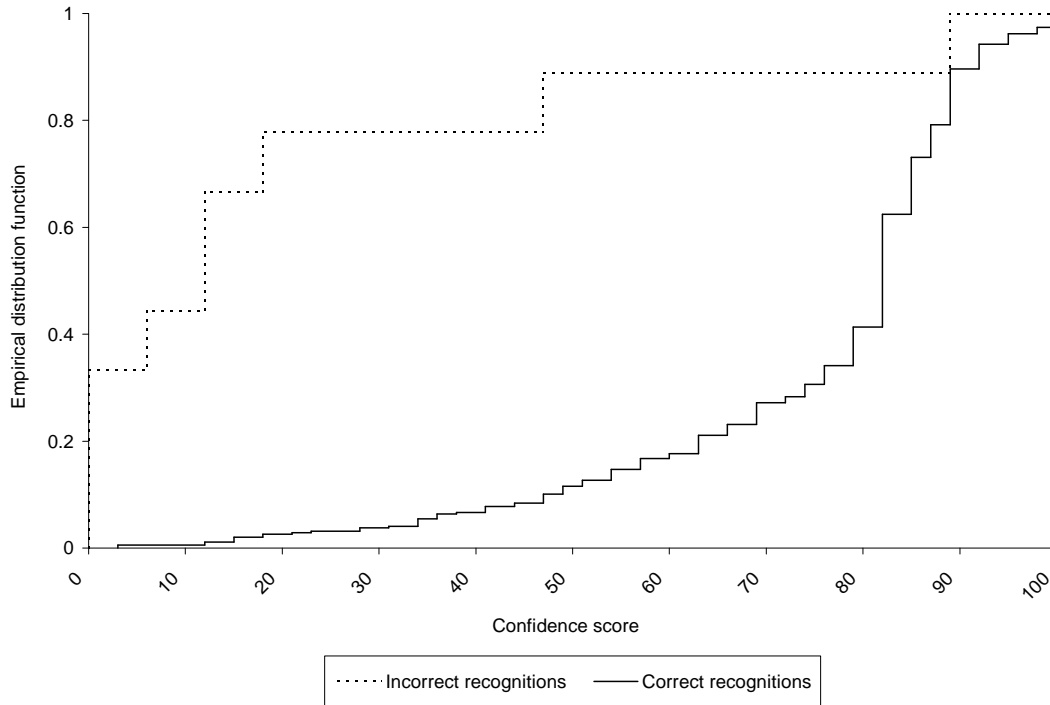


Fig. 6. Empirical distribution function of confidence scores for correct recognitions and incorrect recognitions given that the user’s speech was in-grammar. For correct recognitions, the confidence score is likely to be higher than for incorrect recognitions.

To simulate the speech recognition process at runtime, it is first determined whether the user behavior model’s speech a is in-grammar, out-of-grammar,

Corpus of model dialogs used to compute EDF \hat{F}	$D(F \hat{F})$
Handcrafted user behavior + perfect ASR	0.36
Modeled user behavior + perfect ASR	0.21
Handcrafted user behavior + modeled ASR	0.20
Modeled user behavior + modeled ASR	0.067
Training set (real dialogs)	0.098

Table 6. Cramér-von Mises divergence between the EDF of the test set of real dialogs (F) and other corpora.

or empty. Given the type of user speech, a recognition outcome \tilde{a} is then sampled based on the statistics in Table 5. If the sampled recognition outcome is “Recognized incorrectly as other speech”, then a new utterance is sampled from the grammar at random. Finally, given both the type of user speech and the recognition outcome, a confidence score is sampled from the appropriate EDF, such as the two shown in Figure 6. The recognition result and confidence score are then passed to the dialog system. For each grammar, the dialog manager has a confidence rejection threshold. If the confidence score is greater than this threshold, then it is accepted; otherwise, it is treated as a likely error and discarded.

As an example of this process, suppose the user behavior model generates the output “Jason Williams”. The ASR simulation first checks whether this speech (a) is in the grammar; finding that it is in-grammar, it would sample an outcome (\tilde{a}) from the first three rows in Table 5. Suppose it sampled the outcome “Recognized incorrectly as other speech” (which occurs with $p = 0.190$). In this case, some other output would be sampled from the grammar, such as “Jay Wilpon”, and this would be output as the recognition result instead of “Jason Williams”. Since this is an incorrect recognition, a confidence score is then sampled according to the empirical distribution of the dotted line in Figure 6.

4.3 Experiments and results

Each of the two user behavior models was run with each of the ASR simulations for 1000 dialogs, and each dialog was scored using the scoring function, described in Table 2. The EDF for each of the user behavior model/ASR model pairs was then computed and plotted. Results are shown in Figures 7 and 8.

Finally, using the EDFs computed for each user simulation, the normalized Cramér-von Mises divergences from the test set were computed. Results are shown in Table 6.

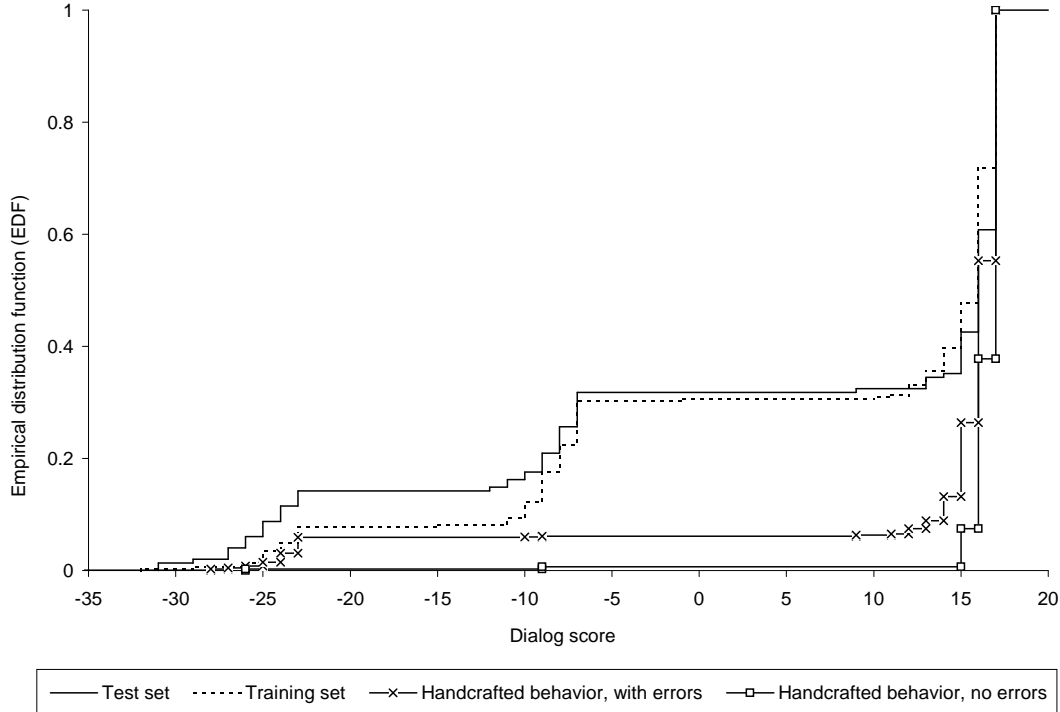


Fig. 7. Empirical distribution function of the training set, test set, and the handcrafted user simulations.

The handcrafted user behavior model with no ASR errors produces the largest Cramér-von Mises divergence; the stochastic user behavior with ASR errors produces the smallest divergence; and the other combinations are between these two. In other words, as the predictive accuracy of the user simulation increases, its Cramér-von Mises divergence decreases, as expected. In this experiment, the best and worst user simulations were known in advance by design: the key result is that the Cramér-von Mises divergence has recovered this ordering, and this finding lends support to our claim that the normalized Cramér-von Mises divergence is a suitable metric for evaluating user simulations.

In addition, the divergence from the held-out test set to the training set is nearly identical to the best user simulation. Of course, formally it can never be proved that two sets of samples are drawn from the same distribution. Even so, for this scoring function and this number of real dialogs N_0 , the predictive accuracy of the best user simulation is within the bounds of sampling error measured with held-out data, implying some reassurance in its quality. Yet this also raises an important question: the empirical distribution functions are subject to sampling noise: indeed, the divergence from the test set to the training set – two sets of *real* data – is not zero but rather 0.098. Might the other differences observed be accounted for by sampling error? More generally, what magnitude of difference in divergence is statistically significant? This is the question addressed in the next section.

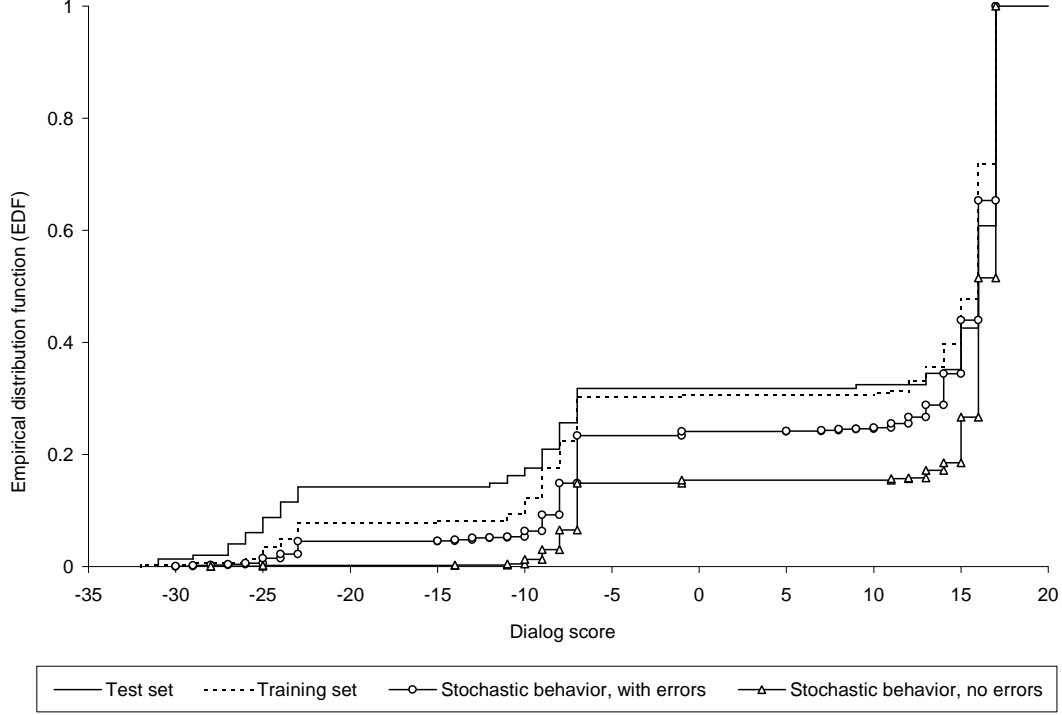


Fig. 8. Empirical distribution function of the training set, test set, and the stochastic user simulations.

5 Statistical significance of the Cramér-von Mises divergence

In the previous section, several user simulations were created and rank-ordered using the Cramér-von Mises divergence. While the rank-ordering agreed with common-sense expectations, it is important to confirm that the differences measured were statistically significant. More generally, given that the number of real dialogs is often limited, we seek to provide guidance to system developers and practitioners on the reliability of a rank ordering of user simulations calculated with the Cramér-von Mises divergence test.

To begin, consider the cumulative distribution functions (CDF) $P_i(x)$ and probability density functions (PDF) $p_i(x)$ for the user population and two user simulations. By definition, these describe the true cumulative distribution and densities of the user population and the two user simulations in the presence of an infinite number of samples. The normalized Cramér-von Mises divergence on the true distributions, which is also called the normalized *Cramer-Smirnov-Von-Mises* test (Eadie et al., 1971), can be computed as:

$$D(P_0||P_j) = \beta \sqrt{\int (P_0(x) - P_j(x))^2 p_0(x) dx} \quad (4)$$

where β is a normalization constant

$$\beta = \sqrt{3}. \quad (5)$$

For a derivation of β , see Appendix 3.

If this test is applied to each user simulation and it is found that $D(P_0||P_1) < D(P_0||P_2)$, then it could be concluded that user simulation 1 is better than user simulation 2 (and visa-versa). Since these quantities are exact, there is no chance that an observed difference would be due to noise: any difference is statistically significant.

In practice however, we will not have access to $P_i(x)$ nor $p_i(x)$. Rather, we have samples from these distributions $\mathcal{S}_j = (x_{(1)}^j, \dots, x_{(N_j)}^j)$ which are used to compute $D(F_0||F_j)$. The key issue is that $D(F_0||F_j)$ is an *estimate* of $D(P_0||P_j)$ and therefore subject to sampling error.

We tackle this problem by constructing a simulation experiment. We first assume that, in principle, the sampled dialog scores are drawn from distributions which can be represented exactly (i.e., parametrically). We randomly generate these exact distributions for a user population $P_0(x)$ and two user simulations $P_1(x)$ and $P_2(x)$. Then, we compute the *true* ordering of the two user simulations as the ordering of $D(P_0||P_1)$ and $D(P_0||P_2)$. Next, we sample from $P_0(x)$, $P_1(x)$, and $P_2(x)$ to produce $F_0(x)$, $F_1(x)$ and $F_2(x)$, and compute *predicted* ordering of the two user simulations as the ordering of $D(F_0||F_1)$ and $D(F_0||F_2)$. Finally we determine if the predicted ordering agrees with the true ordering, and set an indicator variable q :

$$q = \begin{cases} 1, & \text{if predicted rank ordering matches true rank ordering} \\ 0, & \text{if predicted rank ordering does not match true rank ordering.} \end{cases} \quad (6)$$

This whole process is repeated M times, and for each iteration m , $D(F_0||F_1)$, $D(F_0||F_2)$, and q are stored as D_1^m , D_2^m , and q^m , respectively. Once the sampling is complete, a plot is constructed which quantizes D_1 and D_2 into square regions. Within each region, the average value of q notated \bar{q} is computed, which corresponds to the percentage of the time that the sampled data yields the same model ordering as the true data. In other words, the end result is a statement of the accuracy of the ordering of 2 user simulations for a given D_1 , D_2 , N_0 , N_1 and N_2 .

Concretely, $p_j(x)$ are multi-modal densities represented as the weighted sum of Gaussians, with densities

$$p_j(x) = \sum_{k=1}^{K_j} w_{j,k} N(\mu_{j,k}, \sigma_{j,k}) \quad (7)$$

where weights $w_{j,k}$ are sampled uniformly and scaled such that

$$\begin{aligned} 0 \leq w_{j,k} \leq 1, \forall j, k \\ \sum_k w_{j,k} = 1, \forall j \end{aligned} \tag{8}$$

and means (μ), and variances (σ) are sampled uniformly from the range:

$$\begin{aligned} \mu_{j,k} \sim [0, 100], \forall j, k \\ \sigma_{j,k} \sim [1, 5], \forall j, k. \end{aligned} \tag{9}$$

In these experiments, the number of dialogs from each user simulation is $N_1 = N_2 = 1000$, and $M = 40,000$ iterations are run for each experiment.⁵ The number of modes K_j was set to 2.⁶ Experiments are run for various number of dialogs from the “real” user N_0 ranging from 50 to 1000.

Figures 9 and 10 show one iteration of this process. Figure 9 shows the probability density functions $p_j(x)$. Figure 10 show these densities as cumulative distribution functions $P_j(x)$, along with the empirical density functions $F_j(x)$ for $N_1 = N_2 = 1000$ dialogs with each user simulation and $N_0 = 50$ dialogs with the “real” user. In this iteration, the true divergences are $D(P_0||P_1) = 0.319$ and $D(P_0||P_2) = 0.406$, and the sampled divergences are $D(F_0||F_1) = 0.317$ and $D(F_0||F_2) = 0.465$. Since the sampled diversions predict the same ordering as the true divergences – i.e., $D(P_0||P_1) < D(P_0||P_2)$ and $D(F_0||F_1) < D(F_0||F_2)$ – then $q = 1$ and the tuple $(D_1, D_2, q) = (0.317, 0.465, 1)$ is stored.

Figure 11 shows the results of the simulations. In this figure, black regions indicate $\bar{q} < 0.95$, white regions indicate $\bar{q} > 0.99$, and various shades of gray indicate intermediate values.

Figure 11 provides an empirical measurement of the reliability of a rank ordering given the divergences D_1 and D_2 . For example, suppose that two user simulations (run for 1000 dialogs each) are compared to a corpus of 50 real dialogs, and the two divergences measured are $D_1 = 0.1$ and $D_2 = 0.3$. This would imply that user simulation 1 is more accurate than user simulation 2, and the $N_0 = 50$ panel of Figure 11 indicates that this ordering is reliable with $p > 0.98$. Had the two divergences been $D_1 = 0.2$ and $D_2 = 0.3$, the ordering would be reliable with $p < 0.95$. On the other hand, had the corpus of real dialogs consisted of $N = 200$ dialogs, then the ordering implied by the two divergences $D_1 = 0.2$ and $D_2 = 0.3$ would be more reliable, with $p > 0.97$.

⁵ Confidence intervals appeared to reach an asymptote by 1000 simulated dialogs, so generating more than 1000 simulated dialogs (N_1 and N_2) is of marginal benefit.

⁶ Additional experimentation not described here showed that the findings below were unchanged for larger numbers of modes.

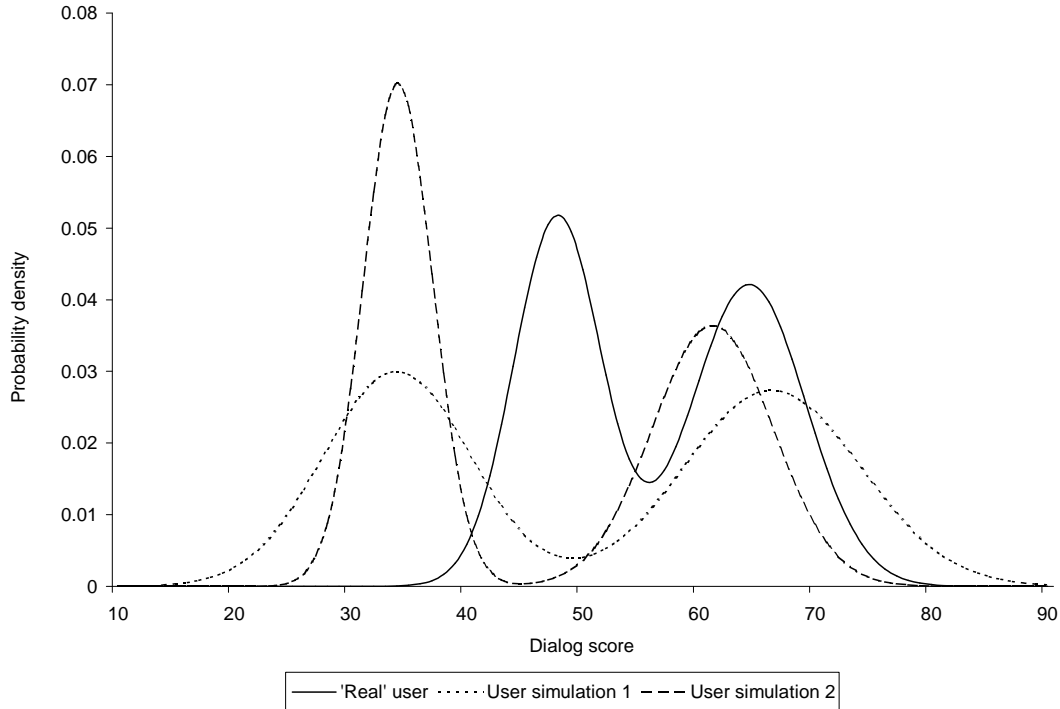


Fig. 9. Example probability density functions for the “real” user $p_0(x)$ and the two user simulations $p_1(x)$ and $p_2(x)$.

These figures also show that the regions of lower probability lie along essentially straight lines parallel to $D_1 = D_2$. This is significant because it implies that the reliability of an ordering is determined mainly by the *difference* between D_1 and D_2 , rather than being dependent on their actual values. This relationship is shown in Figure 12.

Table 7 summarizes our results. This table provides an indication of what differences in divergences are required to conclude an ordering of two user simulations is reliable with confidence 90% and 95%. The normalized Cramér-von Mises divergence, together with this table of critical values, provides a method for practitioners to rank order user simulations, and to determine whether that rank ordering is significant.

At the end of section 4, the question was raised whether the differences observed between the different user simulations in the dialer application were statistically significant. In those experiments, there were 148 dialogs in the testing set. The results in Table 7 indicate that, for 100 real dialogs, a difference of 0.06 indicates a 90% ordering accuracy, and a difference of 0.09 indicates a 95% ordering accuracy. This implies that the handcrafted user behavior with no ASR errors is significantly worse than the other user simulations, and that the stochastic user behavior with ASR errors is significantly better than the other user simulations. Further, the difference observed between the stochastic user behavior model with ASR errors (0.067) and the

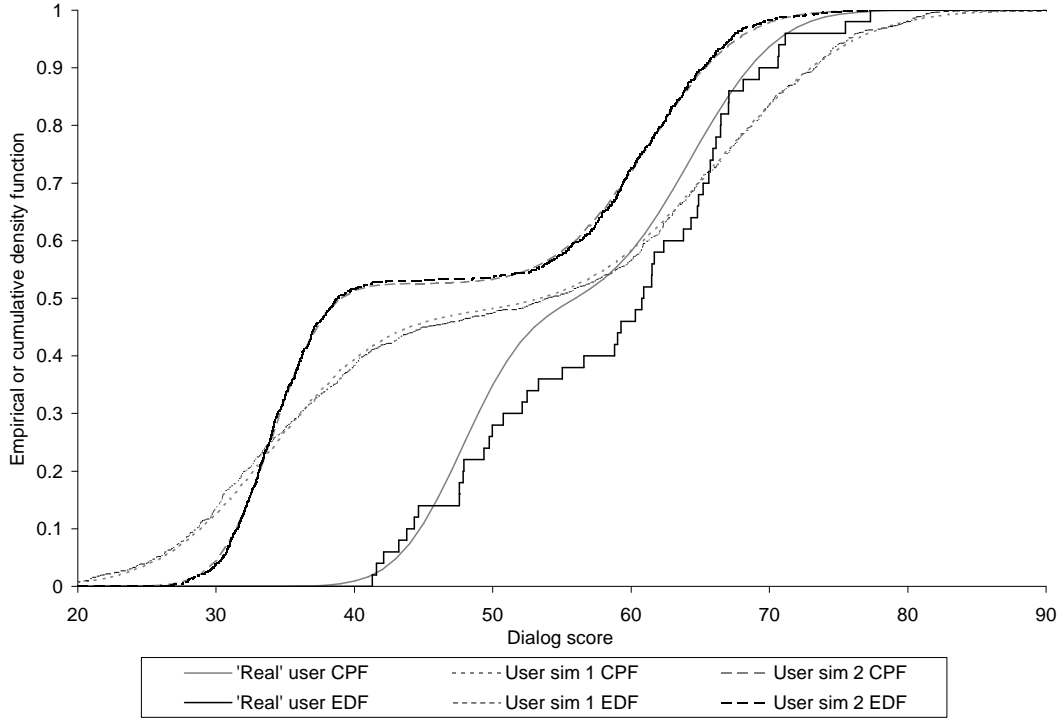


Fig. 10. Example cumulative probability function for the “real” user $P_0(x)$ and the two user simulations $P_1(x)$ and $P_2(x)$, and the empirical distribution functions for the “real” user $F_0(x)$ and the two user simulations $F_1(x)$ and $F_2(x)$. In this example, $N_0 = 50$ dialogs from the “real” user and $N_1 = N_2 = 1000$ dialogs with each user simulation.

N	$p > 0.90$	$p > 0.95$
50	0.08	0.12
100	0.06	0.09
200	0.05	0.07
500	0.04	0.05
1000	0.03	0.04

Table 7. Difference in normalized Cramér-von Mises divergence between two user simulations required for rank-ordering to be correct for various numbers of real dialogs at confidence intervals of $p > 0.90$ and $p > 0.95$. In the simulations used to produce this table, 1000 dialogs were used from each of the two user simulations.

training set (0.098) does not allow a statistically significant ordering to be inferred.

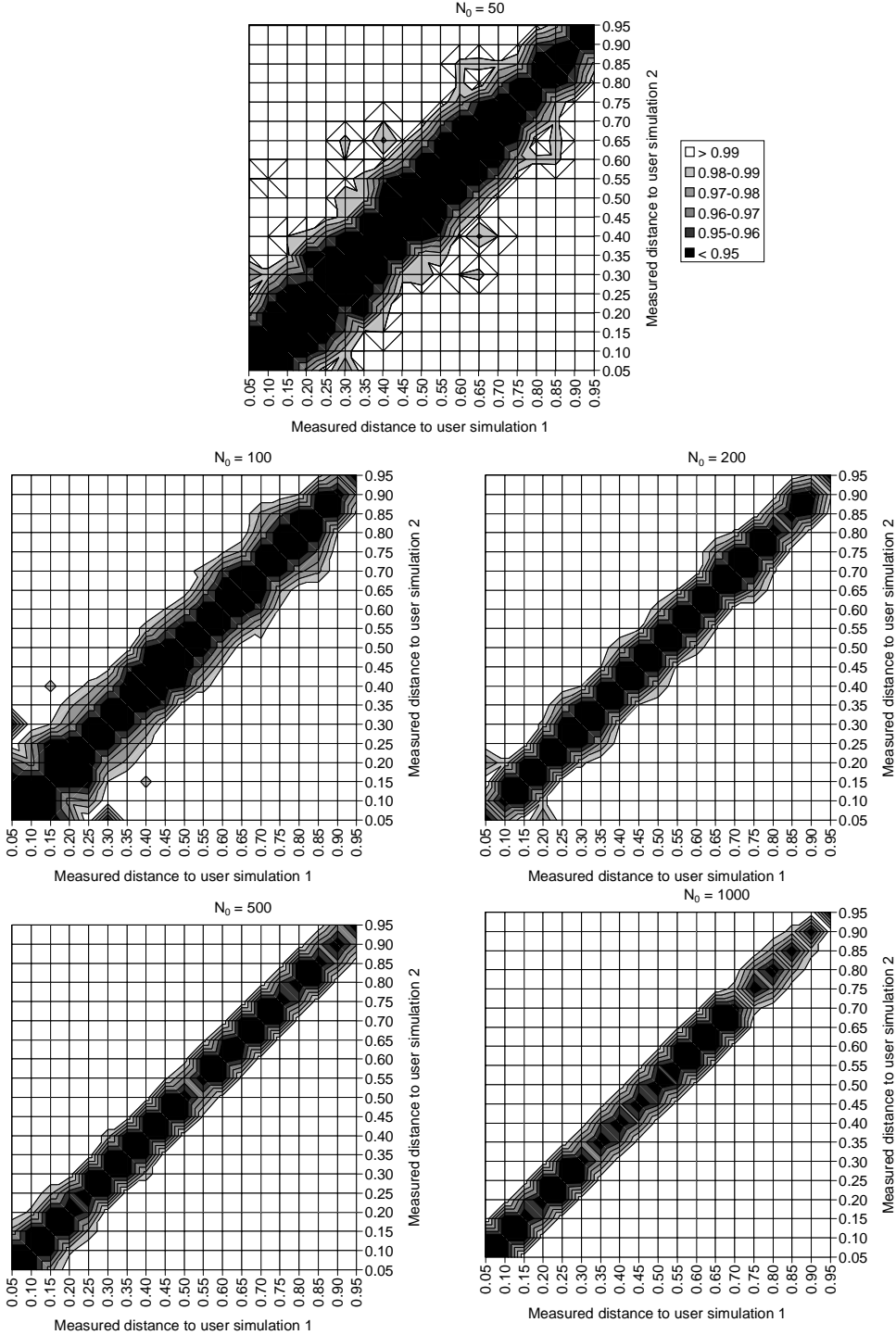


Fig. 11. Measured divergence to user simulation 1 $D(F_0||F_1)$ and user simulation 2 $D(F_0||F_2)$ vs. ordering reliability for $N_1 = N_2 = 1000$ dialogs with each user simulation and various numbers of “real” user dialogs N_0 .

6 Conclusions

In the paper, we have tackled the problem of evaluating and rank-ordering user simulations. This work has sought to provide system designers and prac-

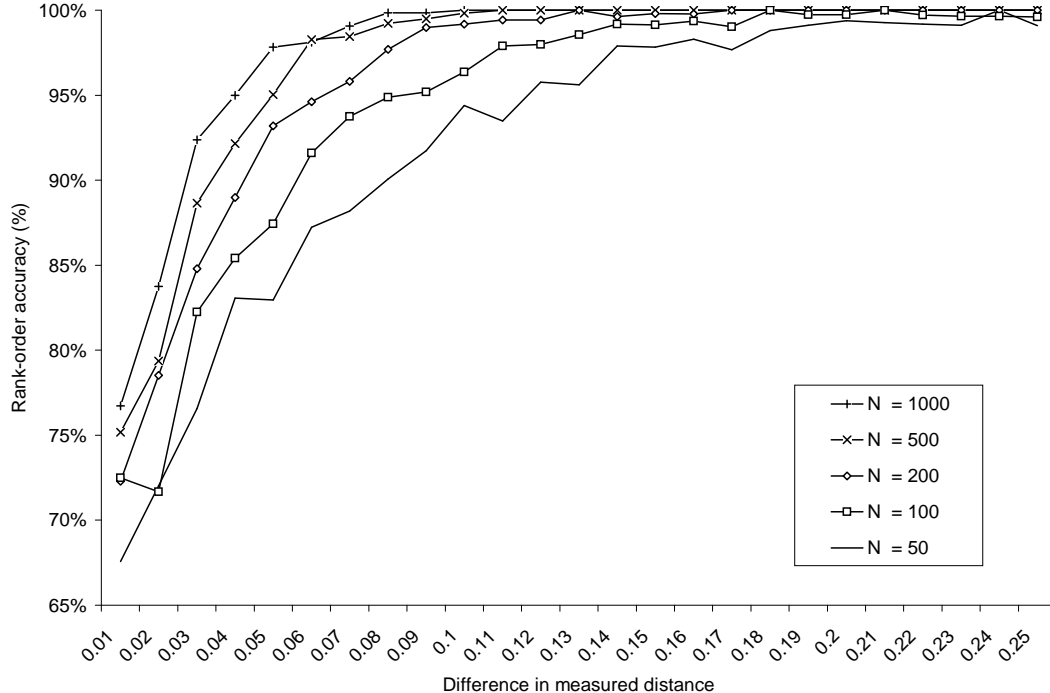


Fig. 12. *Difference* $|D_1 - D_2|$ between divergence to user simulation 1 (D_1) and user simulation 2 (D_2) vs. ordering reliability for $N_1 = N_2 = 1000$ simulated dialogs and various numbers of real dialogs N_0 .

tioners with a simple, principled method of evaluating and rank-ordering user simulations, based on the normalized Cramér-von Mises divergence.

We view a user simulation as a predictive tool: a dialog system interacting with a population of users will produce a distribution over these scores, and the aim of a user simulation is to predict this distribution as accurately as possible. This work has advocated using the normalized Cramér-von Mises divergence to measure the difference between the real and simulated score distributions. The normalized Cramér-von Mises divergence is attractive for this task because it makes no parametric assumptions, requires no tuning parameters, imposes minimal constraints on the implementation of the user simulation, accounts for the notion of samples from a “true” distribution and a “model” distribution, and provides a standardized, intuitive scale. Because different aspects of performance are important in different domains, this method relies on a domain-specific scoring function which assigns a real-valued score to each dialog in the system’s domain. This requirement is easily satisfied because this scoring function is already a prerequisite of many machine learning algorithms, such as Markov decision processes and partially observable Markov decision processes.

An illustration with a corpus of real dialogs collected from real system usage shows that as the user simulation is made increasingly realistic, the normal-

ized Cramér-von Mises divergence between the real dialogs and the synthetic dialogs decreases. This illustration supports our claim that the normalized Cramér-von Mises divergence indeed provides a measurement of the extent to which the user simulation is faithfully imitating real users. Further, a series of simulation experiments has explored what magnitude of difference in Cramér-von Mises divergences is required to infer a statistically significant rank-ordering, and we have developed a concise table that enables researchers and practitioners to judge whether the observed difference between two user simulations implies a true, statistically significant difference.

Although user simulations enable promising new techniques for spoken dialog systems to be realized, important work remains, and a broad program of research is required to understand what properties of a user simulation are optimal for creating new dialog systems with these new approaches. Nevertheless, obtaining a principled method for comparing user simulations on the same dialog system is an important first step. This method is straightforward to apply, concise to report, and easy to interpret, and we believe it has the capacity to play an important role in this area of research.

Acknowledgements

Thanks to Bob Bell for many insightful conversations in developing this method. Thanks also to Vincent Goffin for facilitating access to the voice dialer code and logs, and to Srinivas Bangalore for helpful comments about the presentation in this paper. Finally, thanks to the three anonymous reviewers for insightful comments and guidance.

Appendix 1: ASR model assumptions

In Section 3, it was assumed that the ASR model component of the user simulation is non-trivially stochastic and produces semantically meaningful and random errors a realistic fraction of the time, conditioned only on inputs available at run-time to a real speech recognizer. This appendix explains the importance of this assumption and the possible consequences of violating it.

If the ASR model does not introduce errors (or produces errors only trivially), then it is possible for the user behavior model to predict the output of the dialog system with certainty. When this occurs, the user behavior model can exercise complete control over what path the dialog follows. This control could enable a user behavior model to replicate a distribution of scores in a

corpus without achieving any generality – for example, by simply reproducing the dialogs in the corpus. Because this sort of user simulation would not encounter any unseen dialog situations, it is doubtful that it could make reliable predictions about dialog system performance with real users and real speech recognition. In other words, if the user behavior model in a user simulation can predict the output of the dialog system with certainty, then the user simulation could falsely appear to be a good predictor of system performance.

In reality of course, the output of the dialog system is not deterministic to a (real) user because the speech recognition introduces stochastic, non-trivial errors which are not predictable by the user. The speech recognizer cannot know whether it is making an error or not and thus, even with complete knowledge of the system, the user cannot deterministically control the output of the dialog system. The speech recognition errors cause the (real) user to confront new dialog situations and prevent the user from following a linear, pre-determined script.

Thus, realistic non-determinism in the ASR simulation is a necessary condition for this evaluation metric. This is formalized by assuming that the ASR model is non-trivially stochastic and produces semantically meaningful and random errors a realistic fraction of the time, conditioned only on inputs available at run-time to a real speech recognizer. In other words, for a given input to the ASR model, the output is drawn from a non-deterministic, semantically diverse distribution over outputs. At any point in a dialog, the same semantic input to the ASR model can generate meaningfully different semantic outputs.

This assumption prevents the user behavior model from predicting the output of the dialog system with certainty, while still allowing realistic aspects of the ASR process to be modelled. For example, the ASR model can condition on speaker gender/age or speech prosody. In addition, the ASR model can maintain internal state, which is required for the ASR model to simulate channel or speaker adaptation, in which recognition attempts early in a dialog may be more error-prone than later attempts.

Appendix 2: Derivation of α

The lower bound of $\sqrt{\sum_{i=1}^{N_0} (F_0(x_{(i)}^0) - F_1(x_{(i)}^0))^2}$ is 0: the quantity $(F_0(x_{(i)}^0) - F_1(x_{(i)}^0))^2$ is non-negative and is zero when F_0 and F_1 are co-incident.

The upper bound of $\sqrt{\sum_{i=1}^{N_0} (F_0(x_{(i)}^0) - F_1(x_{(i)}^0))^2}$ occurs when there exists an x such that $|F_0(x) - F_1(x)| = 1$ – i.e., when the ranges $[\min(\mathcal{S}_0), \max(\mathcal{S}_0)]$ and $[\min(\mathcal{S}_1), \max(\mathcal{S}_1)]$ are disjoint. In this case, $F_1(x)$ is either 0 or 1 throughout $[\min(\mathcal{S}_0), \max(\mathcal{S}_0)]$. If $F_1(x) = 0$ in this range, then the sum

$$\sqrt{\sum_{i=1}^{N_0} (F_0(x_{(i)}^0) - F_1(x_{(i)}^0))^2} \quad (10)$$

$$= \sqrt{\sum_{i=1}^{N_0} (F_0(x_{(i)}^0))^2} \quad (11)$$

$$= \sqrt{\sum_{i=1}^{N_0} \left(\frac{i - \frac{1}{2}}{N_0}\right)^2} \quad (12)$$

$$= \sqrt{\left(\frac{1}{N_0}\right)^2 \sum_{i=1}^{N_0} \left(i - \frac{1}{2}\right)^2} \quad (13)$$

$$= \sqrt{\left(\frac{1}{N_0}\right)^2 \left(\sum_{i=1}^{N_0} i^2 - \sum_{i=1}^{N_0} i + \sum_{i=1}^{N_0} \frac{1}{4}\right)} \quad (14)$$

$$= \sqrt{\left(\frac{1}{N_0}\right)^2 \left(\frac{N_0(N_0+1)(2N_0+1)}{6} - \frac{N_0(N_0+1)}{2} + \frac{N_0}{4}\right)} \quad (15)$$

$$= \sqrt{\frac{4N_0^2 - 1}{12N_0}} \quad (16)$$

If $F_1(x) = 1$ in the range $[\min(\mathcal{S}_0), \max(\mathcal{S}_0)]$, then the sum

$$\sqrt{\sum_{i=1}^{N_0} (F_0(x_{(i)}^0) - F_1(x_{(i)}^0))^2} = \sqrt{\sum_{i=1}^{N_0} (F_0(x_{(i)}^0) - 1)^2} \quad (17)$$

$$= \sqrt{\sum_{i=1}^{N_0} (1 - F_0(x_{(i)}^0))^2} \quad (18)$$

$$= \sqrt{\sum_{i=1}^{N_0} \left(1 - \frac{i - \frac{1}{2}}{N_0}\right)^2}. \quad (19)$$

Expanding this series yields the same series as above; thus, the normalization constant *alpha* is

$$\alpha = \sqrt{\frac{12N_0}{4N_0^2 - 1}} \quad (20)$$

Appendix 3: Derivation of β

The lower bound of $\sqrt{\int (P_0(x) - P_1(x))^2 p_0(x) dx}$ is 0: the factors $(P_0(x) - P_1(x))^2$ and $p_0(x)$ are both non-negative and is zero when $P_0(x)$ and $P_1(x)$ are zero.

The upper bound of $\sqrt{\int (P_0(x) - P_1(x))^2 p_0(x) dx}$ occurs when there exists an x such that $|P_0(x) - P_1(x)| = 1$. If such an x exists, then in the region where $p_0(x) > 0$, $P_1(x)$ is either 0 or 1.

In the case where $P_1(x) = 0$ where $p_0(x) > 0$:

$$\int (P_0(x) - P_1(x))^2 p_0(x) dx = \int P_0(x)^2 p_0(x) dx \quad (21)$$

Noting that $P_0'(x) = p_0(x)$ and integrating by parts:

$$\int P_0(x)^2 p_0(x) dx = P_0(x)^2 P_0(x) - \int P_0(x) (P_0(x)^2)' dx \quad (22)$$

$$\int P_0(x)^2 p_0(x) dx = P_0(x)^2 P_0(x) - \int P_0(x) 2P_0(x) p_0(x) dx \quad (23)$$

$$3 \int P_0(x)^2 p_0(x) dx = P_0(x)^3 \quad (24)$$

$$\int P_0(x)^2 p_0(x) dx = \frac{1}{3} P_0(x)^3 \quad (25)$$

$$\sqrt{\int P_0(x)^2 p_0(x) dx} = \sqrt{\frac{1}{3} P_0(x)^3} \quad (26)$$

In the case where $P_1(x) = 1$ where $p_0(x) > 0$, the same procedure as above can be followed, except that the integration by parts uses $\frac{d}{dx}(P_0(x) - 1) = p_0(x)$, which produces the same result.

Integrating from $-\infty$ to $+\infty$ yields $\sqrt{\frac{1}{3}}$. The normalization constant β is the inverse of this, i.e.,

$$\beta = \sqrt{3}. \quad (27)$$

References

- Ai, H., Litman, D. J., 2007. Knowledge consistent user simulations for dialog systems. In: Proc Eurospeech, Antwerp, Belgium.
- Anderson, T., 1962. On the distribution of the two-sample Cramér-von Mises criterion. *Annals of Mathematical Statistics* 33 (3), 1148–1159.
- Bui, T., Poel, M., Nijholt, A., Zwiers, J., 2007. A tractable DDN-POMDP approach to affective dialogue modeling for general probabilistic frame-based dialogue systems. In: Proc Workshop on Knowledge and Reasoning in Practical Dialogue Systems, Intl Joint Conf on Artificial Intelligence (IJCAI), Hyderabad, India. pp. 34–37.
- Cramér, H., 1928. On the composition of elementary errors. second paper: Statistical applications. *Skandinavisk Aktuarietidskrift* 11, 171–180.
- Cuayáhuitl, H., Renals, S., Lemon, O., Shimodaira, H., 2005. Human-computer dialogue simulation using hidden markov models. In: Proc IEEE

- Workshop on Automatic Speech Recognition and Understanding (ASRU), San Juan, Puerto Rico, USA. pp. 290–295.
- Denecke, M., Dohsaka, K., Nakano, M., 2004. Learning dialogue policies using state aggregation in reinforcement learning. In: Proc Intl Conf on Spoken Language Processing (ICSLP), Jeju, Korea. pp. 325–328.
- Eadie, W., Drijard, D., James, F., Roos, M., Sadoulet, B., 1971. *Statistical Methods in Experimental Physics*. North Holland.
- Filisko, E., Seneff, S., 2005. Developing city name acquisition strategies in spoken dialogue systems via user simulation. In: Proc SIGdial Workshop on Discourse and Dialogue, Lisbon, Portugal.
- Frampton, M., Lemon, O., 2006. Learning more effective dialogue strategies using limited dialogue move features. In: Proc Association for Computational Linguistics (ACL), Sydney. pp. 185–192.
- Georgila, K., Henderson, J., Lemon, O., 2005. Learning user simulations for information state update dialogue systems. In: Proc Eurospeech, Lisbon, Portugal. pp. 893–896.
- Georgila, K., Henderson, J., Lemon, O., 2006. User simulation for spoken dialogue systems: Learning and evaluation. In: Proc Intl Conf on Spoken Language Processing (ICSLP), Pittsburgh, Pennsylvania, USA.
- Goddeau, D., Pineau, J., 2000. Fast reinforcement learning of dialog strategies. In: Proc Intl Conf on Acoustics, Speech, and Signal Processing (ICASSP), Istanbul. pp. 1233–1236.
- Heeman, P., 2007. Combining reinforcement learning with information-state update rules. In: Proc Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT), Rochester, New York, USA. pp. 268–275.
- Henderson, J., Lemon, O., Georgila, K., 2005. Hybrid reinforcement/supervised learning for dialogue policies from Communicator data. In: Proc Workshop on Knowledge and Reasoning in Practical Dialogue Systems, Intl Joint Conf on Artificial Intelligence (IJCAI), Edinburgh. pp. 68–75.
- Kolmogorov, A., 1933. Sulla determinazione empirica di una legge di distribuzione. *Giorn. Ist. Ital. Attuari* 4, 1–11.
- Kullback, S., Leibler, R., 1951. On information and sufficiency. *Annals of Mathematical Statistics* 22, 79–86.
- Lemon, O., Georgila, K., Henderson, J., 2006. Evaluating effectiveness and portability of reinforcement learned dialogue strategies with real users: the TALK TownInfo evaluation. In: Proc Workshop on Spoken Language Technologies (SLT), Aruba. pp. 178–181.
- Levin, E., Pieraccini, R., 1997. A stochastic model of computer-human interaction for learning dialog strategies. In: Proc Eurospeech, Rhodes, Greece. pp. 1883–1886.
- Levin, E., Pieraccini, R., 2006. Value-based optimal decision for dialog systems. In: Proc Workshop on Spoken Language Technologies (SLT), Aruba. pp. 198–201.

- Levin, E., Pieraccini, R., Eckert, W., 2000. A stochastic model of human-machine interaction for learning dialogue strategies. *IEEE Trans on Speech and Audio Processing* 8 (1), 11–23.
- Pietquin, O., 2004. A framework for unsupervised learning of dialogue strategies. Ph.D. thesis, Faculty of Engineering, Mons (TCTS Lab), Belgium.
- Pietquin, O., Renals, S., 2002. ASR modelling for automatic evaluation and optimisation of dialogue systems. In: *Proc Intl Conf on Acoustics, Speech, and Signal Processing (ICASSP)*, Florida, USA. pp. 46–49.
- Roy, N., Pineau, J., Thrun, S., 2000. Spoken dialogue management using probabilistic reasoning. In: *Proc Association for Computational Linguistics (ACL)*, Hong Kong. pp. 93–100.
- Schatzmann, J., Georgila, K., Young, S., 2005. Quantitative evaluation of user simulation techniques for spoken dialogue systems. In: *Proc SIGdial Workshop on Discourse and Dialogue*, Lisbon, Portugal. pp. 178–181.
- Schatzmann, J., Thomson, B., Young, S., 2007a. Error simulation for training statistical dialogue systems. In: *Proc IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Kyoto, Japan. pp. 526–531.
- Schatzmann, J., Thomson, B., Young, S., 2007b. Statistical user simulation with a hidden agenda. In: *Proc SIGdial Workshop on Discourse and Dialogue*, Antwerp, Belgium. pp. 273–282.
- Scheffler, K., 2002. Automatic design of spoken dialogue systems. Ph.D. thesis, Cambridge University.
- Scheffler, K., Young, S., 2001. Corpus-based dialogue simulation for automatic strategy learning and evaluation. In: *Proc North American Association for Computational Linguistics (NAACL) Workshop on Adaptation in Dialogue Systems*, Pittsburgh, USA. pp. 12–19.
- Scheffler, K., Young, S., 2002. Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning. In: *Proc Human Language Technologies (HLT)*, San Diego, USA. pp. 12–18.
- Singh, S., Litman, D., Kearns, M., Walker, M., 2002. Optimizing dialogue management with reinforcement learning: experiments with the NJFun system. *Journal of Artificial Intelligence* 16, 105–133.
- Stephens, M., 1992. Introduction to: Kolmogorov (1933) on the empirical determination of a distribution. In: Kotz, S., Johnson, N. (Eds.), *Breakthrough in statistics*. Vol. II. Springer Verlag, pp. 93–105.
- Stuttle, M., Williams, J., Young, S., 2004. A framework for Wizard-of-Oz experiments with a simulated ASR channel. In: *Proc Intl Conf on Spoken Language Processing (ICSLP)*, Jeju, Korea. pp. 241–244.
- Thomson, B., Schatzmann, J., Welhammer, K., Ye, H., Young, S., 2007. Training a real-world POMDP-based dialog system. In: *Proc Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT) Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, Rochester, New York, USA. pp. 9–17.
- von Mises, R., 1931. *Wahrscheinlichkeitsrechnung und ihre Anwendung in der*

- Statistik und Theoretischen Physik. F Deuticke.
- Walker, M., Fromer, J., Narayanan, S., 1998. Learning optimal dialogue strategies: a case study of a spoken dialogue agent for email. In: Proc Association for Computational Linguistics and Intl Conf on Computational Linguistics (ACL/COLING), Montreal. pp. 1345–1351.
- Walker, M., Kamm, C., Litman, D., 2000. Towards developing general models of usability with PARADISE. Natural Language Engineering: Special Issue on Best Practice in Spoken Dialogue Systems.
URL citeseer.ist.psu.edu/article/walker00towards.html
- Walker, M., Rudnicky, A., Prasad, R., Aberdeen, J., Bratt, E., Garofolo, J., Hastie, H., Le, A., Pellom, B., Potamianos, A., Passonneau, R., Roukos, S., Sanders, G., Seneff, S., Stallard, D., 2002. DARPA Communicator: Cross-system results for the 2001 evaluation. In: Proc Intl Conf on Spoken Language Processing (ICSLP), Colorado, USA. pp. 269–272.
- Williams, J., 2006. Partially observable Markov decision processes for spoken dialogue management. Ph.D. thesis, Cambridge University.
- Williams, J., 2007. Applying POMDPs to dialog systems in the troubleshooting domain. In: NAACL-HLT Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies, Rochester, New York, USA. pp. 1–8.
- Williams, J., Young, S., 2007a. Scaling POMDPs for spoken dialog management. IEEE Trans. on Audio, Speech, and Language Processing 15 (7), 2116–2129.
- Williams, J. D., Young, S., 2007b. Partially observable Markov decision processes for spoken dialog systems. Computer Speech and Language 21 (2), 393–422.
- Young, S., Schatzmann, J., Weilhammer, K., Ye, H., 2007. The hidden information state approach to dialog management. In: Proc Intl Conf on Acoustics, Speech, and Signal Processing (ICASSP), Honolulu, Hawaii, USA. pp. IV149–IV152.
- Zhang, B., Cai, Q., Mao, J., Chang, E., Guo, B., 2001. Spoken dialogue management as planning and acting under uncertainty. In: Proc Eurospeech, Aalborg, Denmark. pp. 2169–2172.