Corpus-based dialogue simulation for automatic strategy learning and evaluation

Konrad Scheffler and **Steve Young**
Department of Engineering, Cambridge University, Cambridge, UK
email: khs22,sjy@cam.ac.uk

Abstract

This paper describes a method for simulating mixed initiative human-machine dialogues using data collected by a prototype dialogue system. The behaviour of the user population is modelled probabilistically using an explicit representation of user state. Recognition and understanding errors are also modelled. The simulation can be used for evaluation of competing strategies, as well as automatic learning of dialogue strategies.

1 Introduction

1.1 Approaches to automatic strategy learning

Automatic design of dialogue strategy by reinforcement learning has been proposed by several authors. There are two types of model that can be used for this purpose. The first (as used by Singh et al. (1999; 2000) and Roy et al. (2000)) is a model of dialogue state transitions, which can be estimated directly from a corpus in which these transitions have been logged. The model is then used with a model-based reinforcement learning algorithm such as dynamic programming to find the optimal policy.

An alternative approach (taken here, and also by Eckert, Levin and Pieraccini (Eckert et al., 1997; Levin et al., 1998), and Goddeau and Pineau (2000)) is to use a model of the user behaviour and system recognition performance to generate training episodes by simulation. The optimal policy is then found using a simulation-based reinforcement learning algorithm such as Q-learning.

In the case of the model-based approach, the dialogue state transitions have to be logged (ideally during data collection), which means that the system state representation must be fixed at an early stage. For the simulation-based approach, on the other hand, the system state representation need never be fixed. Thus if the chosen system state representation proves problematic during training, it can be changed and the system retrained without having to change the data set, data transcriptions or user model. This flexibility, which allows the utilization of general data corpora that were collected prior to the research described here, is our main motivation for adopting the simulation-based approach.

1.2 Dialogue simulation

User simulations previously used for this purpose have been very simplistic (eg. dialogues are modelled using an utterance bigram model (Eckert et al., 1997; Eckert et al., 1998), without explicit error modelling or constraints to ensure user consistency), making them insufficient for use with most real applications. We therefore developed a more sophisticated system to simulate cooperative, task oriented human-machine dialogues, in order to serve as a tool for automatic evaluation and reinforcement learning design of dialogue systems (an initial version of this work is described in (Scheffler and Young, 1999; Scheffler and Young, 2000)).

The method is based on probabilistic modelling of both user behaviour and system errors, with the models being constructed using data collected by a prototype dialogue system and making use of an explicit model of user state. While the simulation system was developed in the context of a specific application, care was taken to ensure that it is domain independent. The results presented in this paper were produced on a different application from the one on which the original version of the system was developed.

1.3 Overview

Section 2 presents a brief summary of the most important points of the system, in particular the modelling of user state, some aspects regarding the representation of intentions, and the process by which user utterances are simulated by traversing a lattice of intentions. Next, section 3 discusses the training of the model. Finally, section 4 presents some experimental results according to which the system performance can be judged and draws some conclusions.

2 Approach to simulation

2.1 Goal directed user model

Our previous work on user modelling (Scheffler and Young, 2000) indicated that it is useful to adopt a

Goal field	Value	Status
Type:	FILMLIST	specified
Instructions:	NA	-
Town:	NA	-
Film:	NA	NA
Cinema:	GOODCINEMA	urgent
Day:	GOODDAY	pending

Figure 1: Example of a user goal and associated status variables in the cinema application.

goal directed approach, forcing simulated user actions to be consistent with one another throughout the course of the dialogue. This is in contrast to earlier approaches that modelled user utterances using either a deterministic rule-based approach, or an utterance bigram with no constraints to ensure user consistency. Here we extend our approach by modelling the user state explicitly and in more detail.

A *user goal* can be defined as a specification of the particular dialogue transaction that the user wants to complete. The goal directed user model is then built on the central assumption that *the user always attempts to act in accordance with some goal, which remains fixed until it has been completed.* Any deviations from such behaviour are viewed as errors in formulation, which are modelled probabilistically by the error generation module.

The goal is a simple attribute-value structure consisting of a number of goal fields, each of which can either be uninstantiated, or take one of a field-specific set of values. In addition to these values, which remain fixed until completion of the goal, each attribute may be associated with a status variable that tracks the state of each goal field from the user's point of view. A goal field may be pending (the value has yet to be specified), specified (the value has already been specified and should not be repeated), urgent (the value has high priority to be specified, usually because the dialogue system is believed to have made an error on this field) or not applicable (not to be specified at any point in the current transaction).

Figure 1 shows an example of a goal structure for a cinema information system. In this example, the user is trying to accomplish a "Film listing" goal, to find out which films are showing at a particular cinema on a particular day. The "Instructions", "Town", and "Film" fields are used for other goal types, but are not applicable for this type, while the values of the "Type", "Cinema" and "Day" fields need to be specified. Field values prefixed by "GOOD" or "BAD" denote respectively the correctly and incorrectly recognised versions of content intentions (see section 2.2). The "Instructions" and "Town" fields are not associated with status variables, since in this application they are always prompted for explicitly and in isolation. The other status variables indicate that, at this point in the dialogue, the user has already specified the goal type (and believes the system to have understood it correctly), needs to specify the cinema name with increased urgency (possibly it has been detected from a system prompt that the system has misrecognised the cinema name), and has yet to specify the day.

The full description of user state used in the model comprises the following:

- The current contents of the user goal
- The status variables associated with the goal fields.
- Secondary goal structures representing the user beliefs on the current system goal and the newly completed goal (if any).
- Any necessary application specific variables describing items such as the type (eg. direct question) of the previous prompt.

2.2 Intention language

An *intention* can be defined as the minimum piece of information that can be conveyed independently within a given application, and represents the lowest level of information we deal with in our simulations. Apart from intentions that correspond directly to specific semantic tags in the application, the following are often useful to define:

2.2.1 Content intentions

Many semantic tags such as "Monday" contain some information (the particular day referred to) that, while necessary to be communicated correctly, is not conceptually relevant to the action of the dialogue system. Thus, while it is important whether or not the content of the tag is recognised correctly, it may not be relevant for the purpose of simulation whether a correctly recognised day is Monday or Tuesday.

For cases such as this, it makes sense to replace the full set of possible tags (a different one for each day of the week) with only two intentions: one for the correct and one for some incorrect value. We refer to these as "content intentions", to indicate that the intention implies some further information content that is known but not encoded.

2.2.2 Null intentions

A null intention is one that indicates the absence of some intention that might have been specified in an utterance. For example, it may be of interest whether, after specifying a cinema name, the user goes on to specify a day in the same utterance. The absence of a day intention in an utterance can be detected and marked by adding a "day-null" intention to the representation of the utterance. Using

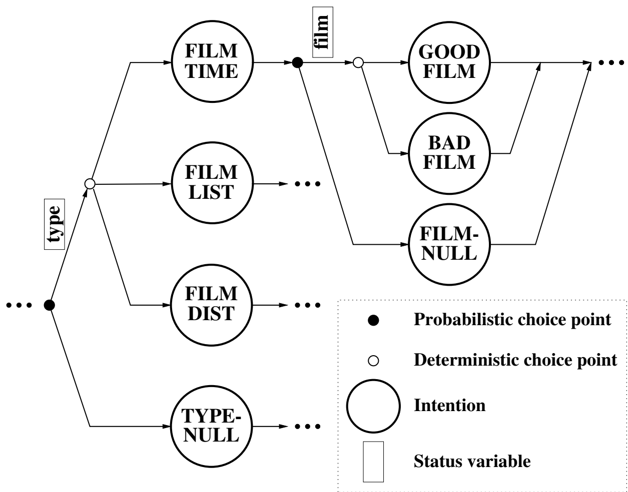

Figure 2: Example of a lattice segment used for utterance generation.

such null intentions removes the need to model insertion and deletion errors during recognition explicitly, since these just become special cases of substitution errors, where the substitution involves a null intention.

2.2.3 Mumble intention

One of the most common types of utterance in human-machine dialogue is (unfortunately) the case where the user says something that is completely unintelligible to the machine, often because it is not part of the system vocabulary. This case is represented by the "mumble" intention: it is useful to model explicitly because of the large number of substitutions in which it plays a role.

2.3 Utterance generation lattices

A dialogue simulation is performed by simply running a modified version of the dialogue system software, with the input/output functions overridden by user modelling software. The process of generating a user utterance is similar to that described in (Scheffler and Young, 2000). The main structural component of the user model is a set of lattices used for utterance generation. These are derived from the grammars used by the recognition engine, with the difference that the lattice edges are associated with intentions rather than words. Figure 2 illustrates a segment of one of the lattices used for the cinema application, which might generate the utterance "FILMTIME GOODFILM". This is an intention level representation of sentences like "Could you tell me when Star Wars is on?" or "I need the times for The Matrix".

2.3.1 Lattice structure

The nodes of the lattice correspond to choice points (either probabilistic or deterministic) where the user chooses among the available options. The distinction between the two types of choice point is another difference between the recognition and generation lattices. This separation between choices that are deterministic decisions based on the user state and those that are to be modelled probabilistically, is the core of the goal-directed user model.

When the lattice is constructed, nodes that have similar structure (i.e. nodes that give the user the same options) are grouped together and assigned to *backoff groups*. During parameter estimation, nodes for which data is sparse are backed off to parameters estimated for the entire backoff group. For example, the first node in Figure 2 is a probabilistic choice between specifying the transaction type and omitting it for the moment. The same choice also occurs in other lattices (used in different dialogue contexts), so that multiple instances of it can be grouped together. If data for this particular choice point is sparse, we can then use the data collected for the entire group to find out how users resolved the same choice in different dialogue contexts.

Some lattice edges are linked to goal field status variables. This causes the edges to be disallowed or assigned a priority according to the current status, and the status to be updated when the edges are traversed.

2.3.2 Utterance generation

During utterance generation, the lattice is traversed, with the following steps being carried out for each node visited:

1. The allowable options (edges) are identified by consulting the status of associated goal fields.
2. The choice is resolved by selecting one of the options, either deterministically or probabilistically depending on the type of choice point. The outcome of a deterministic choice depends on the current user state (often the action taken is simply to specify the value of a particular goal field, but sometimes a more complex application dependent procedure may be invoked), while probabilistic choices are resolved using the probabilities associated with the allowable options.
3. If the selected option has an intention connected to it, it is subjected to error generation, during which the effect of recognition/understanding errors is modelled by performing a probabilistic substitution on the selected intention. This may involve jumping to an edge elsewhere in the lattice.

4. The destination of the edge (possibly changed by the substitution) is identified as the next choice point.

Traversal of the lattice continues by processing each successive node until an exit node is reached.

2.3.3 Example

In the example of Figure 2, we start at the probabilistic choice point on the left. The first option is associated with the status of the "type" goal field. Assume that this variable is currently set to "pending": in this case all options are allowed and a probabilistic choice is made based on how often users specified the goal type in this situation in the training data. Suppose the first option is taken: we set the "type" status to "specified" and proceed to the next choice point. This is a deterministic choice governed by the value of the "type" field in the goal. If this value is "FILMLIST", the second option is chosen. However, this edge has an intention connected to it which can be misrecognised. Another probabilistic choice is made based on how often, in this context, the "FILMLIST" intention is misrecognised as each of the possible intentions that could have followed the first choice point (including the "TYPE-NULL" intention). Suppose we get a misrecognition by generating a substitution of "FILMTIME" for "FILMLIST". We append the "FILMTIME" intention to the generated utterance, jump to the choice point following the "FILMTIME" intention, and continue from there until the end of the lattice is reached.

3 Training

We now discuss the problem of estimating the various parameters from data (the data set used in our experiments is discussed in section 4.1.2). This is non-trivial, since we do not have direct knowledge of either the user or the system state when analysing dialogue transcriptions. It is important, however, to model the fact that the user's actions depend on the particular circumstances s/he is faced with, which include not only the system actions, but also the internal user state. Thus inferring the user state is essential before parameters describing user actions can be estimated.

3.1 User state inference

3.1.1 User goal

Since the user goal is assumed to be constant until it is completed, it need only be estimated once for each subdialogue, where the end of a subdialogue is marked by the completion of some transaction. It can then be used while tracking other aspects of user state from utterance to utterance. Goal inference is performed by scanning each user utterance in the subdialogue and finding, for each goal field, the most recently specified value for that field.

3.1.2 Other user state parameters

The remaining user state variables change throughout the course of the transaction, and must be kept updated as the dialogue progresses during parameter estimation. The necessary steps include extracting information from system prompts that have an impact on user state, adjusting the status variables as the user specifies goal field values, and running any application specific procedures that would have been run during simulation, since these may have the ability to change the user state.

To take account of the continuity of the user state in real dialogues, the state is reset only at the start of a new dialogue and not between successive transactions in a dialogue. This has the effect of retaining contextual information so that the user can refer to something stated in an earlier transaction of the same dialogue without explicitly stating it again.

3.2 Parameter estimation

The parameters for the model are in the form of probabilities or frequency counts from which probabilities can be obtained. In all cases, probabilities are obtained using simple maximum likelihood:

$$P(Event|Context) = \frac{cnt(Event, Context)}{cnt(Context)}, \quad (1)$$

where $cnt(Event, Context)$ is the counted number of occurrences of the event in question in a specific context and $cnt(Context)$ is the counted number of occurrences of the context irrespective of whether or not the event occurred. Counts are obtained for both the exact context and a wider backoff context, so that the backoff context can be used where data sparsity occurs.

In order to establish which parameters are to be updated, both the reference (i.e. actually spoken) and recognised (i.e. with recognition errors) versions of each training utterance must be parsed so that the exact sequence of choice points is known. The path followed by the reference utterance is then used to estimate parameters for the user model, while the error model is obtained from a comparison of the reference and recognised versions.

3.2.1 Modelling user actions

Once the required lattice paths have been found, the lattice is traversed along the reference path, and the following model updating steps are performed at each node:

- Find out which options had "pending" and "urgent" status.
- If there was an "urgent" option, update the counts for the urgent probability according to whether or not the chosen option was urgent.

- Identify the context as the specific node in the lattice, with the backoff context being the set of all nodes in the same backoff group.
- If the chosen option was either "pending" or "urgent", and at least one alternative was open to the user (i.e. an unchosen option marked "urgent" or "pending"), increment the context occurrence count for all available options and the event occurrence count for the chosen option.

Note that if the chosen option was not "urgent" or "pending", the user action was not goal directed. No parameters are updated in this case. Also, automatic choices where the user had no valid alternative are not counted and context occurrence counts are not incremented for options that had status settings making them unavailable.

3.2.2 Error modelling

In addition, the following updating steps are performed at each node where an intention was produced:

- Find the identical context in the recognition path through the lattice. Updating is only done if this context was actually visited in the recognised utterance.
- Content intentions are treated by using separate intentions for correct and incorrect recognition. The reference intention is always assumed to be correct.
- Identify the context choice point: this is the point in the lattice immediately after the last intention was produced, or the initial choice point in the lattice if no intentions have yet been produced. The full context consists of both this context choice point and the reference intention. The backoff context is just the reference intention, with the point in the lattice being ignored.
- Increment the context and event occurrence counts for intentions as appropriate.

By processing the entire data set in this way, counts are obtained from which, during simulation, relative probabilities can be computed for the options that are available at any particular time. Since counts have been obtained for both specific and more general contexts, it is possible to back off to the general case whenever the number of training examples falls below a preset threshold.

4 Experiments

4.1 Experimental setup

The purpose of the experiments reported here is to ascertain to what extent the simulated dialogues give an accurate picture of the dialogues produced by real users using the system in the task domain. We therefore trained a user model on data collected using an existing dialogue system, and compared the resulting simulations with real dialogues from the same system. Unfortunately, comparing the simulation results to real dialogues is problematic due to the fact that the user goals in spontaneous real dialogues are unknown.[1] For this reason, accurate goal achievement rates cannot be calculated for the real data, so we cannot therefore evaluate the accuracy of the goal achievement rates estimated by the simulation. Instead, we restrict ourselves to comparisons between goal completion times for test data and simulations.

4.1.1 Application and user goals

The application used for these experiments is a system providing cinema information over the telephone. It allows the user a large amount of freedom in the way s/he can set about requesting information: a large range of syntactical constructs can be used, giving the user the option of stringing intentions together almost arbitrarily in an utterance. Also, the options available to the user are not constrained severely by the dialogue state, so that user actions depend more on the state of the user than that of the system.

The following user goals were defined according to the functionality offered to the user:

- Instructions: Obtain instructions for using the system on entering the dialogue.
- Town: Specify the town about which information is desired.
- Film times: Find out at what times a specific film is being shown at a specific cinema on a specific day.
- Film listing: Find out which films are being shown at a specific cinema on a specific day.
- Film distribution: Find out at which cinemas a specific film is being shown on a specific day.
- Cinema address: Find out the address and booking number for a specific cinema.
- Exit: Close the dialogue by eliciting the exit message from the system. This involves communicating that the user has no further goals. (All goal fields are uninstantiated). The exit goal fails if the user fails to communicate this

[1] In general, it is impossible to ascertain which of the achieved goals in real dialogues are in fact what the users intended and which are the results of misunderstandings. While this problem can be addressed during data collection by asking callers to act out specific scenarios, such an approach has also been found to present problems, particularly lack of naturalness in the resulting dialogues (Sturm et al., 1999).

and is forced to hang up without concluding the conversation politely.

These goals can be combined in sequences to create specific scenarios to be simulated. Unfortunately, the choice of scenario can have a large effect on goal completion times, and it is not always clear which scenarios should be chosen.[2] For instance, the cinema address goal will usually only be attempted after the desired cinema name has been mentioned by either the user or the system while pursuing one of the other main goal types. This means that the cinema name need not be respecified by the user, which shortens the goal completion time. In the experiments reported here, this goal was attempted after a film listing goal. Other goals were attempted in isolation, except for the film times goal, for which we defined two variants: "Film times (1)" attempts the goal in isolation, while "Film times (2)" attempts it after successful conclusion of a film listing goal. The same test data was used for both variants.

4.1.2 Data corpus

The data used for these experiments was collected by recording trial runs of the cinema information system, with real users calling in to use the system. The corpus consists of 911 dialogues with a total of 7409 utterances. Of these dialogues, 134 were failed conversations where the user did not accomplish any goals.

The training data are in the form of dialogue transcriptions in the tag-level language of the dialogue system. Both a hand-transcribed reference version and the machine-recognised version are available for each conversation in the data set. The reference transcriptions have been recognised and parsed by humans according to the lattice used by the recognition engine.

Because of the high variance of the various goal completion times over the corpus, it is undesirable to use a small proportion of the data as a test set. To overcome this problem, the data was partitioned into 10 sections, each of which was used as a test set while the remaining 9 were used for training. The results over all 10 such runs are combined to obtain a better picture of system performance.

4.1.3 Discussion

There are a number of reasons why the results of such comparisons can only be expected to give a rough indication of the system performance. The most important of these is the low level of reliability of the figures obtained by analysis of the real data. This analysis assumes, for instance, that goal completion times are independent of whether the goal is achieved correctly or not. It also neglects the effect of goals being attempted in different dialogue contexts (different scenarios), while the proper scenarios to be used for simulation purposes are unknown. Further, since the corpus is limited in size, it is also very sensitive to outliers in the data caused by individual callers who may be experimenting with or testing the system rather than using it in the normal fashion. It is important, therefore, that the measured goal completion times should not be expected to give more than a relative indication of how the times for different goal types compare with each other.

[2] Since the corresponding scenarios for real users in the test data are unknown, we are forced to assume that the transaction performance in the scenarios chosen for simulation is representative of that in the entire set of possible scenarios.

4.2 Simulation results and conclusions

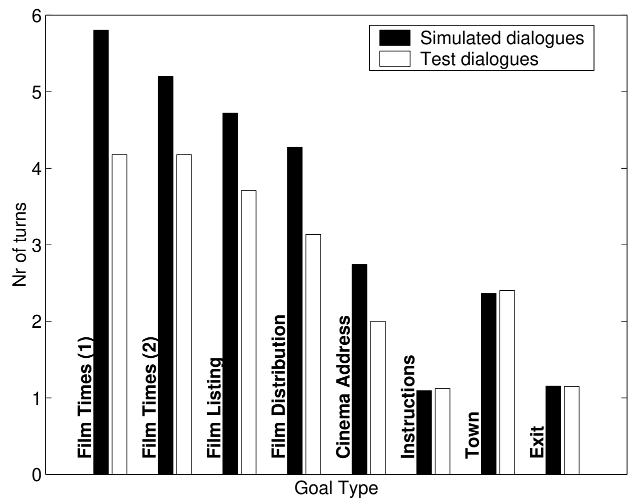

Figure 3: Average goal completion times (number of turns) for different goal types in simulated and test dialogues.

The simulation system implemented for the cinema application has 135 probabilistic choice points and 653 substitution contexts for which a total of 326 choice probabilities and 3069 substitution probabilities (many of them backed off to more general cases) are estimated respectively. The system was trained on each of the training sets, and used to simulate 1000 dialogues. The average goal completion times for both the simulation system and the test data are shown in figure 3, for the various goal types.

From these results it is clear that there is some agreement between the simulations and the test data, but the simulations fail to predict the actual durations accurately. A very good agreement is observed for the simple goals: each of "Instructions", "Town" and "Exit" involve communicating the contents of only one goal field to the system (for the "Town" goal a confirmation is done, which accounts for the extra transaction length). The other goals are more complex, involving communication of various combinations of goal fields, with the time taken to complete the goal in both the simulations and the real data roughly correlating with the amount of information that needs to be communicated. These goals prove considerably more difficult to simulate accurately. We observe that they take consistently longer to accomplish in the simulations than in the real dialogues. Perhaps the most important factor influencing this is the fact that goals can be attempted in different contexts. The effect of varying the context is illustrated by the difference in performance for the two film times goals: the version where an arbitrary context is assumed (Film times (2)) agrees substantially better with the test data than the initial version where no context is used. In general, the effect of discarding inter-goal context in this application is to increase the predicted goal achievement time.

The fact that the model is sensitive to variations in the input data encourages the use of different models for different user populations (such as experienced users and complete novices) where separate data is available. This would allow a dialogue system to be optimised for different types of users, or user populations consisting of combinations of the different user types. It also opens the possibility for designing adaptive systems that estimate user expertise online and adapt their strategy accordingly.

It is important to note that, in spite of the discrepancies between the real and simulated dialogues, the simulations do manage to recognise the relative differences between the different goal complexities and thus predict their relative completion times. For the purpose of relative evaluation of dialogue strategies, the simulations should thus be sufficient even if it is not advisable to rely on the prediction of dialogue duration as an absolute measure.

5 Summary

The problem addressed in this paper is the development of a simulation system for human-machine dialogues, and in particular the modelling of user actions and speech recognition performance.

Our approach to the problem of user modelling is based on the assumption that the user acts in a consistent and goal directed fashion, and a detailed model of the internal user state. In addition, we incorporate a context specific model of recognition and understanding errors, leading to a system that is capable of producing realistic looking simulations of dialogues with complex structure. The performance of the simulation system was evaluated by means of comparative experiments on real data, which demonstrated that it simulates different scenarios well enough to perform relative predictions of their durations.

The system can be used for automatic design or evaluation of dialogue strategies. We are currently investigating its application to training a real dialogue application using reinforcement learning.

References

W. Eckert, E. Levin, and R. Pieraccini. 1997. User modelling for spoken dialogue system evaluation. *Proc. IEEE ASR Workshop*.

W. Eckert, E. Levin, and R. Pieraccini. 1998. Automatic evaluation of spoken dialogue systems. Technical Report TR98.9.1, AT&T Labs Research.

D. Goddeau and J. Pineau. 2000. Fast reinforcement learning of dialog strategies. *Proc. ICASSP*.

E. Levin, R. Pieraccini, and W. Eckert. 1998. Using markov decision process for learning dialogue strategies. *Proc. ICASSP*.

N. Roy, J. Pineau, and S. Thrun. 2000. Spoken dialogue management using probabilistic reasoning. *Proc. Association for Computational Linguistics*.

K. Scheffler and S.J. Young. 1999. Simulation of human-machine dialogues. Technical Report CUED/F-INFENG/TR 355, Cambridge University Engineering Dept.

K. Scheffler and S.J. Young. 2000. Probabilistic simulation of human-machine dialogues. *Proc. ICASSP*, pages 1217–1220.

S. Singh, M. Kearns, D. Litman, and M. Walker. 1999. Reinforcement learning for spoken dialogue systems. *Proc. NIPS*.

S. Singh, M. Kearns, D. Litman, and M. Walker. 2000. Empirical evaluation of a reinforcement learning spoken dialogue system. *Proc. AAAI*.

J. Sturm, E. den Os, and L. Boves. 1999. Issues in spoken dialogue systems: Experiences with the dutch arise system. *ESCA Workshop on Interactive Dialogue in Multi-Modal Systems*, pages 1–4.