Optimising Incremental Dialogue Decisions Using Information Density for Interactive Systems

Nina Dethlefs, Helen Hastie, Verena Rieser and Oliver Lemon
School of Mathematical and Computer Sciences
Heriot-Watt University, Edinburgh, Scotland
n.s.dethlefs | h.hastie | v.t.rieser | o.lemon@hw.ac.uk

Abstract

Incremental processing allows system designers to address several discourse phenomena that have previously been somewhat neglected in interactive systems, such as backchannels or barge-ins, but that can enhance the responsiveness and naturalness of systems. Unfortunately, prior work has focused largely on deterministic incremental decision making, rendering system behaviour less flexible and adaptive than is desirable. We present a novel approach to incremental decision making that is based on *Hierarchical Reinforcement Learning* to achieve an interactive optimisation of Information Presentation (IP) strategies, allowing the system to generate and comprehend backchannels and barge-ins, by employing the recent psycholinguistic hypothesis of *information density (ID)* (Jaeger, 2010). Results in terms of average rewards and a human rating study show that our learnt strategy outperforms several baselines that are not sensitive to ID by more than 23%.

1 Introduction

Recent work on incremental systems has shown that adapting a system's turn-taking behaviour to be more human-like can improve the user's experience significantly, based on incremental models of automatic speech recognition (ASR) (Baumann et al., 2011), dialogue management (Buss et al., 2010), and speech generation (Skantze and Hjalmarsson, 2010). All of these approaches are based on the same general abstract architecture of incremental processing (Schlangen and Skantze, 2011). While this architecture offers inherently incremental mechanisms to update and revise input hypotheses, it is affected by a number of drawbacks, shared by deterministic models of decision making in general: they rely on hand-crafted rules which can be time-consuming and expensive to produce, they do not provide a mechanism to deal with uncertainty introduced by varying user behaviour, they are unable to generalise and adapt flexibly to unseen situations, and they do not use automatic optimisation. Statistical approaches to incremental processing that address some of these problems have been suggested by Raux and Eskenazi (2009), who use a cost matrix and decision theoretic principles to optimise turn-taking in a dialogue system under the constraint that users prefer no gaps and no overlap at turn boundaries. Also, DeVault et al. (2009) use maximum entropy classification to support responsive overlap in an incremental system by predicting the completions of user utterances. Selfridge et al. (2011) use logistic regression models to predict the stability and accuracy of incremental speech recognition results to enhance performance without causing delay. For related work on (deterministic) incremental language generation, please see (Kilger and Finkler, 1995; Purver and Otsuka, 2003).

Recent years have seen a number of data-driven approaches to interactive systems that automatically adapt their decisions to the dialogue context using Reinforcement Learning (Levin et al., 2000; Walker, 2000; Young, 2000; Singh et al., 2002; Pietquin and Dutoit, 2006; Henderson et al., 2008; Cuayáhuitl et al., 2010; Thomson, 2009; Young et al., 2010; Lemon, 2011; Janarthanam and Lemon, 2010; Rieser et al., 2010; Cuayáhuitl and Dethlefs, 2011; Dethlefs and Cuayáhuitl, 2011). While these approaches have been shown to enhance the performance and adaptivity of interactive systems, unfortunately none of them has yet been combined with incremental processing.

In this paper, we present a novel approach to incremental decision making for output planning that is based on Hierarchical Reinforcement Learning (HRL). In particular, we address the problem of optimising IP strategies while allowing the system to generate and comprehend backchannels and barge-ins based on a partially data-driven reward function. Generating backchannels can be beneficial for grounding in interaction. Similarly, barge-ins can lead to more efficient interactions, e.g. when a system can clarify a bad recognition result immediately before acting based on a misrecognition.

A central concept to our approach is Information Density (ID) (Jaeger, 2010), a psycholinguistic hypothesis that human utterance production is sensitive to a uniform distribution of information across the utterance. This hypothesis has also been adopted for low level output planning recently, see e.g. Rajkumar and White (2011). Our results in terms of average rewards and a human rating study show that a learning agent that is sensitive to ID can learn when it is most beneficial to generate feedback to a user, and outperforms several other agents that are not sensitive to ID.

2 Incremental Information Presentation

2.1 Information Presentation Strategies

Our example domain of application is the Information Presentation phase in an interactive system for restaurant recommendations, extending previous work by Rieser et al. (2010). This previous work incrementally constructs IP strategies according to the predicted user reaction, whereas our approach focuses on whether and when to generate backchannels and barge-ins and how to react to user barge-ins in the context of dynamically changing input hypotheses. We therefore implement a simplified version of Rieser et al.'s model. Their system distinguished two steps: the selection of an IP strategy and the selection of attributes to present to the user. We assume here that the choice of attributes is determined by matching the types specified in the user input, so that our system only needs to choose a strategy for presenting its results. Attributes include *cuisine, food quality, location, price range* and *service quality* of a restaurant. The system then performs a database lookup and chooses among three main IP strategies *summary, comparison, recommendation* and several ordered combinations of these. Please see Rieser et al. (2010) for details. Table 1 shows examples of the main types of IP strategies that we generate.

2.2 Backchannels and Barge-ins

An important advantage of incremental processing can be the increased reactiveness of systems. In this paper, we focus on the phenomena of backchannels and barge-ins that can act as feedback in an interaction for both user and system. Figure 1 shows some examples. *Backchannels* can often be interpreted as signals of grounding. Coming from the user, the system may infer that the user is following the presentation of information or is confirming a piece of information without trying to take the turn. Similarly, we can allow a system to generate backchannels to the user to confirm that it understands the user's preferences, i.e. receives high confidence scores from the ASR module. An important decision for a dialogue system is then *when to generate a backchannel?*

Barge-ins typically occur in different situations. The user may barge-in on the system to correct an ASR error (such as 'Italian' instead of 'Indian' in Figure 1) or the system may want to barge-in on the user to confirm a low-confidence ASR hypothesis so as to be able to start an immediate database look up for results. In the former case, the user barging-in on the system, we assume that the system has two choices: *yielding the turn* to the user, or *trying to keep* the turn. In the latter case, the system barging-in on the user, the system would have to decide *if and when it would be beneficial to barge-in* on a user utterance. In the following sections, we will develop a model of dialogue optimisation that can address these question based on Hierarchical RL that optimises system behaviour based on trade-offs defined in terms of ID.

Type	Example
Comparison	The restaurant *Roma* is in the medium price range, but does not serve excellent food. The restaurants *Firenze* and *Verona* both have great food but are more expensive. The restaurant *Verona* has good service, too.
Recommendation	Restaurant *Verona* has the best overall match with your query. It is a bit more expensive, but has great food and service.
Summary	I found 24 Italian restaurants in the city centre that match your query. 11 of them are in the medium price range, 5 are cheap and 8 are expensive.

Table 1: Examples of IP as a *comparison, recommendation* and *summary* for a user looking for Italian restaurants in the city centre that have a good price for value.

Backchannel 1 (the system backchannels)

USR I want Italian food [500 ms] in the city centre...
SYS uh-huh
SYS OK. I found 24 Italian restaurants in the city centre. The restaurant *Roma* is in the medium price range, but does not have great food. The restaurants *Firenze* and *Verona* ...

Backchannel 2 (the user backchannels)

USR I want Italian food in the centre of town ...
SYS OK. I found 35 central Italian restaurants ...
USR OK.
SYS The restaurant *Verona* has great food but is also a bit expensive. The *Roma* is cheaper, but not as central as *Verona* ...

Barge-ins 1 (the user barges-in on system)

USR I want Italian food in the centre of town ...
SYS I found 35 Indian ...
USR Not Indian, I want Italian.
SYS OK, Italian ...
SYS I have 24 Italian restaurants ...

Barge-ins 2 (the system barges-in on user)

USR I need an Italian restaurant that is located ...
SYS I'm sorry. Did you say Indian or Italian?
USR I said Italian. And in the centre of town please.
SYS OK, let me see. I have 24 Italian restaurants ...

Figure 1: Example phenomena generated with the learnt policy. The agent has learnt to produce backchannels and barge-ins at the appropriate moment and alternative strategies to deal with user barge-ins.

3 Information Theory

Information Theory as introduced by Shannon (1948) is based on two main concepts: a *communication channel* through which information is transferred in bits and the *information gain*, i.e. the information load that each bit carries. For natural language, the assumption is that people aim to communicate according to the channel's capacity, which corresponds to the hearer's capacity in terms of cognitive load. If they go beyond that, the cognitive load of the listener gets too high. If they stay (far) below, too little information is transferred per bit (i.e., the utterance is inefficient or uninformative). The information gain of each word, which is indicative of how close we are to the channel's capacity, can be computed using entropy measures.

3.1 Information Density

Psycholinguistic research has presented evidence for users distributing information across utterances uniformly, so that each word is carrying roughly the same amount of information. This has been observed for phonetic phenomena based on words (Bell et al., 2003) and syllables (Aylett and Turk, 2004), and for syntactic phenomena (Levy and Jaeger, 2007; Jaeger, 2010). Relating ID to likelihood, we can say that the less frequent a word is, the more information it is likely to carry (Jaeger, 2010). For example the word *'the'* often has a high corpus frequency but a low ID.

The ID is defined as the log-probability of an event (i.e. a word) (Shannon, 1948; Levy and Jaeger, 2007), so that for an utterance u consisting of the word sequence $w_1 \ldots w_{i-1}$, we can compute the ID at each point during the utterance as:

$$log\frac{1}{P(u)} = \sum_{i=1}^{n} log\frac{1}{P(w_i|w_1 \ldots w_{i-1})} \qquad (1)$$

While typically the context of a word is given by all preceding words of the utterance, we follow Genzel and Charniak (2002) in restricting our computation to tri-grams for computability reasons. Given a language model of the domain, we can therefore optimise ID in system-generated discourse, where we treat ID as "an optimal solution to the problem of *rapid yet error-free communication* in a noisy environment" (Levy and Jaeger (2007), p.2). We will now transfer the notion of ID to IP and investigate the distribution of information over user restaurant queries.

3.2 Information Density in User Utterances

We aim to use ID for incremental IP in two ways: (1) to estimate the best moment for generating backchannels or barge-ins to the user, and (2) to decide whether to yield or keep the current system turn in case of a user barge-in. While we do not have specific data on human barge-in behaviour, we know from the work of (Jaeger, 2010), e.g., that ID influences human language production. We therefore hypothesise a relationship between ID and incremental phenomena. A human-human data collection is planned for the near future.

To compute the ID of user and system utterances at each time step, we estimated an $n-$gram language model (using Kneser-Ney smoothing) based on a transcribed corpus of human subjects interacting with a system for restaurant recommendations of Rieser et al. (2011).[1] The corpus contained user utterances as exemplified in Figure 1 and allowed us to compute the ID at any point during a user utterance.[2] In this way, we can estimate points of low density which may be eligible for a barge-in or a backchannel. Figure 2 shows some example utterances drawn from the corpus and their ID including the first sentence from Figure 1. These examples were typical for what could generally be observed from the corpus. We see that while information is transmitted with varying amounts of density, the main bits of information are transmitted at a scale between 2 and 7.

Due to a lack of human data for the system utterances, we use the same corpus data to compute the ID of system utterances.[3] The learning agent can use

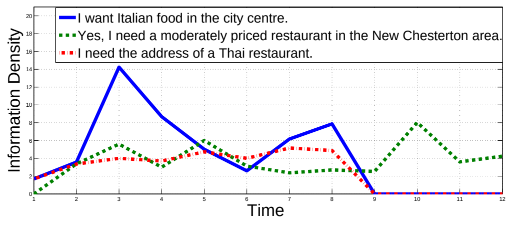

Figure 2: Information Density for example utterances, where peaks indicate places of high density.

this information to consider the trade-off of yielding a current turn to the user or trying to keep it, e.g., in case of a user barge-in given the ID of its own turn and of the user's incoming turn. Such decisions will be made incrementally in our domain given dynamically changing hypotheses of user input.

4 Incremental Utterance Optimisation

To optimise incremental decision making for an interactive system given the optimisation measure of ID, we formalise the dialogue module as a Hierarchical Reinforcement Learning agent and learn an optimal action policy by mapping states to actions and optimising a long-term reward signal. The dialogue states can be seen as representing the system's knowledge about the task, the user and the environment. The dialogue actions correspond to the system's capabilities, such as *present the results* or *barge-in on the user*. They also handle incremental updates in the system. In addition, we need a transition function that specifies the way that actions change the environment (as expressed in the state representation) and a reward function which specifies a numeric value for each action taken. In this way, decision making can be seen as a finite sequence of states, actions and rewards $\{s_0, a_0, r_1, s_1, a_1, ..., r_{t-1}, s_t\}$, where the goal is to induce an optimal strategy automatically using Reinforcement Learning (RL) (Sutton and Barto, 1998).

We used Hierarchical RL, rather than flat RL, because the latter is affected by the *curse of dimensionality*, the fact that the state space grows exponentially according to the state variables taken into account. This affects the scalability of flat RL agents

[1]Available at `http://www.macs.hw.ac.uk/ilabarchive/classicproject/data/login.php`.

[2]Note that our model does not currently handle out-of-domain words. In future work, we will learn when to seek clarification.

[3]We plan a data collection of such utterances for the future, but for now make the assumption that using the corpus data is informative since they are from the same domain.

and limits their application to small-scale problems. Since timing is crucial for incremental approaches, where processing needs to be fast, we choose a hierarchical setting for better scalability. We denote the hierarchy of RL agents as M^i_j where the indexes i and j only identify an agent in a unique way, they do not specify the execution sequence of subtasks, which is subject to optimisation. Each agent of the hierarchy is defined as a Semi-Markov Decision Process (SMDP) consisting of a 4-tuple $< S^i_j, A^i_j, T^i_j, R^i_j >$. Here, S^i_j denotes the set of states, A^i_j denotes the set of actions, and T^i_j is a probabilistic state transition function that determines the next state s' from the current state s and the performed action a. $R^i_j(s', \tau | s, a)$ is a reward function that specifies the reward that an agent receives for taking an action a in state s lasting τ time steps (Dietterich, 1999). Since actions in SMDPs may take a variable number of time steps to complete, the variable τ represents this number of time steps. The organisation of the learning process into discrete time steps allows us to define incremental hypothesis updates as state updates and transitions in an SMDP. Whenever conditions in the learning environment change, such as the recogniser's best hypothesis of the user input, we represent them as transitions from one state to another. At each time step, the agent checks for changes in its state representation and takes the currently best action according to the new state. The best action in an incremental framework can also include generating a *backchannel* to the user to indicate the status of grounding or *barging-in* to confirm an uncertain piece of information. Once information has been presented to the user, it is *committed* or *realised*. Realised information is represented in the agent's state, so that it can monitor its own output.

Actions in a Hierarchical Reinforcement learner can be either primitive or composite. The former are single-step actions that yield single rewards, and the latter are multi-step actions that correspond to SMDPs and yield cumulative rewards. Decision making occurs at any time step of an SMDP: after each single-step action, we check for any updates of the environment that require a system reaction or change of strategy. If no system action is required (e.g. because the user is speaking), the system can decide to do nothing. The goal of each SMDP is to find an optimal policy π^* that maximises the reward for each visited state, according to

$$\pi^{*i}_{j}(s) = \arg\max_{a \in A} Q^{*i}_{j}(s, a), \qquad (2)$$

where $Q^i_j(s, a)$ specifies the expected cumulative reward for executing action a in state s and then following π^*. We use HSMQ-Learning to induce dialogue policies, see (Cuayáhuitl, 2009), p. 92.

5 Experimental Setting

5.1 Hierarchy of Learning Agents

The HRL agent in Figure 3 shows how the tasks of (1) dealing with incrementally changing input hypotheses, (2) choosing a suitable IP strategy and (3) presenting information, are connected. Note that we focus on a detailed description of models $M^1_{0...3}$ here, which deal with barge-ins and backchannels and are the core of this paper. Please see Dethlefs et al. (2012) for details of an RL model that deals with the remaining decisions.

Briefly, model M^0_0 deals with dynamic input hypotheses. It chooses when to listen to an incoming user utterance (M^1_3) and when and how to present information ($M^1_{0...2}$) by calling and passing control to a child subtask. The variable 'incrementalStatus' characterises situations in which a particular (incremental) action is triggered, such as a floor holder *'let me see'*, a correction or self-correction. The variable 'presStrategy' indicates whether a strategy for IP has been chosen or not, and the variable 'userReaction' shows the user's reaction to an IP episode. The 'userSilence' variable indicates whether the user is speaking or not. The detailed state and action space of the agents is given in Figure 4. We distinguish actions for Information Presentation (IP), actions for attribute presentation and ordering (Slot-ordering), and incremental actions (Incremental).

Models $M^1_{0...2}$ correspond to different ways of presenting information to the user. They perform attribute selection and ordering and then call the child agents $M^2_{0...4}$ for attribute realisation. Whenever a user barges in over the system, these agents will decide to either yield the turn to the user or to try and keep the turn based on information density. The variables representing the status of the cuisine,

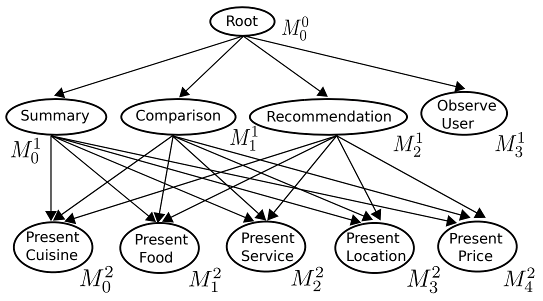

Figure 3: Hierarchy of learning agent for incremental Information Presentation and Slot Ordering.

food, location, price and service of restaurants indicate whether the slot is of interest to the user (we assume that 0 means that the user does not care about this slot), and what input confidence score is currently associated with the value of the slot. For example, if our current best hypothesis is that the user is interested in Indian restaurants, the variable 'statusCuisine' will have a value between 1-3 indicating the strength of this hypothesis. Once slots have been presented to the user, they are *realised* and can only be changed through a correction or self-correction.

Model M_3^1 is called whenever the user is speaking. The system's main choice here is to remain silent and listen to the user or barge-in to request the desired cuisine, location, or price range of a restaurant. This can be beneficial in certain situations, such as when the system is able to increase its confidence for a slot from 'low' to 'high' through barging-in with a direct clarification request, e.g. *'Did you say Indian?'* (and thereby saving several turns that may be based on a wrong hypothesis). This can also be harmful in certain situations, though, assuming that users have a general preference for not being barged-in on. The learning agent will need to learn to distinguish these situations. This agent is also responsible for generating backchannels and will over time learn the best moments to do this.

Models $M_{0...4}^2$ choose surface forms for presentation to the user from hand-crafted templates. They are not the focus of this paper, however, and therefore not presented in detail. The state-action space size of this agent is roughly 1.5 million.[4] The agent

[4]Note that a flat RL agent, in contrast, would need 8×10^{25} million state-actions to represent this problem.

States M_0^0
incrementalStatus {0=none,1=holdFloor,2=correct,3=selfCorrect}
observeUser {0=unfilled,1=filled}
presStrategy {0=unfilled,1=filled}
userReaction {0=none,1=select,2=askMore,3=other}
userSilence={0=false,1=true}
Actions M_0^0
IP: compare M_1^1, recommend M_2^1, summarise M_0^1, summariseCompare, summariseRecommend, summariseCompareRecommend,
Incremental: correct, selfCorrect, holdFloor, observeUser
Goal State M_0^0 0, 1, 1, 0, ?

States $M_{0...2}^1$
IDSystem={0=low,1=medium, 2=high}
statusCuisine {0=unfilled,1=low,2=medium,3=high,4=realised}
statusQuality {0=unfilled,1=low,2=medium,3=high,4=realised}
statusLocation {0=unfilled,1=low,2=medium,3=high,4=realised}
statusPrice {0=unfilled,1=low,2=medium,3=high,4=realised}
statusService {0=unfilled,1=low,2=medium,3=high,4=realised}
turnType {0=holding, 1=resuming, 2=keeping, 3=yielding}
userBargeIn {0=false, 1=true}
Actions $M_{0...2}^1$
Slot-ordering: presentCuisine M_0^2, presentQuality M_1^2, presentLocation M_2^2, presentPrice M_3^2, presentService M_4^2,
Incremental: yieldTurn, keepTurn
Goal State $M_{0...2}^1$?, $\vee$ 4, 0 $\vee$ 4, 0 $\vee$ 4, 0 $\vee$ 4, 0 $\vee$ 4, ?, ?

States M_3^1
bargeInOnUser={0=undecided,1=yes, 2=no}
IDUser={0=low,1=medium, 2=high, 3=falling, 4=rising}
statusCuisine {0=unfilled,1=low,2=medium,3=high,4=realised}
statusLocation {0=unfilled,1=low,2=medium,3=high,4=realised}
statusPrice {0=unfilled,1=low,2=medium,3=high,4=realised}
Actions M_3^1
Incremental: doNotBargeIn, bargeInCuisine, bargeInLocation, bargeInPrice, backchannel
Goal State M_3^1 $>$0, ?, 0 $\vee$ 4, 0 $\vee$ 4, 0 $\vee$ 4

States $M_{0...4}^2$
IDSystem={0=low,1=medium, 2=high}
IDUser={0=low,1=medium, 2=high, 3=falling, 4=rising}
surfaceForm {0=unrealised,1=realised}
Actions $M_{0...4}^2$
Surface Realisation: [alternative surface realisations]
e.g. '$number$ restaurants serve $cuisine$ food', '$number$ places are located in $area$, etc.
Goal State $M_{0...4}^2$?, ?, 1

Figure 4: The state and action space of the HRL agent. The goal state is reached when all items (that the user specified in the search query) have been presented. Question marks mean that a variable does not affect the goal state, which can be reached regardless of the variable's value.

reaches its goal state (defined w.r.t. the state variables in Fig. 4) when an IP strategy has been chosen and all information has been presented.

5.2 The Simulated Environment

For a policy to converge, a learning agent typically needs several thousand interactions in which it is exposed to a multitude of different circumstances. For our domain, we designed a simulated environment with three main components addressing IP, incremental input hypotheses and ID. Using this simulation, we trained the agent for 10 thousand episodes, where one episode corresponds to one recommendation dialogue.

5.2.1 Information Presentation

To learn a good IP strategy, we use a user simulation[5] by Rieser et al. (2010) which was estimated from human data and uses bi-grams of the form $P(a_{u,t}|IP_{s,t})$, where $a_{u,t}$ is the predicted user reaction at time t to the system's IP strategy $IP_{s,t}$ in state s at time t. We distinguish the user reactions of *select* a restaurant, *addMoreInfo* to the current query to constrain the search, and *other*. The last category is usually considered an undesirable user reaction that the system should learn to avoid. The simulation uses linear smoothing to account for unseen situations. In this way, we can predict the most likely user reaction to each system action. Even though previous work has shown that n-gram-based simulations can lead to dialogue inconsistencies, we assume that for the present study this does not present a problem, since we focus on generating single utterances and on obtaining user judgements for single, independent utterances.

5.2.2 Input Hypothesis Updates

While the IP strategies can be used for incremental and non-incremental dialogue, the second part of the simulation deals explicitly with the dynamic environment updates that the system will need to be sensitive to in an incremental setting. We assume that for each restaurant recommendation, the user has the option of filling any or all of the attributes *cuisine, food quality, location, price range* and *service quality*. The possible values of each attribute and possible confidence scores for each value are shown in Table 2. A score of 0 means that the user does not care about the attribute, 1 means that the system's confidence in the attribute's value is low, 2 that the confidence is medium, and 3 means that the confidence is high. A value of 4 means that the attribute has already been *realised*, i.e. communicated to the user. At the beginning of a learning episode, we assign each attribute a possible value and confidence score with equal probability. For food and service quality, we assume that the user is never interested in bad food or service. Subsequently, confidence scores can change at each time step. In future work these transition probabilities will be estimated from a data collection, though the following assumptions are realistic based on our experience. We assume that a confidence score of 0 changes to any other value with a likelihood of 0.05. A confidence score of 1 changes with a probability of 0.3, a confidence score of 2 with a probability of 0.1 and a confidence score of 3 with a probability of 0.03. Once slots have been realised, their value is set to 4. They cannot be changed then without an explicit correction. We also assume that realised slots change with a probability of 0.1. If they change, we assume that half of the time, the user is the origin of the change (because they changed their mind) and half of the time the system is the origin of the change (because of an ASR or interpretation error). Each time a confidence score is changed, it has a probability of 0.5 for also changing its value. The resulting input to the system are data structures of the form *present(cuisine=Indian), confidence=low*. The probability of observing this data structure in our simulation is 0.1 (for Indian) $\times$ 0.2 (for low confidence) $=$ 0.02. Its probability of changing to *present(cuisine=italian), confidence=high* is 0.1 (for changing from low to medium) $\times$ 0.05 (for changing from Indian to Italian) $= 0.005$.

5.2.3 Information Density Updates

We simulate ID of user utterances based on probabilistic context-free grammars (PCFG) that were automatically induced from the corpus data in Section 3.2 using the ABL algorithm (van Zaanen, 2000). This algorithm takes a set of strings as input and computes a context-free grammar as output by aligning strings based on Minimum Edit Distance. We use the $n-$gram language models trained earlier to

[5]The simulation data are available from `www.classic-project.org`.

Attribute	Values	Confidence
Cuisine	Chinese, French, German, Indian, Italian, Japanese, Mexican, Scottish, Spanish, Thai	0, 1, 2, 3, 4
Quality	bad, adequate, good, very good	0, 1, 2, 3, 4
Location	7 distinct areas of the city	0, 1, 2, 3, 4
Price	cheap, good-price-for-value, expensive, very expensive	0, 1, 2, 3, 4
Service	bad, adequate, good, very good	0, 1, 2, 3, 4

Table 2: User goal slots for restaurant queries with possible values and confidence scores.

add probabilities to grammar rules. We use these PCFGs to simulate user utterances to which the system has to react. They can be meaningful utterances such as *'Show me restaurants nearby'* or less meaningful fragments such as *'um let me see, do you... hm'*. The former type is more frequent in the data, but both types can be simulated along with their ID (clearly, the first type is more dense than the second).

In addition to simulating user utterances, we hand-crafted context-free grammars of system utterances and augmented them with probabilities estimated using the same user corpus data as above (where again, we make the assumption that this is to some extent feasible given the shared domain). We use the simulated system utterances to compute varying degrees of ID for the system.

Both measures, the ID of user and system utterances, can inform the system during learning to balance the trade-off between them for generating and receiving backchannels and barge-ins.

5.3 A Reward Function for Incremental Dialogue Based on Information Density

To train the HRL agent, we use a partially data-driven reward function. For incremental IP, we use rewards that are based on human intuition. The agent receives

$$R = \begin{cases} +100 & \text{if the user selects an item,} \\ 0 & \text{if the user adds further constraints to the search,} \\ -100 & \text{if the user does something else or a self-correction,} \\ -0.5 & \text{for the system holding a turn,} \\ -1 & \text{otherwise.} \end{cases}$$

The agent is encouraged to choose those sequences of actions that lead to the user selecting a restaurant as quickly as possible. If the agent is not sure what to say (because planning has not finished), it can generate a floor holding marker, but should in any case avoid a self-correction due to having started speaking too early.

The remaining rewards are based on ID scores computed incrementally during an interaction. The agent receives the following rewards, where infoDensity(Usr) and infoDensity(Sys) refer to the ID of the current user and system utterance, respectively, as defined in Equation 1.

$$R = \begin{cases} \text{-infoDensity(Usr)} & \text{for keeping a turn, barging-in or a backchannel,} \\ \text{-infoDensity(Sys)} & \text{for yielding a turn.} \end{cases}$$

These two measures encourage the agent to consider the trade-offs between its own ID and the one transmitted by an incoming user utterance. Barging-in on a user utterance at a low ID point then yields a small negative reward, whereas barging-in on a user utterance at a high ID point yields a high negative reward. Both rewards are negative because barging-in on the user always contains some risk. Similarly, keeping a turn over a non-dense user utterance receives a smaller negative reward than keeping it over a dense user utterance. A reward of -2 is assigned for barging-in over a user utterance fragment with a falling ID to reflect results from a qualitative study of our corpus data: humans tend to barge-in *between* information peaks, so that a barge-in to clarify a low-confidence slot appears immediately before the ID is rising again for a new slot. The exact best moment for barge-ins and backchannels to occur will be subject to optimisation.

6 Experimental Results

The agent learns to barge-in or generate backchannels to users at points where the ID is low but rising. In particular, the agent learns to barge-in *right before* information density peaks in an incoming user utterance to clarify or request slots that are still open from the previous information density peak. If a user has specified their desired cuisine type but the system has received a low ASR confidence score for it, it may barge-in to clarify the slot. This case was illustrated in the last example in Figure 1, where the system clarified the previous (cuisine) slot (which is associated with a high ID) just before the user specifies the location slot (which again would have a high ID). The main benefit the system can gain through clarification barge-ins is to avoid self-corrections when having acted based on a low ASR confidence, leading to more efficient interactions.

The system learns to generate backchannels *after* information peaks to confirm newly acquired slots that have a high confidence. An example is shown in the first dialogue fragment in Figure 1.

In addition, the system learns to yield its current turn to a user that is barging-in if its own ID is low, falling or rising, or if the ID of the incoming user utterance is high. If the system's own ID is high, but the user's is not, it will try to keep the turn.[6] This is exemplified in the third dialogue fragment in Figure 1.

We compare our learnt policy against two baselines. **Baseline 1** was designed to always generate barge-ins *after* an information peak in a user utterance, i.e. when ID has just switched from *high* to *falling*. We chose this baseline to confirm that users indeed prefer barge-ins before information peaks rather than at any point of low ID. Baseline 1 yields a turn to a user barge-in if its own ID is low and tries to keep it otherwise. **Baseline 2** generates barge-ins and backchannels randomly and at any point during a user utterance. The decision of yielding or keeping a turn in case of a user barge-in is also random. Both baselines also use HRL to optimise their IP strategy. We do not compare different IP strategies, which has been done in detail by Rieser et al. (2010). All results are summarised in Table 3.

[6]Incidentally, this also helps to prevent the system yielding its turn to a user backchannel; cf. Example 2 in Fig. 1.

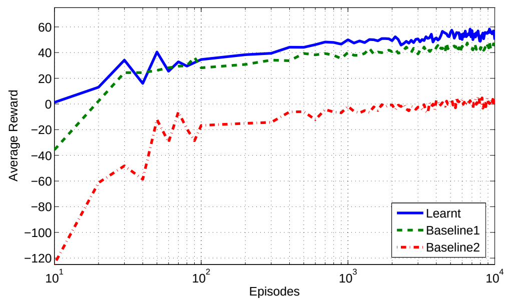

Figure 5: Performance in terms of rewards (averaged over 10 runs) for the HRL agent and its baselines.

6.1 Average Rewards over Time

Figure 5 shows the performance of all systems in terms of average rewards in simulation. The learnt policy outperforms both baselines. While the learnt policy and Baseline 1 appear to achieve similar performance, an absolute comparison of the last 1000 episodes of each behaviour shows that the improvement of the HRL agent over Baseline 1 corresponds to 23.42%. The difference between the learnt policy and its baselines is significant at $p < 0.0001$ according to a paired t-test and has a high effect size of $r = 0.85$.

The main reason for these different performances is the moment each system will barge-in. Since Baseline 1 barges-in on users after an information peak, when ID may still be high, it continuously receives a negative reward reflecting the user preference for late barge-ins. As a result of this continuous negative reward, the agent will then learn to avoid barge-ins altogether, which may in turn lead to less efficient interactions because low confidence ASR scores are clarified only late in the interaction.

The main problem of the random barge-ins of Baseline 2 is that users may often have to restart a turn because the system barged-in too early or in the middle of an information peak. In addition, Baseline 2 needs to occasionally self-correct its own utterances because it started to present information too early, when input hypotheses were not yet stable enough to act upon them.

Policy	Average Reward	User Rating (%)
Learnt	$55.54^{**,*}$	$43\%^{**}$
Baseline 1	45.0^{**}	26%
Baseline 2	1.47	31%

Table 3: Comparison of policies in terms of average rewards and user ratings. * indicates a significant improvement over Baseline 1 and ** over Baseline 2.

6.2 Human Rating Study

To confirm our simulation-based results, we conducted a user rating study on the CrowdFlower crowd sourcing platform.[7] Participants were shown user utterances along with three options of barging-in over them. For example: `| I want [OPTION 1] Italian food [OPTION 2] in the city [OPTION 3] centre|`, where `OPTION 1` corresponds to the learnt policy, `OPTION 2` to Baseline 2 and `OPTION 3` to Baseline 1.

Users were asked to choose one option which they considered the best moment for a barge-in. Participants in the study rated altogether 144 utterances. They preferred the *learnt* system 63 times (43%), Baseline 1 37 times (26%) and Baseline 2 44 times (31%). This is statistically significant at $p < 0.02$ according to a Chi-Square test ($\chi^2 = 7.542$, df = 2). In a separate test, directly comparing the *learnt* policy and Baseline 1, *learnt* was chosen significantly more often than Baseline 1; i.e. 79% of the time (for 127 utterances, using a 1-tailed Sign test, $p < 0.0001$). Finally, *learnt* was directly compared to Baseline 2 and shown to be significantly more often chosen; i.e. 59% of the time (138 utterances, 1-tailed Sign test, $p < 0.025$). These results provide evidence that an optimisation of the timing of generating barge-ins and backchannels in incremental dialogue can be sensitive to fine-grained cues in evolving ID and therefore achieve a high level of adaptivity. Such sensitivity is difficult to hand-craft as can be concluded w.r.t. the performance of Baseline 1, which received similar rewards to *learnt* in simulation, but is surprisingly beaten by the random Baseline 2 here. This indicates a strong human dislike for late barge-ins. The bad performance of Baseline 2 in terms of average rewards was due to the random barge-ins leading to less efficient dialogues.

[7] `www.crowdflower.com`

Regarding user ratings however, Baseline 2 was preferred over Baseline 1. This is most likely due to the timing of barge-ins: since Baseline 2 has a chance of barging-in at earlier occasions than Baseline 1, it may have received better ratings. The evaluation shows that humans care about timing of a barge-in regarding the density of information that is currently conveyed and dislike late barge-ins. ID is then useful in determining when to barge-in. We can therefore further conclude that ID can be a feasible optimisation criterion for incremental decision making.

7 Conclusion and Future Work

We have presented a novel approach to incremental dialogue decision making based on *Hierarchical RL* combined with the notion of *information density*. We presented a learning agent in the domain of IP for restaurant recommendations that was able to generate backchannels and barge-ins for higher responsiveness in interaction. Results in terms of average rewards and a human rating study have shown that a learning agent that is optimised based on a partially *data-driven reward function* that addresses information density can learn to decide when and if it is beneficial to barge-in or backchannel on user utterances and to deal with backchannels and barge-ins from the user. Future work can take several directions. Given that ID is a measure influencing human language production, we could replace our template-based surface realiser by an agent that optimises the information density of its output. Currently we learn the agent's behaviour offline, before the interaction, and then execute it statistically. More adaptivity towards individual users and situations could be achieved if the agent was able to learn from ongoing interactions. Finally, we can confirm the human results obtained from an overhearer-style evaluation in a real interactive setting and explicitly extend our language model to discourse phenomena such as pauses or hesitations to take them into account in measuring ID.

Acknowledgements

The research leading to this work has received funding from EC's FP7 programmes: (FP7/2011-14) under grant agreement no. 287615 (PARLANCE); (FP7/2007-13) under grant agreement no. 216594 (CLASSiC); (FP7/2011-14) under grant agreement no. 270019 (SPACEBOOK);

and (FP7/2011-16) under grant agreement no. 269427 (STAC). Many thanks to Michael White for discussion of the original idea of using information density as an optimisation metric.

References

Matthew Aylett and Alice Turk. 2004. The smooth signal redundancy hypothesis: A functional explanation for the relationships between redundancy, prosodic prominence, and duration in spontaneous speech. *Language and Speech*, 47(1):31–56.

Timo Baumann, Okko Buss, and David Schlangen. 2011. Evaluation and Optimisation of Incremental Processors. *Dialogue and Discourse*, 2(1).

Alan Bell, Dan Jurafsky, Eric Fossler-Lussier, Cynthia Girand, Michelle Gregory, and Daniel Gildea. 2003. Effects of disfluencies, predictability, and utterance position on word form variation in english conversation. *Journal of the Acoustic Society of America*, 113(2):1001–1024.

Okko Buss, Timo Baumann, and David Schlangen. 2010. Collaborating on Utterances with a Spoken Dialogue Systen Using an ISU-based Approach to Incremental Dialogue Management. In *Proceedings of 11th Annual SIGdial Meeting on Discourse and Dialogue*.

Heriberto Cuayáhuitl and Nina Dethlefs. 2011. Spatially-aware Dialogue Control Using Hierarchical Reinforcement Learning. *ACM Transactions on Speech and Language Processing (Special Issue on Machine Learning for Robust and Adaptive Spoken Dialogue System)*, 7(3).

Heriberto Cuayáhuitl, Steve Renals, Oliver Lemon, and Hiroshi Shimodaira. 2010. Evaluation of a hierarchical reinforcement learning spoken dialogue system. *Computer Speech and Language*, 24(2):395–429.

Heriberto Cuayáhuitl. 2009. *Hierarchical Reinforcement Learning for Spoken Dialogue Systems*. PhD Thesis, University of Edinburgh, School of Informatics.

Nina Dethlefs and Heriberto Cuayáhuitl. 2011. Combining Hierarchical Reinforcement Learning and Bayesian Networks for Natural Language Generation in Situated Dialogue. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, Nancy, France.

Nina Dethlefs, Helen Hastie, Verena Rieser, and Oliver Lemon. 2012. Optimising Incremental Generation for Spoken Dialogue Systems: Reducing the Need for Fillers. In *Proceedings of the International Conference on Natural Language Generation (INLG)*, Chicago, Illinois, USA.

David DeVault, Kenji Sagae, and David Traum. 2009. Can I finish? Learning when to respond to incremental interpretation result in interactive dialogue. In *Proceedings of the 10th Annual SigDial Meeting on Discourse and Dialogue*, Queen Mary University, UK.

Thomas G. Dietterich. 1999. Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition. *Journal of Artificial Intelligence Research*, 13:227–303.

Dmitriy Genzel and Eugene Charniak. 2002. Entropy Rate Constancy in Text. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 199–206.

James Henderson, Oliver Lemon, and Kallirroi Georgila. 2008. Hybrid Reinforcement/Supervised Learning of Dialogue Policies from Fixed Data Sets. *Computational Linguistics*, 34(4):487–511.

T. Florian Jaeger. 2010. Redundancy and reduction: Speakers manage syntactic information density. *Cognitive Psychology*, 61:23–62.

Srini Janarthanam and Oliver Lemon. 2010. Learning to Adapt to Unknown Users: Referring Expression Generation in Spoken Dialogue Systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 69–78, July.

Anne Kilger and Wolfgang Finkler. 1995. Incremental generation for real-time applications. Technical report, DFKI Saarbruecken, Germany.

Oliver Lemon. 2011. Learning What to Say and How to Say It: Joint Optimization of Spoken Dialogue Management and Natural Language Generation.

Esther Levin, Roberto Pieraccini, and Wieland Eckert. 2000. A Stochastic Model of Computer-Human Interaction for Learning Dialogue Strategies. *IEEE Transactions on Speech and Audio Processing*, 8:11–23.

Roger Levy and T. Florian Jaeger. 2007. Speakers optimize information density through syntactic reduction. *Advances in Neural Information Processing Systems*, 19.

Olivier Pietquin and Dutoit. 2006. A Probabilistic Framework for Dialogue Simulation and Optimal Strategy Learning. *IEEE Transactions on Speech and Audio Processing*, 14(2):589–599.

Matthew Purver and Masayuki Otsuka. 2003. Incremental Generation by Incremental Parsing. In *Proceedings of the 6th UK Special-Interesting Group for Computational Linguistics (CLUK) Colloquium*.

Rajakrishnan Rajkumar and Michael White. 2011. Linguistically Motivated Complementizer Choice in Surface Realization. In *Proceedings of the EMNLP-11 Workshop on Using Corpora in NLG*, Edinburgh, Scotland.

Antoine Raux and Maxine Eskenazi. 2009. A Finite-State Turn-Taking Model for Spoken Dialog Systems. In *Proceedings of the 10th Conference of the*

North American Chapter of the Association for Computational Linguistics—Human Language Technologies (NAACL-HLT), Boulder, Colorado.

Verena Rieser, Oliver Lemon, and Xingkun Liu. 2010. Optimising Information Presentation for Spoken Dialogue Systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, Uppsala, Sweden.

Verena Rieser, Simon Keizer, Xingkun Liu, and Oliver Lemon. 2011. Adaptive Information Presentation for Spoken Dialogue Systems: Evaluation with Human Subjects. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, Nancy, France.

David Schlangen and Gabriel Skantze. 2011. A General, Abstract Model of Incremental Dialogue Processing. *Dialogue and Discourse*, 2(1).

Ethan Selfridge, Iker Arizmendi, Peter Heeman, and Jason Williams. 2011. Stability and Accuracy in Incremental Speech Recognition. In *Proceedings of the 12th Annual SigDial Meeting on Discourse and Dialogue,* Portland, Oregon.

Claude Shannon. 1948. A Mathematical Theory of Communications. *Bell Systems Technical Journal,* 27(4):623–656.

Satinder Singh, Diane Litman, Michael Kearns, and Marilyn Walker. 2002. Optimizing Dialogue Management with Reinforcement Learning: Experiments with the NJFun System. *Journal of Artificial Intelligence Research*, 16:105–133.

Gabriel Skantze and Anna Hjalmarsson. 2010. Towards Incremental Speech Generation in Dialogue Systems. In *Proceedings of the 11th Annual SigDial Meeting on Discourse and Dialogue*, Tokyo, Japan.

Richard Sutton and Andrew Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

Blaise Thomson. 2009. *Statistical Methods for Spoken Dialogue Management*. Ph.D. thesis, University of Cambridge.

Menno van Zaanen. 2000. Bootstrapping Syntax and Recursion using Alignment-Based Learning. In *Proceedings of the Seventeenth International Conference on Machine Learning,* ICML ’00, pages 1063–1070.

Marilyn Walker. 2000. An Application of Reinforcement Learning to Dialogue Strategy Selection in a Spoken Dialogue System for Email. *Journal of Artificial Intelligence Research (JAIR)*, 12:387–416.

Steve Young, Milica Gasic, Simon Keizer, Francois Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The Hidden Information State Model: A Practical Framework for POMDP-based Spoken Dialogue Management. *Computer Speech and Language*, 24(2):150–174.

Steve Young. 2000. Probabilistic Methods in Spoken Dialogue Systems. *Philosophical Transactions of the Royal Society (Series A)*, 358(1769):1389–1402.