CLASSiC

D4.2: Optimal Wizard NLG Behaviours in Context

Verena Rieser, Xingkun Liu, Oliver Lemon

Distribution: Public

CLASSiC
Computational Learning in Adaptive Systems for Spoken Conversation
216594 Deliverable 4.2

October 2009

Project funded by the European Community under the Seventh Framework Programme for Research and Technological Development

The deliverable identification sheet is to be found on the reverse of this page.

Project ref. no.	216594
Project acronym	CLASSiC
Project full title	Computational Learning in Adaptive Systems for Spoken Conversation
Instrument	STREP
Thematic Priority	Cognitive Systems, Interaction, and Robotics
Start date / duration	01 March 2008 / 36 Months

Security	Public
Contractual date of delivery	M18 = August 2009
Actual date of delivery	October 2009
Deliverable number	4.2
Deliverable title	D4.2: Optimal Wizard NLG Behaviours in Context
Type	Report
Status & version	Final 1.0
Number of pages	63 (excluding front matter)
Contributing WP	4
WP/Task responsible	WP4, leader UEDIN
Other contributors	Paul Crook
Author(s)	Verena Rieser, Xingkun Liu, Oliver Lemon
EC Project Officer	Philippe Gelin
Keywords	Natural Language Generation, Wizard-of-Oz, corpus, supervised learning, human dialogue behaviour, Information Presentation

The partners in CLASSiC are:		
	Heriot-Watt University	HWU
	University of Cambridge	UCAM
	University of Geneva	GENE
	Ecole Superieure d'Electricite	SUPELEC
	France Telecom/ Orange Labs	FT
	University of Edinburgh HCRC	EDIN

For copies of reports, updates on project activities and other CLASSIC-related information, contact:

The CLASSIC Project Co-ordinator:
Dr. Oliver Lemon
School of Mathematical and Computer Sciences (MACS)
Heriot-Watt University
Edinburgh
EH14 4AS
United Kingdom
O.Lemon@hw.ac.uk
Phone +44 (131) 451 3782 - Fax +44 (0)131 451 3327

Copies of reports and other material can also be accessed via the project's administration homepage, http://www.classic-project.org

Contents

Executive summary

This report describes a method for learning Natural Language Generation (NLG) behaviours from data collected from humans in a "Wizard of Oz" (WoZ) study. In a WoZ study a human operator, also known as a "wizard", performs some of the tasks of a spoken dialogue system (SDS), e.g. Spoken Language Understanding and Dialogue Management, while the user is left in the belief that s/he is interacting with a real system. Our experiment is novel in the sense that it aims to investigate the NLG decisions of the human wizards.

In particular, we are interested in the wizards' Information Presentation (IP) decisions in the presence of uncertainty, following the framework of NLG as "planning under uncertainty" [Rieser and Lemon, 2009b, Lemon, 2008].

We use statistical learning techniques in order to describe the wizards' NLG behaviour in context, following the general framework of [Rieser and Lemon, 2009a]. We show that the dialogue context influences the wizards' decisions and how highly rated NLG strategies differ to low-rated NLG strategies with respect to the context they are applied in. We present a model for learning NLG strategy choice and attribute selection in a hierarchical manner using Supervised Learning. The models trained for NLG strategy selection significantly ($p = .001$) outperform a majority baseline, and the models trained for attribute selection perform as well as the majority baseline.

These models reflect the average human wizard NLG behaviour. We learn those models for the data subset which was rated highly by the users, reflecting "optimal" wizard behaviour, and compare them against models learned on the low-rated part of the data. We show that good NLG strategies are context-adaptive: wizards adapt to different context features and follow different strategies for highly-rated utterances than in the case of the low-rated NLG behaviour. We discuss these differences for NLG Information Presentation strategy selection and attribute selection.

Furthermore, feature ranking and selection techniques provide evidence that wizards' NLG strategy and attribute selection can be modelled as a sequential hierarchical decision process, where NLG information is shared over several turns, and higher level NLG actions feed into the decision model for lower level NLG actions.

These NLG models will later serve as a baseline for Reinforcement Learning-based NLG strategies to be "bootstrapped" from this data following [Rieser and Lemon, 2008].

Data-collection aspects of this work are described in deliverable D6.1, and the corpus collected will be released publicly as part of the project's Annotated Data Archive, in deliverable D6.5.

(Some aspects of this work were published at EACL 2009, [Rieser and Lemon, 2009b] and at the First International Workshop on Spoken Dialogue Systems [Liu et al., 2009], and have been submitted as an invited book chapter [Rieser and Lemon, subm].)

1 Introduction

This report describes a method for learning Natural Language Generation (NLG) behaviours from data collected from humans in a "Wizard of Oz" study, following the general framework of [Rieser and Lemon, 2009a]. NLG is concerned with mapping from non-linguistic to linguistic expressions, e.g. [Reiter and Dale, 2000]. NLG for Spoken Dialogue Systems (SDSs) converts Speech Acts from the Dialogue Manager (DM) into spoken prompts. Information Presentation (IP) is a central aspect of Natural Language Generation (NLG) for Spoken Dialogue Systems (SDS). Information presentation strategies are one of the main contributors to dialogue duration and are positively correlated with task success and user satisfaction [Walker et al., 2001]. In this report we learn context dependent, human "wizard" NLG behaviours for IP in the TownInfo domain.

We therefore collect data in a Wizard-of-Oz (WoZ) experiment [Dahlbäck et al., 1993], [Fraser and Gilbert, 1991]. In a WoZ study a human operator, also known as "wizard", performs some of the tasks of a SDS, e.g. Spoken Language Understanding and Dialogue Management, while the user is left in the belief that he is interacting with a real system. Previous studies have also used the WoZ technique to study user reactions to NLG policies, e.g. [Whittaker et al., 2002, Winterboer and Moore, 2007, Janarthanam and Lemon, 2009]. Our experiment is novel in the sense that it aims to investigate the behaviour of the human wizards, as well as the preferences of the user, following [Rieser and Lemon, 2009a]. In other words, this study has 2 experimental subjects: the wizard and the user.

In particular, we are interested in the wizards' IP policy decisions in the presence of uncertainty, following the framework of NLG as "planning under uncertainty" [Rieser and Lemon, 2009b, Lemon, 2008]. For example, the wizards' decisions might depend on the number of database items retrieved, a noisy user query, and a stochastic NLG surface realiser.

In the following we shortly summarise our approach to NLG for SDS.

1.1 NLG as planning under uncertainty

We follow the overall framework of NLG as planning under uncertainty [Rieser and Lemon, 2009b, Lemon, 2008], where each NLG action is a sequential decision problem, based on the current (uncertain) dialogue state and the expected utility/"reward" of the interaction. Other recent approaches describe this task as planning, e.g. [Koller and Stone, 2007, Koller and Petrick, 2008], or as contextual decision making according to a utility/ cost function [van Deemter, 2009].

We apply this framework to Information Presentation strategies in SDS, where the example task is to present a set of restaurants to the user.

However, we believe that the presented framework can be applied to many other domains which require complex information to be conveyed to the user, e.g. instruction giving and tutorial dialogue, or other types of Information Presentation tasks, such as sales agents, tourist or health information systems.

We consider 7 possible policies (see Figure 1): Recommending one single restaurant (RECOMMEND), compare two (COMPARE), summarise all (SUMMARY), or an ordered combination of those, e.g. first summarise all the retrieved restaurants and then recommend one. The wizard has to decide which NLG action to take next (RECOMMEND, COMPARE, SUMMARY), which attributes to mention (cuisine, price range, location, food quality, service quality), or whether to stop generating. In the WoZ study, we want the wizards to explore these actions in different dialogue contexts, in order to use this data for RL-based strategy development, following [Rieser and Lemon, 2008].

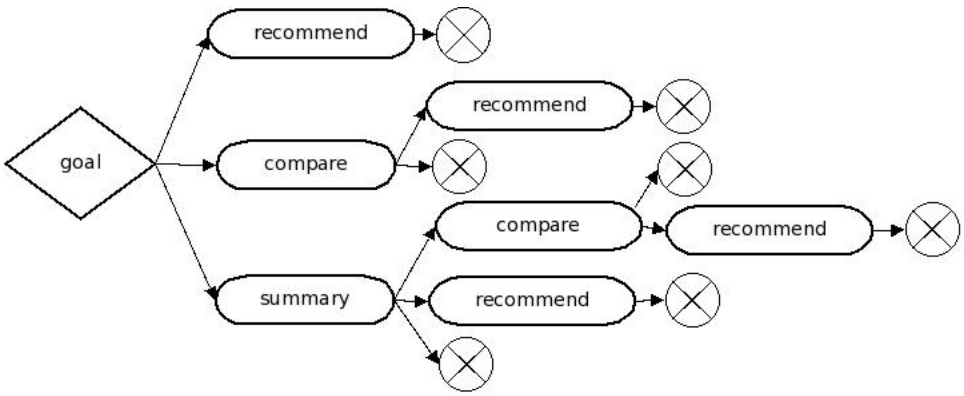


Figure 1: Possible NLG policies (X=stop generation)

The report proceeds as follows. Section 2 describes the WOZ data collection environment, the experimental setup, data collected, and the annotated features. In Section 3 we present an analysis of the corpus in terms of observed IP strategies, their performance, and differences between wizards and noise conditions. Section 4 presents a method for for learning optimal wizard NLG strategies in context. Section 5 applies the same method to attribute selection. In section 6 we discuss the obtained models and results.

2 WoZ data collection

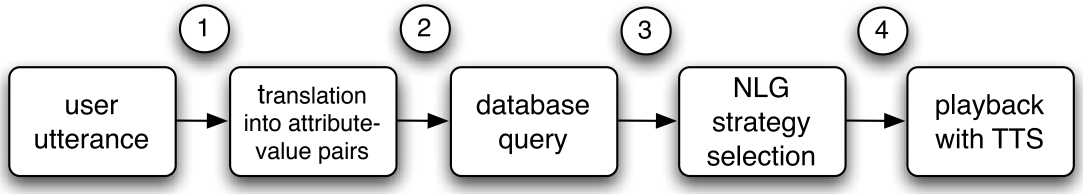


Figure 2: Overview of the WoZ setup

Our WoZ setup involves 4 stages (see Figure 2): First, the user's utterance is tagged with attribute values by the wizard. For some of the cases artificial ASR noise is introduced (see Section 2.1.2). In the noise condition it is the experimenter who listens to the user's utterance and does the tagging. For the no-noise condition the wizard directly listens to the user's utterance and translates into attribute values. For example, "I am looking for Indian restaurants in the Old Town" gets tagged as `cuisine=Indian, location=Old Town`. The wizard then queries the database, where we use a real database of Edinburgh restaurants provided by TheList, [1] and selects an NLG strategy.

There were no general time constraints on the wizards. However, wizards were encouraged to act as quickly as possible.

The strategy then gets generated by a surface realiser (see Section 2.1.1). The final utterance is played back to the user via TTS, using the Cereproc speech synthesiser. [2] We manually transcribed restaurant names (especially those with international names) in order to improve the TTS quality for proper names.

2.1 WoZ environment

Figure 3 shows the web-based interface for the wizard. The experimenter, the user, and the experimenter have similar interfaces, which communicate with the wizard's page using a web-based server-client architecture. The audio is transmitted and recorded using VOIP [3]. The wizard GUI contains 5 main panels (see Figure 3):

A: The wizard receives the user's query as noisy attribute values from the noise model. The experimenter has a similar input panel. There are 5 searchable attributes in total, which can also be negative ("not expensive").

[1] http://www.list.co.uk/

[2] http://www.cereproc.com/

[3] Skype, `http://www.skype.com/`

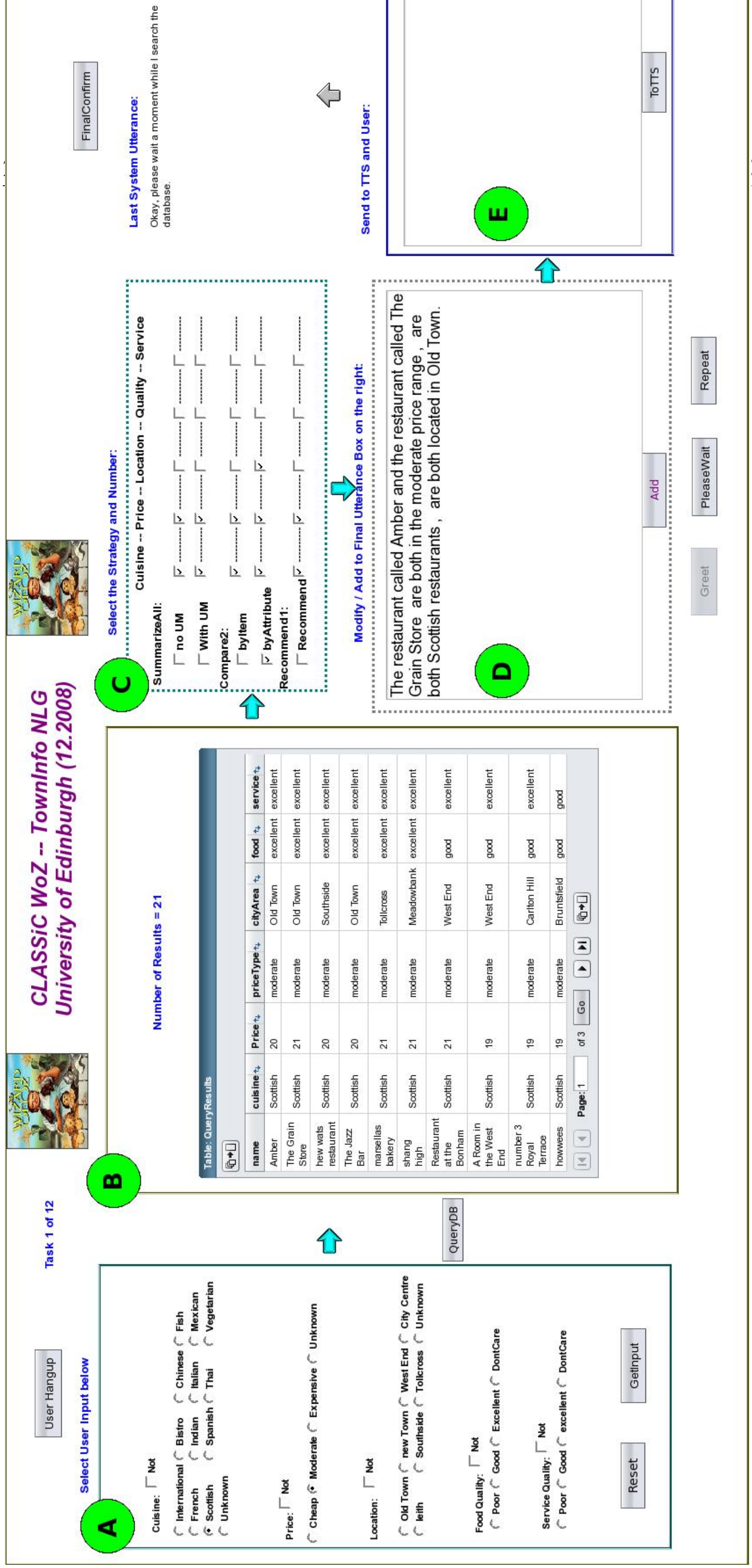
B: The retrieved database items are presented in an ordered list. We use a User Modelling approach for ranking the restaurants, see e.g. [Polifroni and Walker, 2008], where we assume that a default user cares about cheap food with high quality and good service.

C: The wizard then chooses which strategy and which attributes to generate next, by clicking radio buttons. The attribute/s specified in the last user query are pre-selected by default. The strategies can only be combined in the orders as specified in Figure 1.

D: An utterance is automatically generated by the NLG surface realiser every time the wizard selects a strategy, and is displayed in an intermediate text panel.

E: The wizard can decide to add the generated utterance to the final output panel. The text in the final panel is sent to the user via TTS, once the wizard decides to stop generating.

(Please see figure 3 on the next page.)



name	cuisine	Price	priceType	cityArea	food	service
Amber	Scottish	20	moderate	Old Town	excellent	excellent
The Grain Store	Scottish	21	moderate	Old Town	excellent	excellent
new wats restaurant	Scottish	20	moderate	Southside	excellent	excellent
The Jazz Bar	Scottish	20	moderate	Old Town	excellent	excellent
marsellas bakery	Scottish	21	moderate	Tollcross	excellent	excellent
shang high	Scottish	21	moderate	Meadowbank	excellent	excellent
Restaurant at the Bonham	Scottish	21	moderate	West End	good	excellent
A Room in the West End	Scottish	19	moderate	West End	good	excellent
number 3 Royal Terrace	Scottish	19	moderate	Carlton Hill	good	excellent
howwees	Scottish	19	moderate	Bruntsfield	good	good

Page: 1 of 3 Go

Figure 3: Wizard interface

2.1.1 NLG prompt generation

We have implemented an NLG surface realiser in order to generate IP strategies in real time. This generator is based on data from a stochastic sentence planner called SPaRKy [Stent et al., 2004]. We replicate the variation observed in SPaRKy by analyzing high-ranking example outputs (given the highest possible score by both SPaRKy judges) and implement the variance in dynamic templates. The realisations vary in sentence aggregation, aggregation operators (e.g. 'and', 'full stop', or ellipsis), contrasts (e.g. 'however', 'on the other hand') and referring expressions (e.g. 'it', 'this restaurant') used. In particular, we consider the following realisations of Information Presentation actions (see examples in Table 1):

- RECOMMEND the top-ranking restaurant (according to UM).
- COMPARE the top 2 restaurants by Item or by Attribute.
- SUMMARY of all matching restaurants with or without User Model (UM), following [Polifroni and Walker, 2008]. The approach using a UM assumes that the user has certain preferences (e.g. cheap) and only tells him about the relevant options, whereas the approach with no UM lists all the options.

strategy	example utterance
SUMMARY no UM	I found 26 restaurants, which have Indian cuisine. Furthermore, 11 of the restaurants are in the expensive price range. However, 10 of the restaurants are in the cheap price range and 5 of the restaurants are in the moderate price range.
SUMMARY UM	26 restaurants meet your query. There are 10 restaurants which serve Indian food and are in the cheap price range. There are also 16 others which are more expensive.
COMPARE by Item	The restaurant called Kebab Mahal is an Indian restaurant. It is in the cheap price range. And the restaurant called Saffrani, which is also an Indian restaurant, is also in the cheap price range.
COMPARE by Attr	The restaurant called Kebab Mahal and the restaurant called Saffrani are both in the cheap price range , and are both Indian restaurants.
RECOMMEND	The restaurant called Kebab Mahal has the best overall quality amongst the matching restaurants. It is an Indian restaurant , and it is in the cheap price range.

Table 1: Example NLG actions generated for user provided `cuisine=Indian`, and the wizard also selects the additional attribute `price`.

2.1.2 Noise modelling

We also introduce artificial noise in the WoZ setup, replicating non- and misunderstanding errors as introduced by ASR. In contrast to previous work, e.g. [Rieser et al., 2005, Stuttle et al., 2004], we simulate noise on the semantic level, modelling deletions, substitutions, and insertions for the attribute-value pairs. Our noise model introduces noise for 36.3% of the attributes provided by the user, where 61.7% are deleted, 26.4% are replaced, and 11.8% are inserted, according to an analysis of data gathered by [Lemon et al., 2006]. A similar noise model has also been used by [Crook and Lemon, 2009].

We are interested in the kind of effects the input noise has on the wizard's decisions, and also whether the user prefers different IP strategies in the presence of noise. For example, the user might prefer shorter system prompts if there was some mis-recognition.

2.2 Experimental setup and Data collected

2.2.1 Experimental setup

We use a fixed, hand-coded dialogue management strategy, since we are only interested in IP strategies generated by our NLG module. In particular, we hand-code dialogue behaviour such as greeting the user in the beginning, triggering wait messages when the database is searched by the wizard, informing the user if no items are found, and final messages (see Example 1). The wizard can also directly answer user requests by typing messages into the final TTS panel (**E**) and sending them to the user.

The user can either provide more constraints, select a restaurant, ask a question, e.g. request more information, ask to repeat the last system utterance, or hang up.

(1) **WoZ:** Welcome to the Edinburgh restaurant system. How may I help you today? (**greeting**)

User: I am looking for a French restaurant which is not in the City Centre. (**provideInfo**)

WoZ: Okay, please wait a moment while I search the database. (**wait message**)

WoZ: I found 16 restaurants , which are not located in City Centre, and which have French cuisine. The restaurant called La Garrigue is in the cheap price range. It is located in Old Town. The restaurant called Jacques is in the cheap price range and it is located in Tollcross. Do you want to choose this one? (**IP**)

User: What is the name again? (**RequestMoreInfo**)

WoZ: La Garrigue (**typed message**)

User: Ok, great. I'll take it. (**select**)

WoZ: Please click the Finish button and then take some time to fill out the questionnaire. (**final message**)

We collect data from 2 different wizards (w1,w2) on this task. Both are native speakers of English and have some background in computer science. The wizard and the users are located in two different rooms.

Task sets: We designed 10 task sets. Each task set contains 12 tasks of 3 different types. Type I tasks specified constraints so that 2-4 database hits were returned. Type II tasks define a set of 5-13 database hits, and Type III tasks 15-46. We designed different task types in order to expose the wizards to sets of different sizes to present to the user. Task types are arranged into different orders for every task set. Furthermore, the task description included some vagueness about the values, in order not to prime the user to use certain words. The following task description, for example, can be translated into attribute-value pairs as {`location=Leith`}, {`price= moderate or cheap`}.

> *You're on a visit to Leith and it's time for dinner. You're open to any type of food, as long as it's not expensive.*

Each user has to perform a total of 12 tasks, where no task set is seen twice by one wizard. The users are given general instructions and an agreement form in the beginning of each session. We also ask the users to provide us some background information on age group, profession, native language, how often the users eat out in restaurants (in order to find out their general interest in the domain task), and whether they used a SDS before and his prior expectations/experience with a SDS.
After each task the users answer a questionnaire on a forced-choice, 6 point Likert scale (*strongly disagree, disagree, slightly disagree, slightly agree, agree, strongly agree*). This task-based questionnaire enquires about different aspects of the Information Presentation strategy (see Section 3.1). At the end of the session the users were asked to fill out a final questionnaire about his general experience with the "system" (see Section 3.1). Each user received a £10 compensation, with £2 extra when s/he successfully completed all the tasks. We introduced this additional reward in order to encourage attention to task completion.

2.2.2 Data

We have collected 213 dialogues with 18 subjects and 2 wizards in this setup. The dialogues are on average $7.03(\pm 3.0)$ turns and $02:21(\pm 01:28)$ minutes long. Table 2 presents an overview of the data. Note that both wizards contributed to both noise conditions, however the proportions are in diametrical opposition: w1 contributed almost $\frac{3}{4}$ of the noise condition, whereas w2 contributed almost $\frac{3}{4}$ of the no-noise condition. Ideally these proportions would be balanced.
Most of the subjects were female (77.6%), most were students between 20 and 30 years old (63.1%), and most were native speakers of English (83.2%). They can be assumed to have a reasonable interest in restaurants: 66.8% eat out once a month, and 27.67% eat out every week. Most of the subjects have not used a SDS before (57.9%). The ones who did rated their previous experience with 3.5 on average (out of 6) for convenience of use. [4]

[4] Note that we are reporting on the mean for Likert Scales. There is some discussion in the literature whether mean is appropriate for ordinal measures, or whether one should rather report on the median or mode of the data. However, since most of the work in SDS and NLP reports on means for questionnaire data, we do so as well.

wizard	noise	no-noise	total
w1	5 subjects × 12 dialogues = 60 dialogues 72.43% of noise condition	3 subjects ×12 dialogues = 36 dialogues 27.9% of no-noise condition	= 96 dialogues, 8 subjects, 249 NLG actions
w2	2 subjects × 12 dialogues = 24 dialogues 28.6% of noise condition	8 subjects x (11+6+12+18 +10+3×12) dialogues = 93 dialogues 72.1% of no-noise condition	= 117 dialogues, 10 subjects, 219 NLG actions
total	= 84 dialogues, 7 subjects, 194 NLG actions	= 129 dialogues, 11 subjects, 274 NLG actions	213 dialogues, 18 subjects, 468 NLG actions

Table 2: Data collected

The data contains ca. 2236 utterances in total: 1465 system prompts and ca. 771 user prompts.[5] 32% of the system utterances are NLG Information Presentation strategies (n=468). The other system prompts are greeting the user (14.7%), wait messages (33.2%), final prompts (14.2%), informing the user that no matches were found (1.43%), and other system prompts such as the wizard typing in text to send to the user (4.5%).

All the interactions are logged as XML files and the whole conversation is recorded.

2.2.3 Dialogue features/annotations

We automatically extracted 81 features from the XML logfiles, see Appendix A. These features can be categorised into 7 broader categories: General information, e.g. user information; turn information, e.g. turn number; NLG related information, e.g. the chosen IP strategy; task-based information, e.g. slot values; information logged from noise modelling; features from the annotated user reply; and user questionnaire ratings & objective measures.

[5] User prompts are calculated as system prompts minus wait messages minus final messages.

3 Data analysis

This section provides a general analysis of the data set. Note that this analysis doesn't yet address the questions of context-dependent or optimal wizard behaviour, but describes the data set as a whole. We are especially interested in how the two wizards differ in performance and behaviour.

3.1 Performance measures

Dialogue performance can be measured in objective measures, such as dialogue length, and subjective measures, such as user ratings on a questionnaire.

3.1.1 Objective performance measures

Wizards significantly differ in dialogue length in turns (independent samples t-test, $p = .000$) and in minutes (independent samples t-test, $p = .000$), where w2 produces significantly longer dialogues.

wizard		finalNoTurns	totalDuration
w1	Mean	6.22	00:01:53
	N	118	118
	Std. Deviation	2.539	00:01:08.740
w2	Mean	8.03	00:02:56
	N	96	96
	Std. Deviation	3.184	00:01:35.909
Total	Mean	7.03	00:02:21
	N	214	214
	Std. Deviation	2.979	00:01:27.748

However, the noise condition also significantly influences dialogue length in turns (independent samples t-test, $p = .002$) and minutes (independent samples t-test, $p = .000$), where the noise condition produces longer dialogues than the no-noise condition (which confirms our experimental setup). To separate effects of noise condition and individual wizards, we split data by noise condition and compare by wizard. It turns out that dialogue duration in turns and minutes only significantly differs between wizards for the no-noise condition. Wizards produce dialogues with about equal length for the noise condition. We assume that this is due to the user having to repeat/ re-provide lots of slot-values/ attributes to the system because of distortion by the noise model.

3.1.2 Subjective performance measures

To measure subjective dialogue performance we looked at the ratings of the task questionnaire and the final questionnaire on a 6-point Likert scale.

Task questionnaire:

1. The system's voice was easy to understand. (**TTS quality**)
2. The way the system presented information was good. (**Information Presentation**)
3. The system's utterances had the right length. (**Utterance Length**)
4. The system gave me a good overview of all the available options. (**Coverage**)
5. The restaurant I chose was a good match for this task. (**Task Success**)

In general, users gave high ratings: the average user satisfaction (defined as the normalised sum across all the 5 questions from the task questionnaire, see Appendix A) is $4.7 \pm .8$ out of 6 points. Wizards score significantly different on question3/Utterance length (non-parametric 2-independent samples Mann Whitney Test, $p = .005$) and question5/ Task success ($p = .026$), where w2 is scored higher than w1. A possible explanation why w1 is scored worse on question3/Utterance length is that w1 generates more combined NLG actions (e.g. summaryRecommend) than w2. A possible explanation why w2 is scored better on question5/ Task success is that w2 chooses more attributes than w1, which gives the user additional information to perform the task.

wizard		N	Mean	Std. Deviation
w1	question1	721	4.74	1.086
	question2	721	4.50	1.307
	question3	721	4.85	.840
	question4	721	4.20	1.347
	question5	721	4.75	1.167
	usersatisfaction	721	4.6072	.83563
w2	question1	744	5.13	.915
	question2	744	4.30	1.247
	question3	744	4.55	.911
	question4	744	4.54	1.330
	question5	744	5.18	.936
	usersatisfaction	744	4.7403	.73592

Also, the no-noise condition is significantly higher ranked than the noise condition ($p \leq .004$ for question 1-5 and User Satisfaction). In the noise condition the wizards get rated significantly different on question1/TTS quality ($p = .000$) and question5/Task success ($p = .030$), where wizard w2 gets rated higher. In the no-noise condition the wizards get rated significantly different on question1/TTS quality ($p = .035$) and question4/Coverage ($p = .004$), where w2 gets rated higher as well.

Final questionnaire:

1. The time delay of the system affected my ratings. (**timeDelay**)
2. I wasn't able to choose a restaurant because I could not understand the system's voice. (**TTSandTaskSuc**)
3. I'd like to call the system in the future for restaurant information. (**Future Use**)

wizard		N	Minimum	Maximum	Mean	Std. Deviation
w1	timeDelay	721	0	6	3.12	1.613
	TTSandTaskSuc	721	0	4	1.68	1.070
	future	721	0	6	4.40	1.532
w2	timeDelay	744	0	5	3.01	1.608
	TTSandTaskSuc	744	0	5	2.06	1.769
	future	744	0	6	4.00	2.080

The two wizards are rated about equal on the final questionnaire. However, users want to use the no-noise system significantly more in the future ($p = .000$). The noise system is rated significantly worse on time delay ($p = .004$) and TTS quality ($p = .001$), which again confirms our experimental setup. The wizards still get equally rated in the noise vs. no-noise condition (no significant difference).

Users were also encouraged to enter additional comments in a text panel below the final questionnaire. A list of all additional comments (with annotations on corresponding wizard and noise condition in brackets) can be found in Appendix B. Many users left comments about the voice and the voice quality, where opinions on the audibility highly varied between 'hard to understand' to 'easy to understand'. Furthermore, the scope of the system was not clear to some of the users. They would ask for information the system didn't know about, like the phone number or whether it had parking space etc. This might partially be due to the abstract task descriptions which confused some users about the system's capabilities.

3.2 NLG behaviour

In this Section we report on average NLG behaviour (NLG actions and attributes chosen) in terms of frequencies for the whole data set (i.e. not context dependent or optimal). We also investigate differences in NLG behaviour between different wizards and noise condition.

3.2.1 Average NLG behaviour

NLG strategies We have extended the 7 possible Information Presentation strategies presented in Section 1.1 with the different realisations listed in Section2.1.1. The wizard has to choose between 17 possible NLG realisations every time he presents some information (not counting variation in attributes here):

1. Recommend

2. CompareByItem
3. CompareByAttr
4. SummaryWithUM
5. SummaryNoUM
6. CompareByItemRecommend
7. CompareByAttrRecommend
8. SummaryWithUMRecommend
9. SummaryNoUMRecommend
10. SummaryWithUMCompareByItem
11. SummaryNoUMCompareByItem
12. SummaryWithUMCompareByAttr
13. SummaryNoUMCompareByAttr
14. SummaryWithUMCompareByItemRecommend
15. SummaryNoUMCompareByItemRecommend
16. SummaryWithUMCompareByAttrRecommend
17. SummaryNoUMCompareByAttrRecommend

We observe 13 of these actions in the data, see Figure 4. Four have never been explored by the wizards (CompareByAttributeRecommend, SummaryWithUMByItemRecommend, Summary-WithUMCompareByAttributeRecommend, SummaryNoUMCompareByAttributeRecommend). The full sequence (Summary+Compare+Recommend) appears 5 times.

It is also interesting to note that 29.2% of the time more than one NLG action was generated, i.e. a more complex combined action was chosen, such as summarise and compare in a single utterance (SummaryCompare). Also, wizard w1 generated significantly ($p = .000$) more combined actions than w2 (w1: 40.6%, w2: 18.9%).

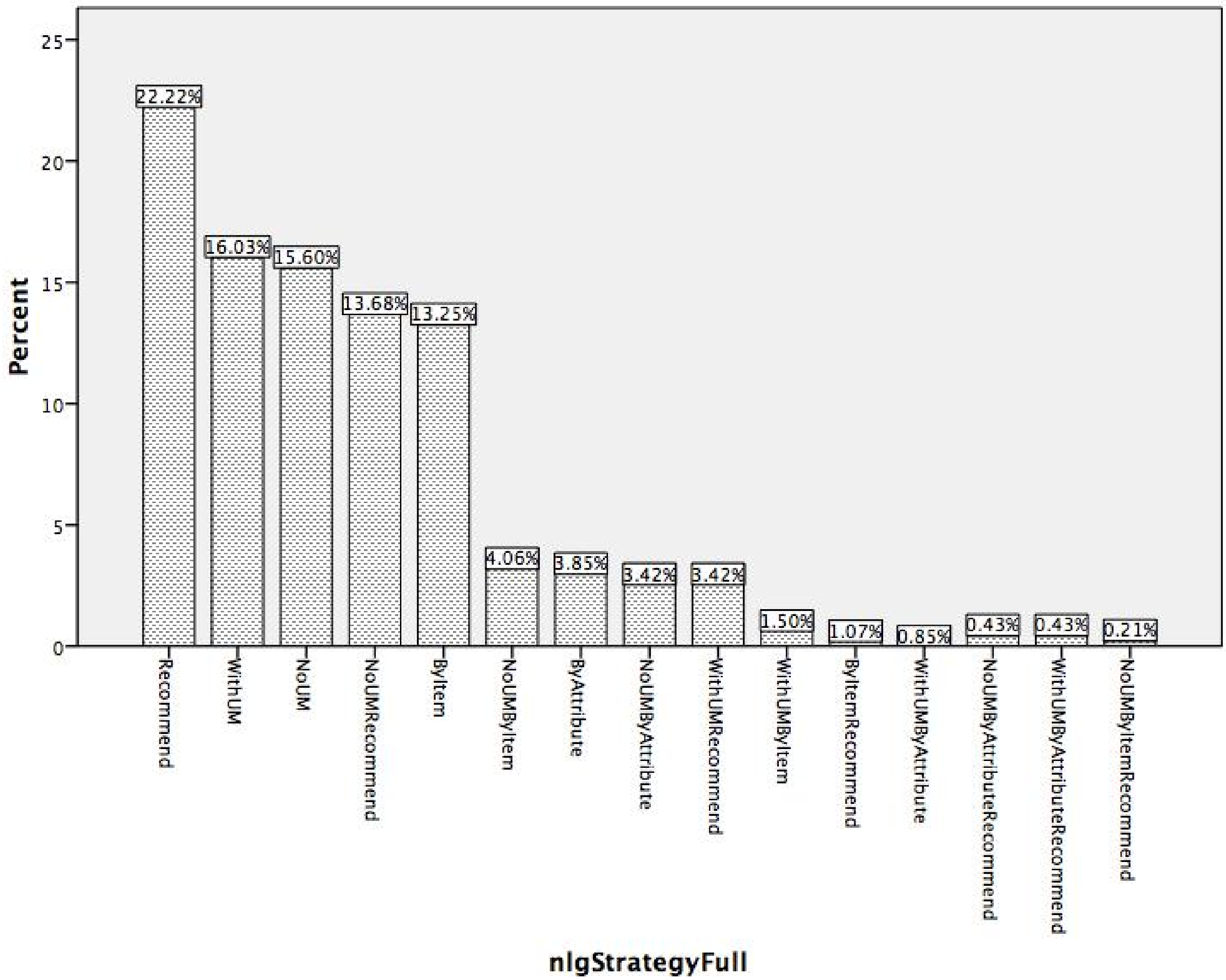
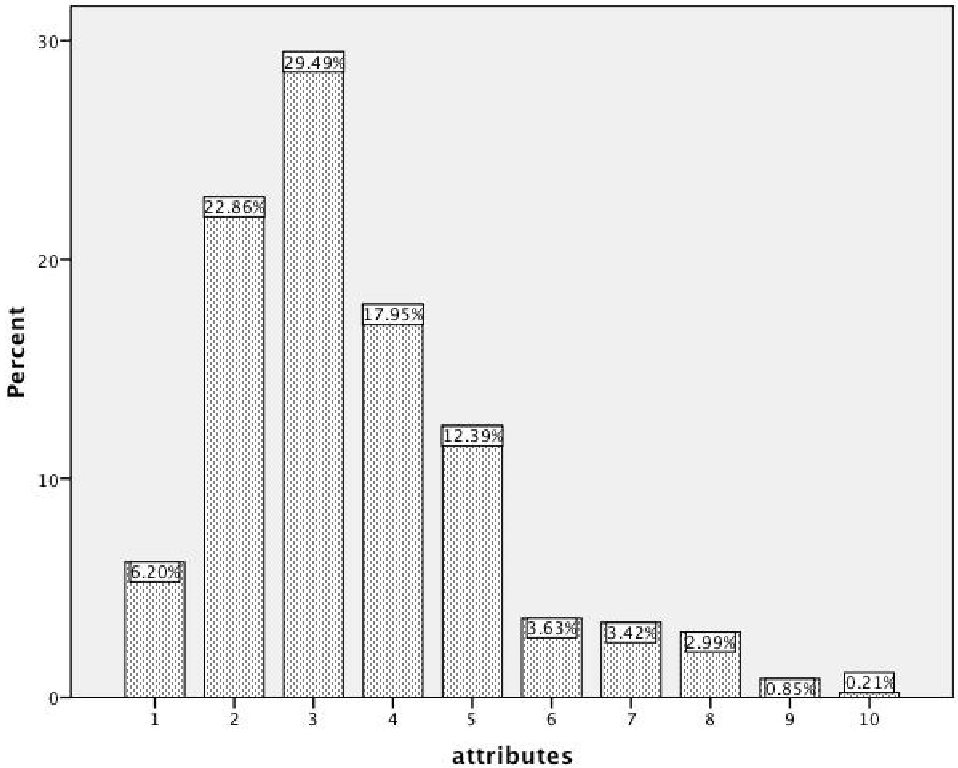
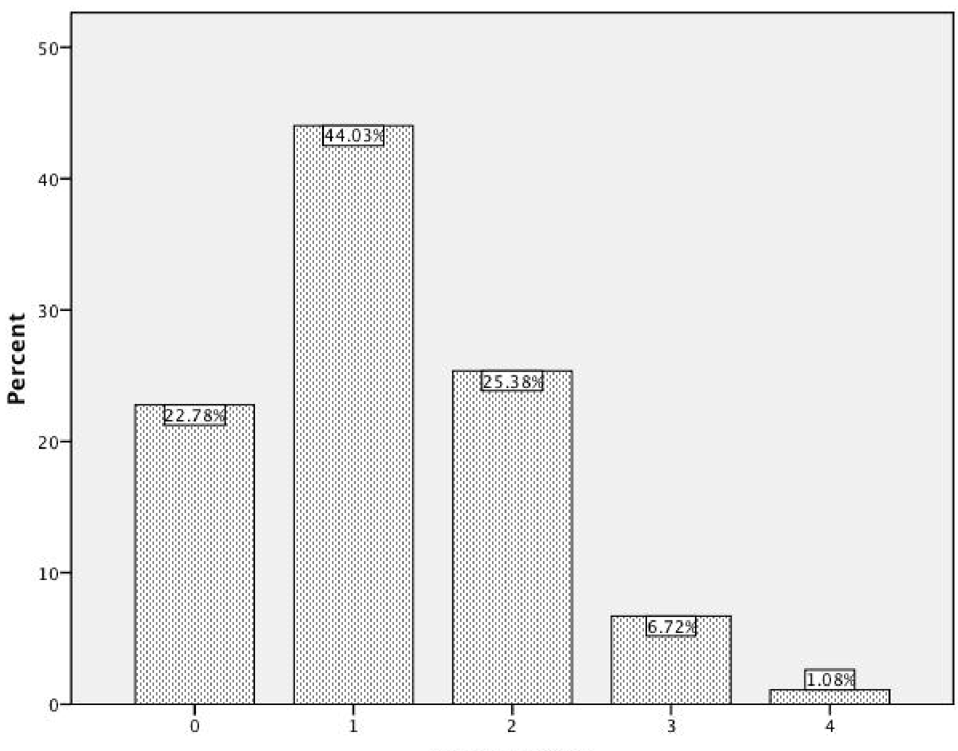


Figure 4: Frequencies of NLG strategies in the full WoZ corpus

Table 3: Percentages of number of attributes (top) and number of Added Slots (bottom) in the corpus





Attributes We are interested in 3 different aspects of attribute selection: how many attributes were generated in total (*attributes*), whether the attribute/s specified in the last user query are chosen by the wizard (`userSlotsSelected`) [6], and how many new, unique attributes the wizard chooses to add (`noAddedSlots`). Note that this division between known/old and new information is similar to the theme/rheme idea in Information Structure theory [Halliday, 1985]. The wizards generate 1-10 attributes in total, [7] with 3 attributes being the most frequent (29.49%).

The number of added slots only considers unique attribute tokens, i.e. duplicates are only counted once. As such, the maximum number of added slots is 4. Wizards added one additional attributes in most of the cases (44.03%). Wizards also include the default user slots in 85.3% of the cases.

3.2.2 Unified NLG actions

In the following we argue for unifying/ collapsing the 17 possible realisations into the 7 original strategies again. First of all, the wizards reported that they did not pay much attention to the different realisations (withUM/noUM, byItem/byAttribute), due to the time pressure.

In the following we list the frequencies of the different realisations for each NLG action, and frequencies for each wizard.

- **Summary**: in 62.7% NoUM (n=175), in 37.3% WithUM (104)

  w1: in 75.5% NoUM (n=87), in 24.3% WithUM (28)

  w2: in 53.7% NoUM (n=88), in 46.3% WithUM (76)

- **Compare**: in 69.1% ByItem (94), in 30.9% ByAttr (42)

  w1: in 66.25% ByItem (53), in 33.75% ByAttr (27)

  w2: in 73.2% ByItem (41), in 26.8% ByAttr (15)

For all these reasons, and in the interest of avoiding data sparsity, we only investigate the following 7 "unified" NLG strategies for the rest of this document.

		Frequency	Percent	Cumulative Percent
	summary	148	31.6	31.6
	Recommend	104	22.2	53.8
	compare	80	17.1	70.9
	summaryRecommend	80	17.1	88.0

[6] Note that the attribute/s specified in the last user query are pre-selected on the GUI by default, see Section 2

[7] Note that there are 5 unique attributes (cuisine, price, location, food quality, service) which can be selected for each single action. That is, the maximum number which can be generated is $3 \times 5 = 15$ attributes.

	summaryCompare	46	9.8	97.9
	compareRecommend	5	1.1	98.9
	summaryCompareRecommend	5	1.1	100.0
	Total	468	100.0	

3.3 Sequences of "unified" NLG strategies

We also explore sequences of NLG IP strategies from a dialogue perspective, i.e. NLG choices over several turns. The model we are following in this document by [Rieser and Lemon, 2009b] considers the NLG planning task per turn. However, we think that NLG and DM planning are two closely inter-related problems in general. We explored some combined models in previous work [Rieser and Lemon, 2008, Lemon, 2008]. Here we represent the NLG history as a feature for learning (`previousNLG`, see Appendix A).

A complete list of all the NLG sequences explored by our wizards can be found in Appendix C. Most of the time, the wizards start the Information Presentation phase in the dialogue with a summary. The following sequences occur 10 or more times in the data (with "-" denoting a new turn):

1. summaryRecommend (23)
2. summary-compare (18)
3. summary-Recommend (15)
4. Recommend (11)
5. summaryCompare (11)

3.4 Differences in NLG behaviour between wizards

We now investigate differences in NLG behaviour across wizards and noise condition in terms of frequencies for the whole data set. We use Pearson's chi-square test for the NLG action analysis, since we are dealing with nominal data. For analysing the number of attributes, we use independent non-parametric tests (Wilcoxon-Mann-Whitney for 2 independent samples, and Kruskal-Wallis one-way analysis of variance for k-independent samples), since we have numerical, non-parametric data.

3.4.1 NLG strategies

Wizards significantly differ (Chi-square, $p = .000$) in how often they choose a specific NLG strategy, see Figure 5. The most frequent strategy for wizard w1 is Recommend. The most frequent strategy for w2 is Summary. Wizard w2 never explores the full sequence.

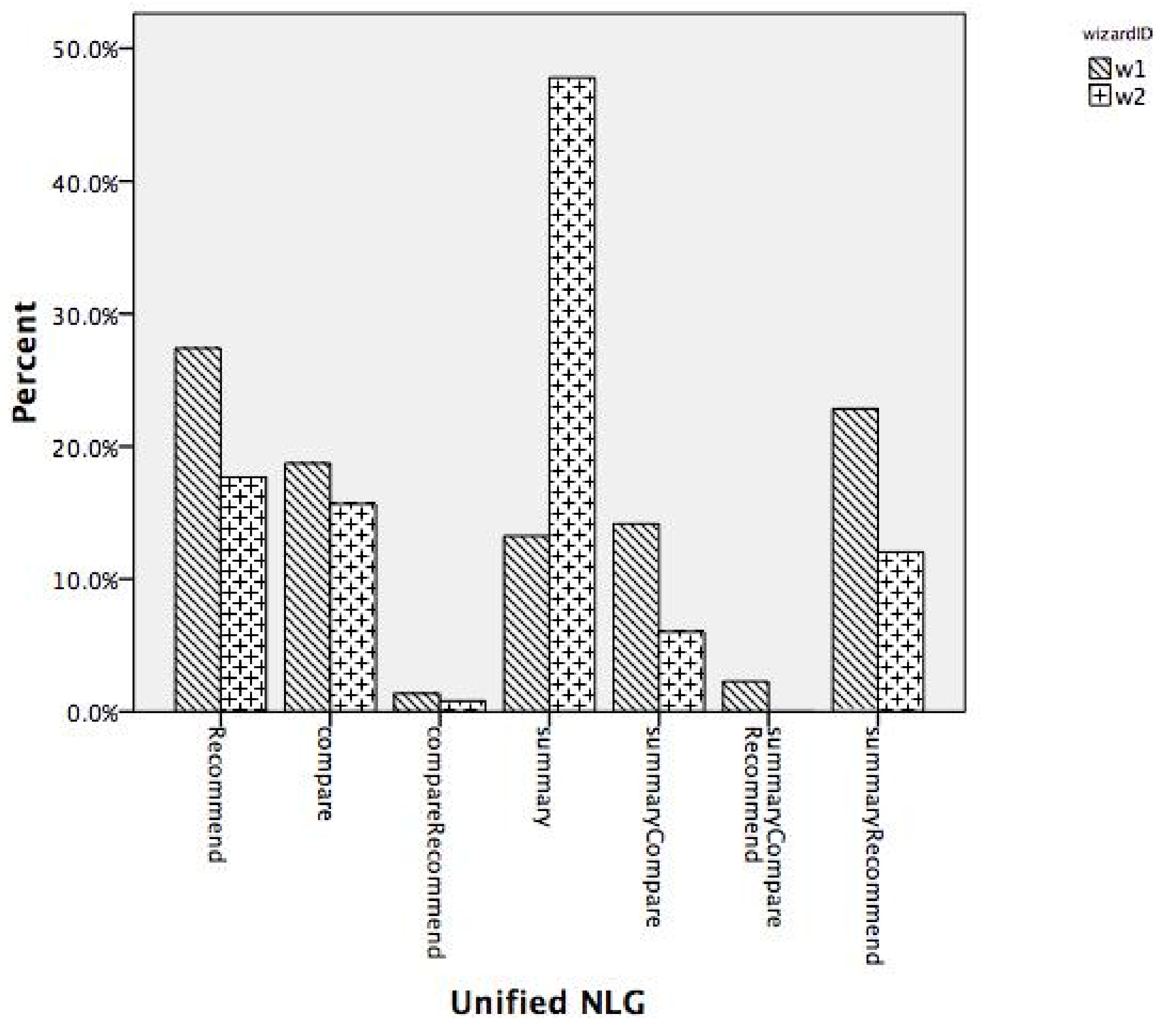


Figure 5: Differences in NLG strategies: frequencies per wizard

The dialogue sequences of NLG IP strategies also significantly differ (Chi-square, $p = .000$). The most frequent sequences for w1 (≥ 9) are:

- SummaryRecommend (15)
- SummaryRecommend - Recommend (10)

The most frequent sequences for w2 (≥ 10) are:

- Summary - Compare (11)
- Summary - Recommend (10)

3.4.2 Attribute Presentation

The total number of attributes presented significantly differs (Wilcoxon-Mann-Whitney, $p = .013$) between wizards (w1: $3.31 \pm .108 <$ w2: $3.78 \pm .118$). Wizards also significantly differ (Wilcoxon-Mann-Whitney, $p = .000$) in the number of additional attributes/slots chosen (w1: $.78 \pm .053 <$ w2: $1.49 \pm .058$). However, both wizards tend to select the attributes given by the user in most of the cases (not significant difference between wizards, Chi-square).

wizard			Frequency	Percent	Cumulative Percent
w1		no	34	15.5	15.5
		yes	185	84.5	100.0
		Total	219	100.0	
w2		no	35	14.1	14.1
		yes	214	85.9	100.0
		Total	249	100.0	

We also compare attributes per NLG strategy, i.e. we investigate whether the NLG strategy has an influence on how many attributes are getting chosen by the wizard. A correlation between NLG strategy and attributes chosen will support a hierachical model, as suggested by [Rieser and Lemon, 2009b].

We find that the number of attributes and added attributes both differ significantly (Kruskal-Wallis, $p = .000$) per NLG strategy. The individual wizards chose different numbers of attributes for each strategy (Wilcoxon-Mann-Whitney, $p = .05$), and also add different numbers of attributes/slots per strategy (Wilcoxon-Mann-Whitney, $p = .05$).

Furthermore, we investigate attribute type (cuisine, price, location, food quality, service) per strategy. We find that significantly different (Chi-square with Yate's correction for sparse data, $p = .000$) attribute types are chosen per strategy, for example the most frequent ones for Compare and Recommend are *cuisine, price, location*, for summary *cuisine, location*, and for SummaryRecommend all 5 attribute types. However, these results are influenced by data sparsity (lots of zero counts).

3.4.3 Conclusions

In sum, both wizards explore significantly different NLG strategies in terms of frequencies of strategies, number of attributes, and added attributes. They also differ in how many attributes and which attribute type they choose for each strategy. We conclude that this is interesting data to instantiate a Reinforcement Learning (RL) system from, as many different strategies are explored.

Furthermore, wizards also choose different IP strategies at different points of the dialogue: NLG sequences on a dialogue level are significantly different, which suggests that the dialogue position is an important feature for choosing a NLG strategy. Furthermore, the inter-dependency between NLG strategy and attributes chosen favours a hierarchical generation model.

3.5 Differences in NLG behaviour between noise conditions

NLG strategies also significantly differ between noise conditions (Chi-square, $p = .000$). However, wizards contributed unequal portions to the noise conditions, so that this effect might be due to differences in wizard behaviour. We therefore compare the NLG choices for each wizard in each noise condition (i.e. we split the data by noise condition and compare by wizard). This analysis shows that each wizard changes his strategy per noise condition significantly ($p = .000$). However, the number of attributes stays constant across noise conditions.

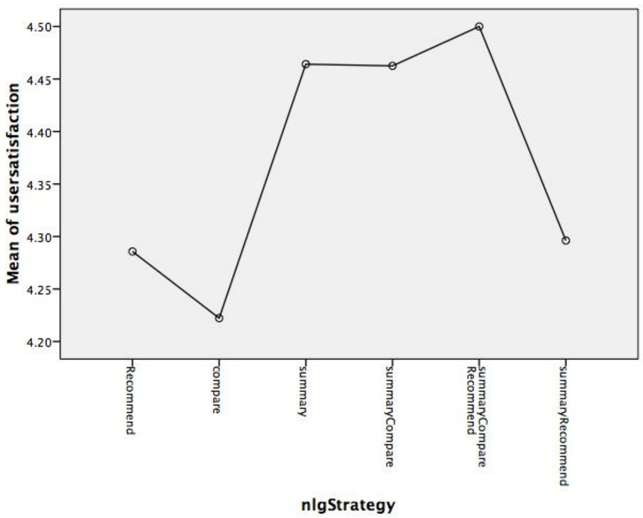
Another interesting question is whether users change their preferences for NLG output/ Information Presentation strategies in the presence of noise. Our hypothesis was that shorter utterances are preferred in the presence of noise, see Section 2.1.2. We therefore took all the utterances rated above the median (see Section 4) and compared frequencies of NLG actions and attributes chosen.

There are more combined NLG actions amongst the highest rated for the noise condition (29.3%) than for the no-noise condition (22.1%), see Figure 6, which contradicts our hypothesis.[8] The same seems to be true for the number of attributes: more attributes are added for the noise condition than the no-noise condition, see Figure 7 (left). However, if we look at number of added attributes we see that fewer attributes are getting added for the noise condition, see Figure 7 (right). Therefore we can assume that the high number of attributes for the noise condition is a function of grounding, i.e. the attributes provided by the users are repeated by the wizard in order to make sure that they were understood correctly.

In previous work [Rieser and Lemon, 2008] we explored the hypothesis that SUMMARY serves a different function to COMPARE and RECOMMEND, in the sense that it gives an overview of the data and also serves a "grounding" function [Clark and Schaefer, 1989]. Therefore, our hypothesis is that SUMMARY gets rated higher in the noise condition than in the no-noise condition. Since strategies under the no-noise conditions are in general higher rated than in the noise condition (see 3.1.2) we cannot compare user scores for SUMMARY directly (unless we normalise in some way), but compare relative user rankings within individual noise conditions. Table 4

[8]Note that 4 of the 5 full NLG strategies (Summary+Compare+Recommend) are rated above average. However, only one is in the noise condition.

shows mean user satisfaction scores per NLG strategy for noise and no-noise conditions. Strategies including a SUMMARY are ranked highest under the noise condition. SUMMARY as a single strategy is ranked significantly ($p > .5$) higher than COMPARE or RECOMMEND. In contrast, strategies including SUMMARY are not amongst the highest rating ones for the no-noise condition. In addition, SUMMARY, COMPARE, and RECOMMEND are rated equally for the no-noise condition.



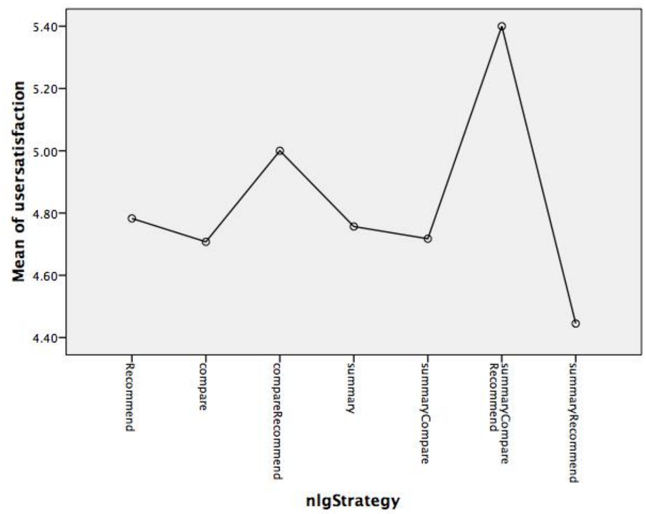


Table 4: User scores for NLG strategies under noise (left) and no-noise condition (right).

In sum, users did not seem to prefer shorter utterances in the noise condition, as we assumed. Users also prefer to get feedback on the attributes they already have provided and prefer to hear more summaries, both of which serve a "grounding" function in dialogue.

event	effect/ prompt	noise	no-noise
Zero DB items returned	"*Sorry, there are no matching restaurants.*"	2.4%	0.6%
Only one DB item is found	Only RECOMMEND is enabled."*There is only one matching item. I can recommend....*"	5.1%	2.8%
Restaurants with "bad" qualities are found.	"*I can recommend the restaurant called Bonsai. It is a seafood restaurant . Unfortunately it has poor food quality. But it is in the cheap price range.*"	4.1%	0.1%

Table 5: Effects of noise on the dialogue: Frequencies of events for noise and no-noise condition.

On the other hand, wizards don't seem to change their NLG behaviour (in terms of strategy and attributes) after noise was introduced. One possible explanation is that wizards in this setup can only monitor the dialogue in terms of slot values shown to them on the GUI and did not hear the user directly (the experimenter did the actual interpretation in this setup). Thus, the wizards did not know when noise was introduced, nor did they know whether the strategy was unsuccessful due to noise-related problems. However, noise obviously did have an effect on the dialogue, primarily in respect with the items returned from the database. Table 5 lists the main observed effects of noise and their frequencies in the noise and no-noise condition.

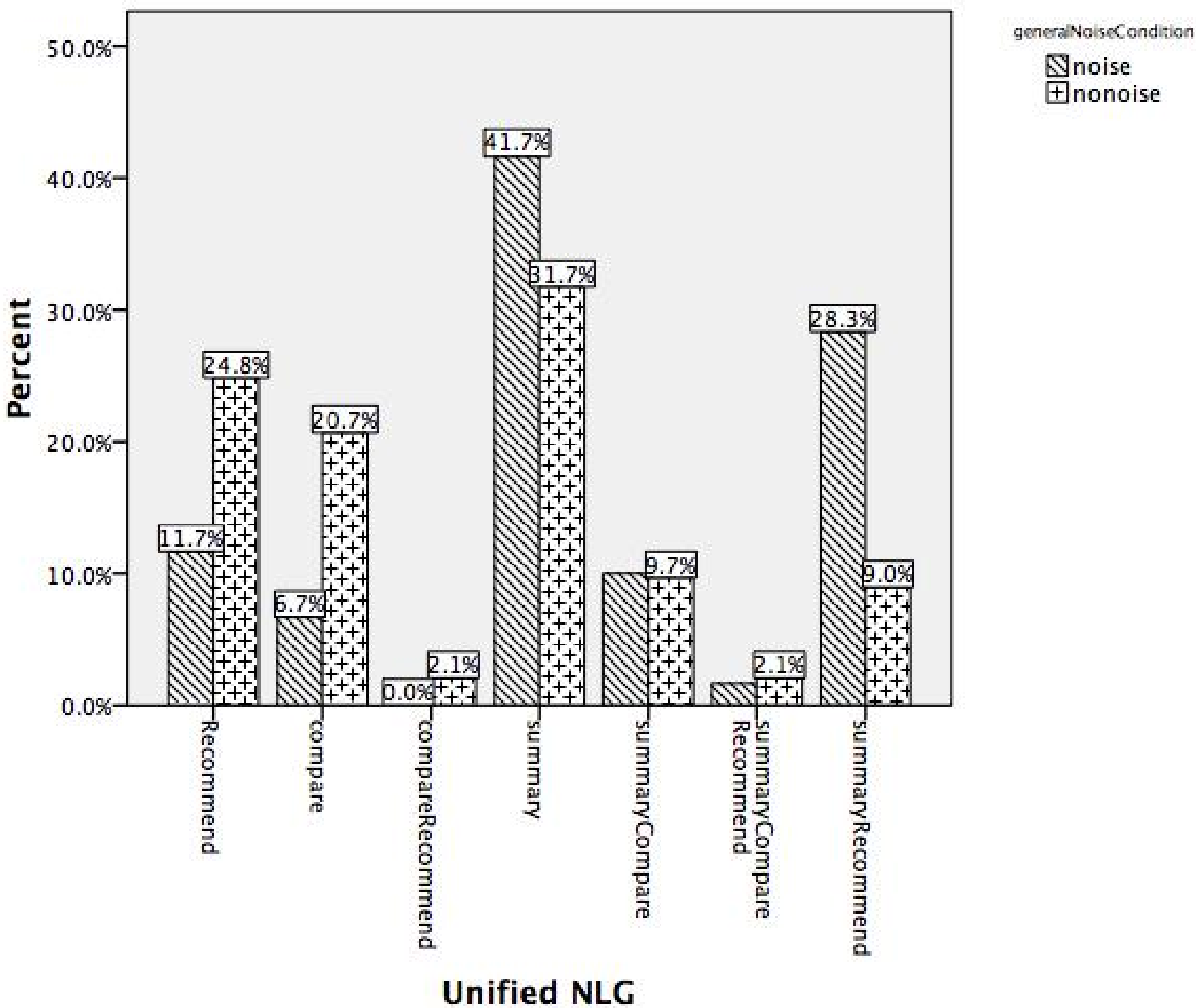


Figure 6: Highly rated NLG strategies for noise vs. no-noise condition

3.6 User replies

We are also interested in the reply a user might give to a specific IP strategy. In future work we will use the user reply to build user simulations for NLG as well as part of our reward function,

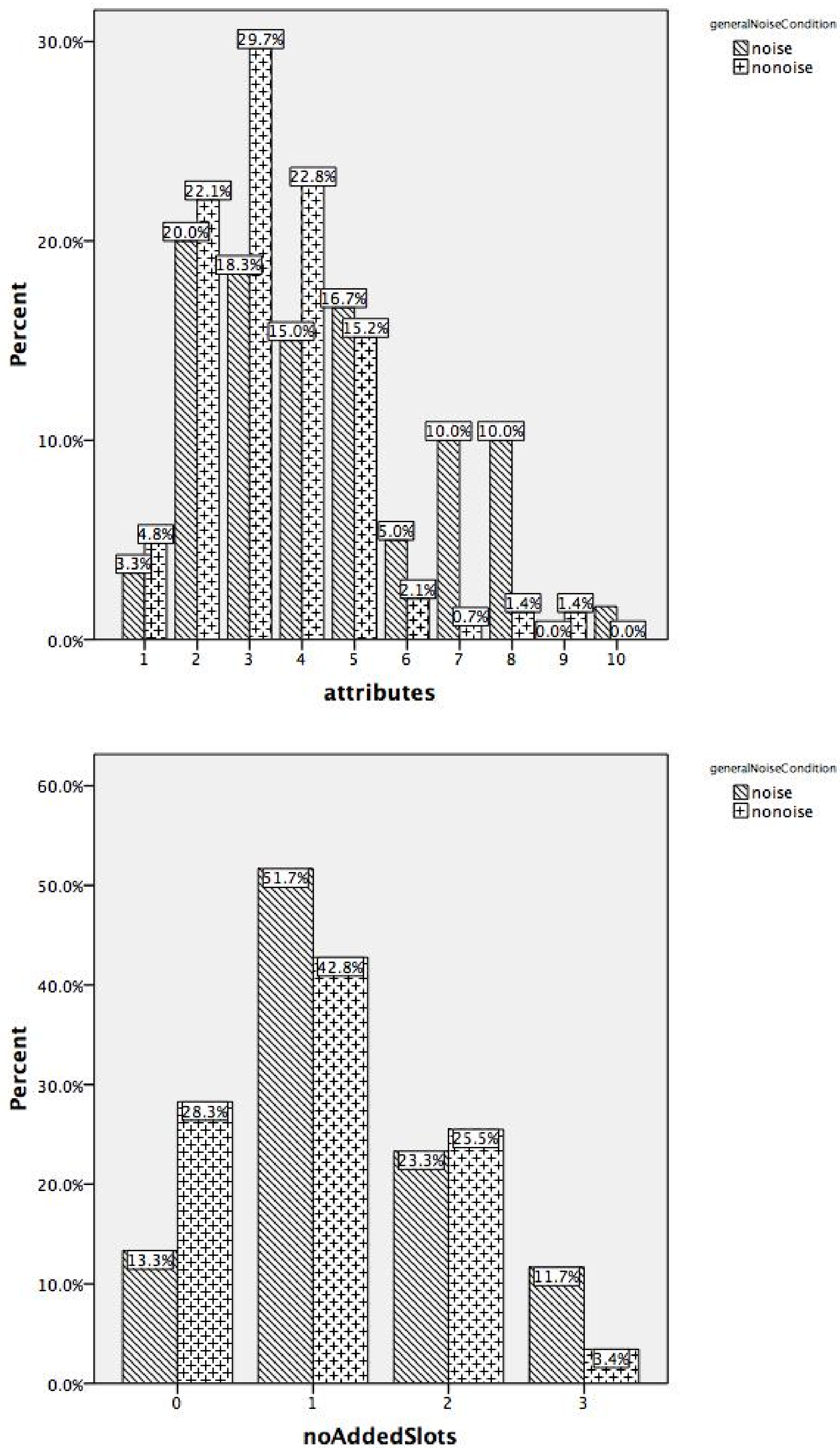


Figure 7: Highly rated attribute selection (top) and number of attributes (bottom) for noise vs. no-noise

see [Rieser and Lemon, 2009b]. In the following we survey the user replies in our data.
We are interested in the following types of user reply:

1. `select`: the user chooses one of the presented items, e.g. *"Yes, I take this one."* This reply type indicates that the information presentation was sufficient for the user to make a choice.

2. `requestMoreInfo`: the user asks for more information, e.g. *"Can you recommend me one?", "What is the price range of the last item?"* This reply type indicates that the wizard failed to present the information the user was looking for.

3. `addInfo`: the user provides more attributes, e.g. *"I am searching for something cheap."* This reply type indicates that the user has more specific requests, which s/he is able to specify after being presented with the current information.

4. `askRepeat`: the user asks the system to repeat the same message again, e.g. *"Can you repeat?"* This reply type indicates that the utterance was either too long for the user to remember, or the TTS quality was not good enough, or both.

5. `silence`: the user is not saying anything. In this case it is up to the wizard to take initiative.

6. `hangup`: the user slams down the phone.

7. `other`:

Most of the time users select an item, request more information, or add more information, see Figure 8.

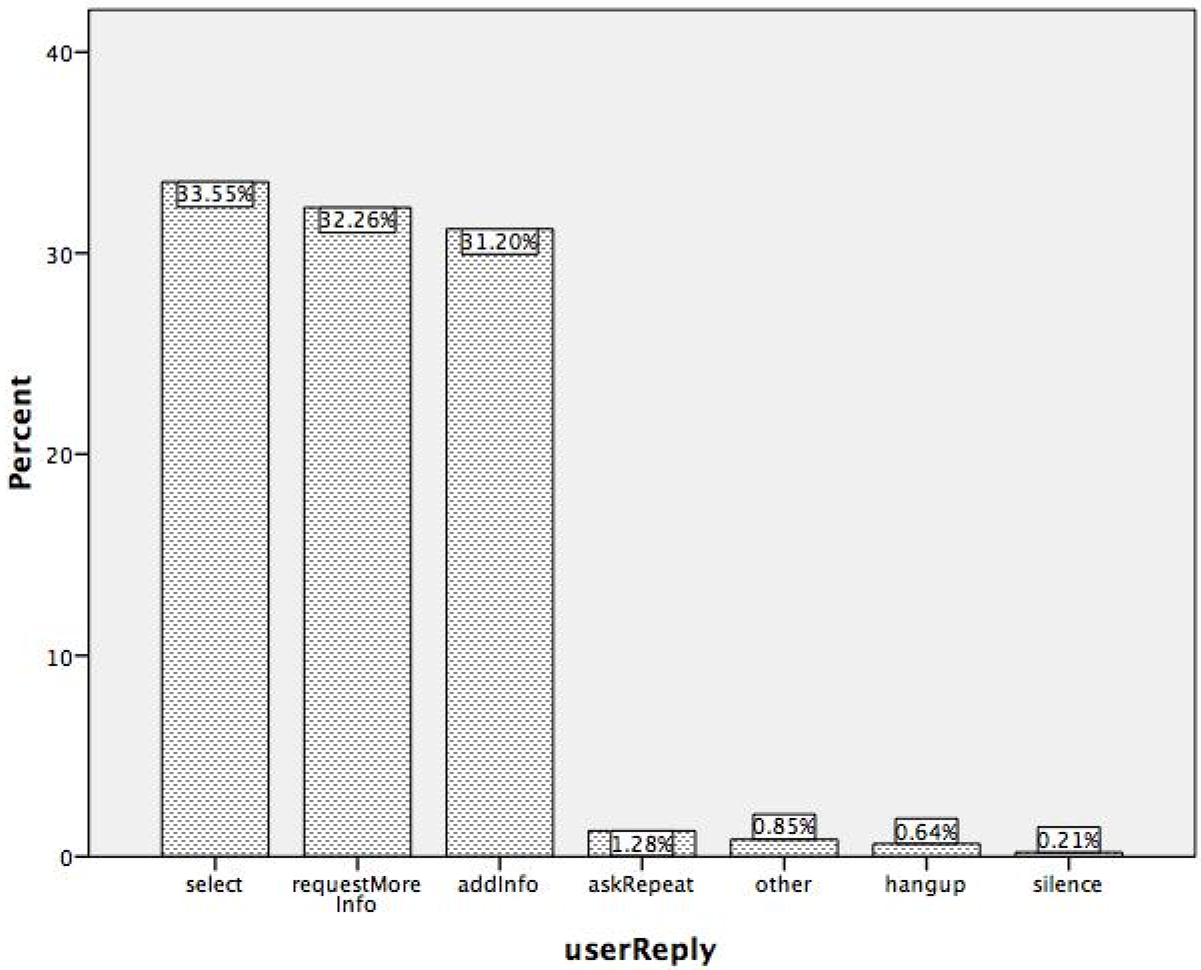


Figure 8: User reply frequencies

We are especially interested in the type of reply the user gives after a specific NLG strategy. Figure 9 shows the percentage of each user reply after a NLG strategy. The following table presents userReply counts per NLG strategy:

Wizard NLG	User Reply	Frequency	Percent	Cumulative Percent
Recommend	select	65	62.5	62.5
	requestMoreInfo	28	26.9	89.4
	addInfo	11	10.6	100.0
	Total	104	100.0	
compare	select	35	43.8	43.8
	requestMoreInfo	27	33.8	77.5
	addInfo	14	17.5	95.0
	other	2	2.5	97.5
	hangup	1	1.2	98.8
	silence	1	1.2	100.0
	Total	80	100.0	
compareRecommend	select	5	100.0	100.0
summary	addInfo	85	57.4	57.4
	requestMoreInfo	52	35.1	92.6
	askRepeat	6	4.1	96.6
	select	4	2.7	99.3
	hangup	1	.7	100.0
	Total	148	100.0	
summaryCompare	requestMoreInfo	20	43.5	43.5
	select	15	32.6	76.1

	addInfo	11	23.9	100.0
	Total	46	100.0	
summaryCompareRecommend	select	3	60.0	60.0
	requestMoreInfo	2	40.0	100.0
	Total	5	100.0	
summaryRecommend	select	30	37.5	37.5
	addInfo	25	31.2	68.8
	requestMoreInfo	22	27.5	96.2
	other	2	2.5	98.8
	hangup	1	1.2	100.0
	Total	80	100.0	

It is interesting to note that the users only ever ask the system to repeat after hearing a summary (100% askRepeat, see Figure 9). The only time users stay silent is after comparisons (100% silence, see Figure 9). The case where the user hangs up the phone happens 5 times in the whole corpus. 3 of those cases are after IP strategies. [9] A closer analysis of the data shows that these 3 cases are due to the noise, i.e. the system repeatedly presenting the wrong (noisy) information to the user.

Furthermore, we have data sparsity for almost every system utterance (i.e. some user reply types have 0 frequencies after a system NLG act). User simulations for training Reinforcement Learning based strategies, however, should facilitate the exploration of the full state action space. We therefore will experiment with techniques to overcome data sparsity for user simulations, e.g. cluster-based models [Rieser and Lemon, 2006a].

[9] The other 2 are after no matches were found and a system error.

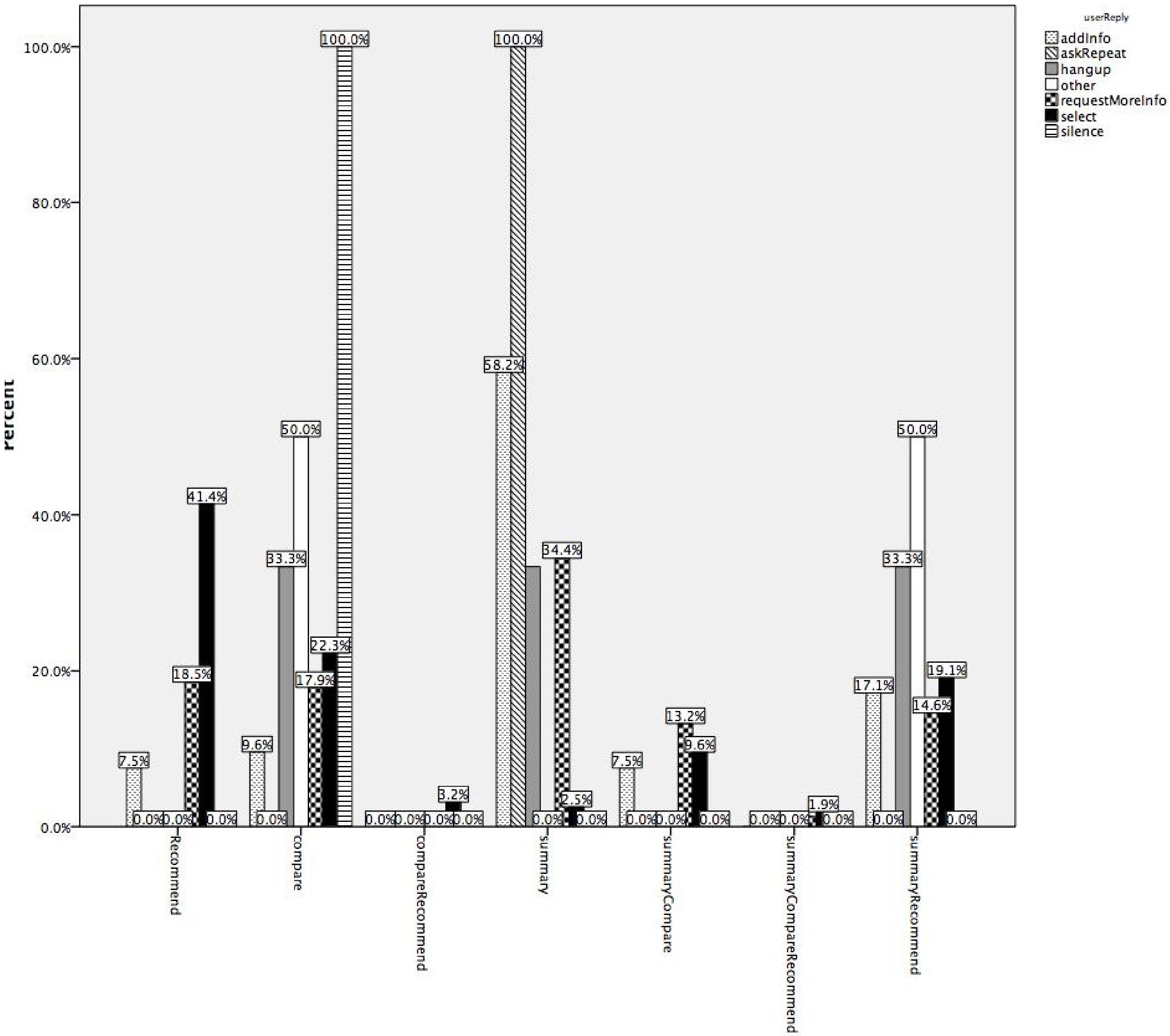


Figure 9: User reply frequencies after a NLG strategy

4 Optimal Wizard NLG strategies in context

4.1 Introduction

In this section we present a method for discovering "optimal" wizard NLG behaviours in context. In order to avoid data sparsity, we define "optimal" as having above average user ratings. The presented method can be applied to other definitions of "optimal". We first describe the best NLG strategies as they occur in the data (see Section 4.2).

We then describe optimal context-dependent NLG strategies. The hypothesis we set out to test is that some NLG decisions are better than others in some dialogue contexts, and that these can be discovered using Machine Learning techniques.

We define the dialogue context as a set of runtime dialogue features, see Section 4.3. We apply feature engineering methods, such as feature ranking and feature selection to investigate which context features influence the wizards' decisions. We also learn a supervised classification model in order to describe patterns in the data. We compare the features and the models learned on the best-scoring data against the lower ranked data set. In Section 5 we apply the same methods for discovering the wizards' attribute selection strategy.

We investigate the hypothesis that human wizards' NLG behaviour can be modelled in a hierarchical manner (see Figure 10).

A Chi-square test shows that there is a highly significant ($p < .001$) association between NLG strategy and user attributes selected and attributes added, as well as between user attributes selected and number of attributes added ($p = .002$). However, a Chi-square test does not tell us anything about the direction of the association, nor whether outcomes of higher level decisions are predictive of lower level decisions. In the following, we use Machine Learning and Feature Selection techniques to investigate this hypothesis.

First, the NLG strategy gets chosen. The selected NLG strategy then feeds as a feature into attribute selection. For attribute selection we first predict whether the attributes last mentioned by the user get repeated by the system. This information then feeds into a final classifier which predicts how many new/unknown attributes are added by the wizard.

4.2 Optimal NLG strategy

In order to avoid data sparsity, we define "optimal" as having above average user ratings. In particular, we learn a model of all dialogues/utterances which fall above the median of user satisfaction ($median_{usersatisfaction} = 5$). This definition results in 205 learning instances for the optimal wizard strategy, including data from 16 subjects and 93 dialogues. It also contains about equal proportions from each wizard (w1: 42%, w2: 58%) so that we learn an "average" optimal policy, as generated by both wizards. The non-optimal part of the data contains 263 learning instances. Note that many other definitions of "optimal" are possible. [10] For example, we could only

[10] In optimal control theory, "optimal" can be defined as 'most favourable' according to a cost function [Bellman, 1957]. In annotated data this cost function can just be 'according to a label', in our case the ratings

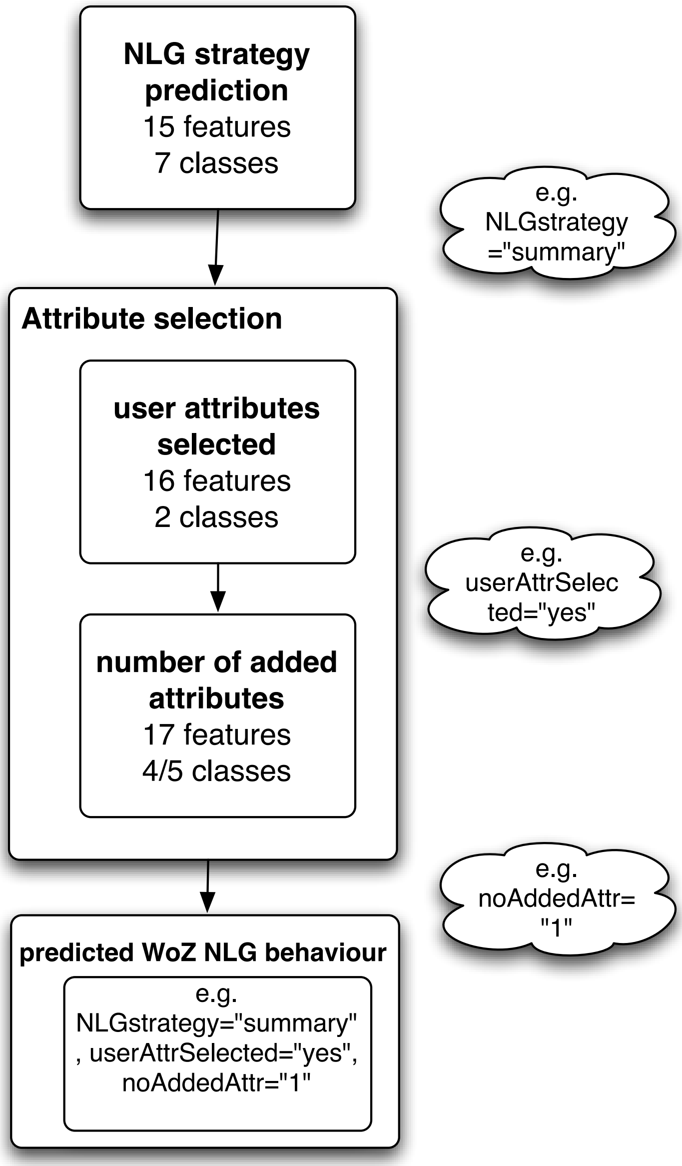


Figure 10: Predicting NLG behaviour in a hierarchical manner

consider examples were the user satisfaction is highest (= 6 points). However, this definition results in only 30 learning instances from 16 dialogues and 5 users, and does not yield to enough data to build good statistical models. Also, each of the 12 dialogues contain different NLG sequences (see section 4.2.2), so that we can conclude that there is no unique NLG pattern which is considered to be "optimal" by the users.

4.2.1 Description of best-scoring data

70.0% of the highest rated dialogues are in the no-noise condition, which confirms our setup. Furthermore, all seven possible NLG strategies are present, see Figure 11. Four of the five full NLG sequences (summary+compare+recommend) are rated above average.

We compare the percentages/distribution of NLG sequences for the data set rated below the median (in the rest of this document referred to as "non-optimal") and the data set rated above the median (referred to as "optimal"), as displayed in Figure 11. We observe that the distribution of NLG actions between data sets is non-significant (Chi-square, $p = .441$); nor does the distribution of `userSlotsSelected` differ (Chi-square, $p = .748$), see see Figure 12, nor the `noAddedSlots` (Chi-square, $p = .307$), see Figure 13. We therefore conclude that there must be contextual differences between strategies present in the optimal and the non-optimal data set, since the users rated strategies differently. In other words, our hypothesis is that good NLG strategy choice is context-dependent. In the following we use Machine Learning techniques to verify our hypothesis and describe contextual differences.

4.2.2 Discussion: High ranked NLG sequences

We need to take into account that users only gave ratings at the end of a dialogue. We therefore also investigate the highest rated sequences of NLG strategies (spread over several IP turns) within one dialogue. An overview of all the highly rated NLG sequences can be found in Appendix D. In the following we list the highest rated NLG sequences which occur more than 3 times:

1. turn1: summary - turn2: compare (10)
2. turn1: summary - turn2: recommend (9)
3. in one turn: summaryCompare (6)
4. in one turn: summaryRecommend (6)
5. in one turn: recommend (5)
6. in one turn: compare (4)

given by the users. In situation where there is no annotated data available, however, the cost function must be inferred/ modelled from data first, e.g. using PARADISE [Walker et al., 2000] or inverse RL [Ng and Russell, 2000].

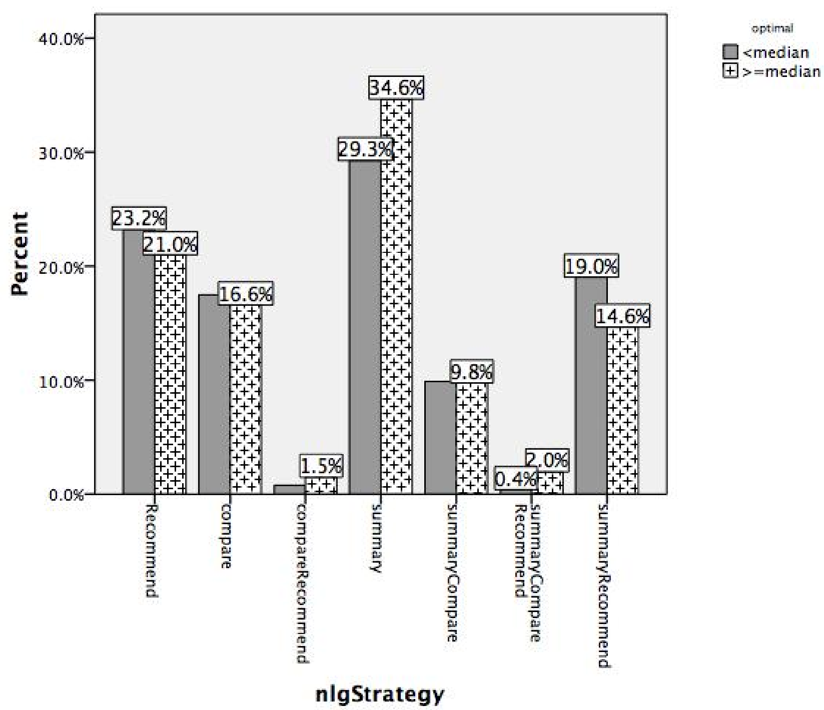


Figure 11: Percentages for highly rated / “optimal” NLG strategies vs. lower rated/ “non-optimal” NLGstrategies

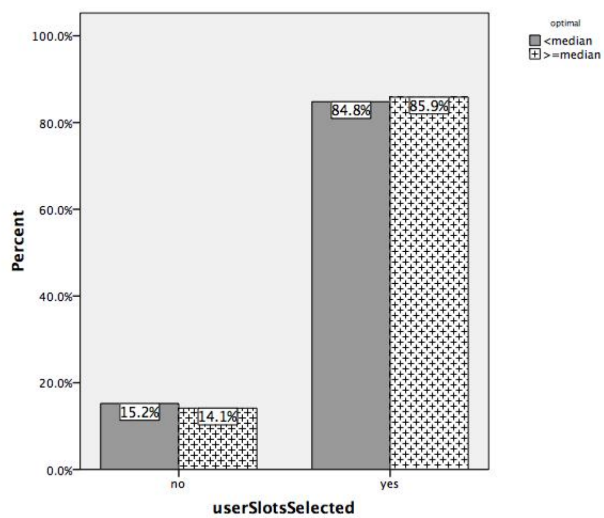


Figure 12: Percentages for “optimal” vs. lower rated/ “non-optimal” `userSlotsSelected`

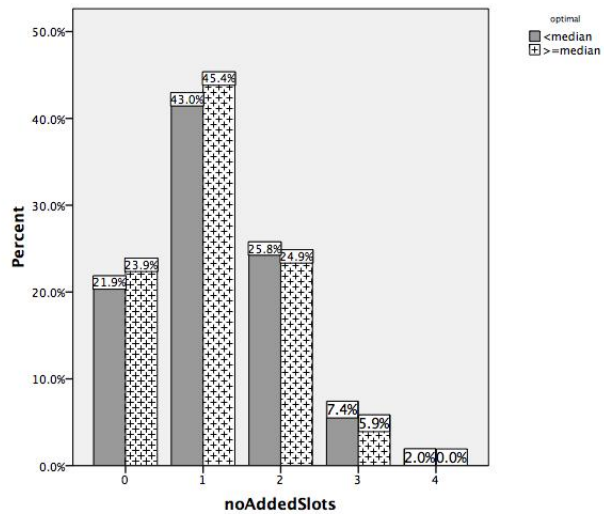


Figure 13: Percentages for “optimal” vs. lower rated/ “non-optimal” `noAddedSlots`

7. turn1: summary - turn2: summaryRecommend (4)
8. turn1: summaryCompare - turn2: Recommend (4)

Note that the top 4 NLG sequences are the same (summary followed by compare or recommend), but with the user taking an intervening turn in the top 2 cases.

4.3 Learning highly-rated NLG strategies

4.3.1 Method

We now learn/describe NLG strategies in context, using similar techniques to those we successfully applied to learning wizards’ multimodal dialogue strategies [Rieser and Lemon, 2009a]. We apply feature engineering methods, such as feature ranking and selection, in order to determine the context features which influence the wizards’ decisions. We then apply supervised learning (classification) on the data in order to discover the general patterns/ behaviour which generated the data. The outcome variable we are trying to predict is the NLG strategy {recommend, compare, compareRecommend, summary, summaryCompare, summaryCompareRecommend, summaryRecommend}. We compare the model learned on the low ranked data set (n=263) against the model learned on the highest ranked part of the data (n=205), i.e. what we define to be “optimal”.

4.3.2 Features

We first describe the dialogue context as a set of 15 (possible) runtime features (see Appendix A for feature descriptions):

1. turnNo (numeric)
2. db (numeric)
3. cuisineBinary {0,1}
4. cuisineNotBinary {0,1}
5. foodBinary {0,1}
6. foodNotBinary {0,1}
7. locationBinary {0,1}
8. locationNotBinary {0,1}
9. priceBinary {0,1}
10. priceNotBinary {0,1}
11. generalNoiseCondition {nonoise,noise}
12. noPrevUserSlots numeric
13. previousUserAct {addInfo,requestMoreInfo,silence,askRepeat}
14. elapsedTime (HH:mm:ss)
15. prevNLG {recommend,compare,compareRecommend,summary,summaryCompare, WithUMByAttributeRecommend, summaryCompareRecommend,summaryRecommend}

Note that we choose the features to be as domain-independent as possible, e.g. using binary values to represent whether a slot was filled or not instead of the actual values, [11] and number of previous user slots filled instead of the actual slot values.

[11] We have also experimented with using the actual slot values for learning, which slightly increases the accuracy for our prediction models at the expense of being tailored to this specific domain.

4.3.3 Feature ranking and feature selection

After defining the set of possible features, we apply feature engineering methods in order to determine which context features influenced the wizards' decisions. We also investigate differences between the lower ranked data set and the high ranked/'optimal' part of the data.

We first apply feature ranking methods, which evaluate the predictive power of each feature on its own with respect to the outcome variable. We apply WEKA's implementation of Information Gain Attribute Selection, which evaluates the worth of a feature by measuring the information gain with respect to the class ($InfoGain(Class, Feature) = H(Class) - H(Class|Feature)$) [Witten and Frank, 2005].

We then apply feature selection methods which address the problem of selecting an optimum subset of features that are most predictive of a given outcome. We apply WEKA's implementation of Correlation-based Feature Subset Selection (CFS) by [Hall, 2000]. This algorithm evaluates the worth of a subset of features by considering the individual predictive ability of each feature along with the degree of redundancy between them. Subsets of features that are highly correlated with the class while having low intercorrelation are preferred.

Table 6: Feature ranking for learning NLG strategies

non-optimal model	optimal model
weights - features	weights - features
0.23433 prevNLG	0.3571 db
0.13622 previousUserAct	0.2814 prevNLG
0.1046 generalNoiseCondition	0.1309 previousUserAct
0.10336 noPrevUserSlots	0.0962 priceBinary
0.09963 db	0.0806 generalNoiseCondition
0.08523 locationBinary	0.0656 cuisineBinary
0.05945 priceBinary	0.0514 locationBinary
0.0496 locationNotBinary	0.0282 cuisineNotBinary
0.03533 cuisineBinary	0.0198 priceNotBinary
0.02915 foodBinary	0.0179 foodBinary
0.01527 foodNotBinary	0.0154 locationNotBinary
0.01453 priceNotBinary	0.0111 foodNotBinary
0.00963 cuisineNotBinary	0 turnNo
0 turnNo	0 noPrevUserSlots
0 elapsedTime	0 elapsedTime

The results for feature ranking can be found in Table 6. Both models indicate that the previous NLG strategy (`prevNLG`) and the previous user act (`previousUserAct`) are most important for the wizards, which suggests that wizards take sequential NLG information (i.e. from previous dialogue turns) into consideration when deciding on the current NLG strategy.

Database hits (db) are considerably more important for the optimal model than for the non-optimal model. This confirms earlier findings for multimodal dialogues, where we found that number of database hits is an important feature to predict user rating [Rieser and Lemon, 2008]. However, the optimal policy pays less attention to the number of slots previously mentioned by the user (noPrevUserSlots).

Both models do not consider dialogue length measured either in turns (turnNo) or duration in time (elapsedTime) to be important. This indicates that wizards do not seem pay attention to the length of the dialogues. In contrast, this feature is considered to be important for Dialogue Management [Henderson et al., 2008].

Table 7: CFS feature selection for learning NLG strategies

non-optimal model	optimal model
db, generalNoiseCondition, noPrevUserSlots, previousUserAct, prevNLG	db, foodBinary, priceBinary, prevNLG

The results for CFS feature selection can be found in Table 7. The features db and prevNLG get selected for both models. The previous NLG strategy being included, indicates again that the wizards paid attention to the NLG sequences they generated. Furthermore, the status of the slots "food type" and "price range" are important for learning a model from the high ranked data.

4.3.4 Results and interpretation

We now learn predictive models for each data set (non-optimal vs. optimal). We use 2 feature sets for each data set: with and without CFS feature selection. We use 2 different Machine Learners, both producing a set of rules which can be interpreted and (later-on) implemented in a NLG system. We use the rule induction algorithm JRip, the WEKA implementation of [Cohen, 1995]'s "Repeated Incremental Pruning to Produce Error Reduction" (RIPPER). And the decision tree J4.8 classifier (WEKA's implementation of the C4.5 system [Quinlan, 1993]). We train and test the models using 10-fold cross validation and compare the accuracy of models against a majority baseline (i.e. always predicting summary in this case) using a paired t-test with correction.

The results are displayed in Table 8. We find an 6-12% improvement in accuracy over the majority baseline using Supervised Learning techniques. Furthermore, feature selection techniques improve prediction accuracy, which confirms previous results for Dialogue Management [Rieser and Lemon, 2006b].

The best performing model is learned by JRip on the feature set pre-selected by CFS for both data sets with 41.34% accuracy for the non-optimal data set and 43.19% accuracy for the high ranked data set. The decision tree algorithm J48 reaches higher accuracy, however it has a larger

Table 8: Accuracy for predicting NLG strategy on full data set and highly rated ("optimal") data.

Dataset	majority	rules.JRip	trees.J48
optimal	34.65±1.47	40.97±9.29	41.52±9.24 *
optimal-CFS	34.65±1.47	**43.19**±6.57 ***	44.03±9.72 **
non-optimal	29.28±1.73	40.54±7.68 ***	40.29±8.10 ***
non-optimal-CFS	29.28±1.73	**41.34**±7.82 ***	39.88±7.66 ***

*** denotes $p < .001$, ** denotes $p < .01$,* denotes $p < .05$.

standard deviation.

```
IF (generalNoiseCondition = nonoise) and (prevNLG = summary) and (db >= 6):
   THEN nlgStrategy=compare;
IF (db = 2):
   THEN nlgStrategy=compare;
IF (generalNoiseCondition = noise) and (prevNLG = 0.0) and (noPrevUserSlots >= 2)
  and (previousUserAct = addInfo) and (3<=db <= 9):
   THEN nlgStrategy=summaryRecommend;
IF (1<=db <= 3)
   THEN nlgStrategy= Recommend ;
IF (generalNoiseCondition = nonoise) and (previousUserAct = requestMoreInfo):
   THEN nlgStrategy= Recommend ;
IF (previousUserAct = silence) and (prevNLG = summaryRecommend):
   THEN nlgStrategy= Recommend ;
ELSE nlgStrategy=summary ;
```

Figure 14: Reformulation of the rules learned by JRip for the non-optimal model

The error analysis shows that most of the errors are due to misclassification as `summary` for both of the models, which is the default rule in both cases. The reformulated set rule set as learned by JRip on the CFS data for the general model is displayed in Figure 14, and for the optimal model in Figure 15.

The prediction model for the optimal and the non-optimal data set differ with respect to the following aspects: Firstly, different features are used in their rule sets. The non-optimal model's parameters are `generalNoiseCondition`, `previousUserAct`, `db`, `noPrevUserSlots` and `prevNLG`. Whereas the optimal model only models variation according to `db` and `prevNLG` (which are a subset of the non-optimal model's features) and `priceBinary`.

The prediction model for the non-optimal data set generates 4 of the 7 possible strategies, which are mostly single strategies (see Figure 14). The learned model fails to generate the combined NLG strategies `summaryCompare`, `summaryCompareRecommend` and `compareRecommend`. However, these 3 NLG strategies only account for about 10.6% of the data (see Figure 11.).

The prediction model for the optimal data set generates 3 of the 7 possible strategies. It does not

```
IF (db <= 9) and (prevNLG = summary) and (priceBinary = 1):
   THEN nlgStrategy=compare;
IF (db <= 1):
   THEN nlgStrategy= Recommend;
IF (prevNLG = summaryRecommend) and (db >= 10):
   THEN nlgStrategy= Recommend;
ELSE nlgStrategy=summary;
```

Figure 15: Reformulation of the rules learned by JRip for the optimal model

generate any of the combined NLG strategies, which account for 28.1% of the optimal data set (see Figure 11).

We conclude that there exist good and bad NLG strategies by human wizards (in terms of user ratings). The NLG strategies in the high ranked and the low-ranked/ non-optimal data set are different, and we discover them using Supervised Machine Learning techniques. They also depend on different context features, such as sequential NLG information.

5 Learning attribute selection policies

We follow the NLG model for Information Presentation strategies proposed by [Rieser and Lemon, 2009b], which suggests an hierarchical model for NLG strategy selection and attribute selection , i.e. the attributes selected will be dependent on the NLG strategy chosen.

For learning wizard NLG behaviour in context, the chosen NLG strategy thereby becomes a feature for attribute selection. Furthermore, we represent attribute selection itself as an hierarchical model. We are interested in two different aspects of attribute selection (also see Section 3.2.1): We first investigate whether human wizards include the attributes last provided by the user. Repeating the user's attributes serves the function of implicitly confirming the user's request. We also want to know how many additional attributes the wizards present to the user. Adding new attributes serves the function of providing an overview of the available options [Demberg et al., 2009]. These NLG actions therefore overlap with Dialogue Management actions.

We use the same features as presented in Section 4.3, and we add previous NLG decisions for every completed prediction step in order to implement an hierarchical decision model. Also see Figure 10.

5.1 User attributes selected

We first investigate whether the wizards select the attributes mentioned previously by the user for the non-optimal data set and the optimal data set. The frequencies are about equal for both data sets, as noted in Section 4.2.1, see Figure 12. This indicates, again, that context dependency is important in making good NLG choices.

5.1.1 Feature ranking and feature selection

We now examine which features influenced the wizards' decisions on whether to select the user's attributes or not. We use the same feature ranking and selection techniques as for NLG strategy prediction, see Section 4.3.3, plus we add the current NLG strategy as a feature to reflect the hierarchical approach (15 original features plus previously predicted `nlgStrategy` as a new feature, 16 features in total, see Figure 10).

The results for feature ranking can be found in Table 9. The current NLG strategy (`nlgStrategy`), the previous NLG strategy (`prevNLG`), the turn number (`turnNo`), and the previous user act (`previousUserAct`) are in general most important for the wizards' decisions. These features again indicate that the wizards' decisions depend on the dialogue and NLG sequence (`prevNLG`,`previousUserAct` However, attribute selection also depends on higher level NLG decisions (`nlgStrategy`), which suggests that NLG strategy and attribute selection can be jointly modelled as a hierarchical model.

For the optimal data set, `elapsedTime` and `noPrevUserSlots` are more important than in the non-optimal part of the data. This indicates that good attribute selection techniques take the amount of time as well as the number of generated attributes into account.

Table 9: Feature ranking for user attributes selected

non-optimal model	optimal model
weights - features	weights - features
0.14028541 nlgStrategy	0.117286 nlgStrategy
0.10167797 prevNLG	0.115927 prevNLG
0.09648305 previousUserAct	0.096447 turnNo
0.06288032 turnNo	0.061996 priceBinary
0.0496877 locationBinary	0.053168 previousUserAct
0.02322596 generalNoiseCondition	0.051543 generalNoiseCondition
0.02272544 priceBinary	0.037619 noPrevUserSlots
0.01128875 foodBinary	0.035053 elapsedTime
0.01128875 foodNotBinary	0.025813 locationBinary
0.00742303 cuisineNotBinary	0.009733 cuisineBinary
0.00504521 priceNotBinary	0.005439 priceNotBinary
0.00001684 cuisineBinary	0.003246 foodBinary
0.00000202 locationNotBinary	0.001891 locationNotBinary
0 elapsedTime	0.001076 foodNotBinary
0 noPrevUserSlots	0.000109 cuisineNotBinary
0 db	0 db

It is interesting to note that the number of database items doesn't seem to influence whether the user's attributes get selected by the wizards (zero weights), in contrast to NLG strategy selection where `db` was amongst the most important features.

Table 10: CFS feature selection for user attributes selected

non-optimal model	optimal model
`locationBinary, foodBinary, previousUserAct, prevNLG, nlgStrategy`	`turnNo, priceBinary, generalNoiseCondition, noPrevUserSlots, previousUserAct, elapsedTime, prevNLG, nlgStrategy`

The results for feature selection can be found in Table 10. The features `previousUserAct`, `prevNLG`, and `nlgStrategy`) overlap between the optimal and the non-optimal data set, confirming previous results indicating that wizard NLG strategy and attribute selection is a sequential, hierarchical model.

In addition to those, the features selected for the non-optimal data set also contains the `location` and `foodQuality` slots, whereas for the optimal data set `turnNo`, `generalNoiseCondition`, `noPrevUserSlots`, and `elapsedTime` are getting chosen. Again, this indicates that good attribute selection techniques depend on dialogue length and total number of generated attributes. Note that in contrast to NLG strategy selection, dialogue length plays an important role. Whereas the number of database items retrieved is not getting selected (as already indicated by attribute ranking).

5.1.2 Results and interpretation

We do not find any significant improvement for this binary decision problem, see results in Table 11, as the majority baseline is already rather high (over 84% "`yes`") and there seems to be not enough data to learn the exceptions.

Table 11: Accuracy for predicting whether user attributes are getting selected.

Dataset	majority	rules.JRip	trees.J48
optimal	85.86±1.43	82.88±5.33	85.61±2.30
optimal-CFS	85.86±1.43	83.40±4.94	**85.86**±1.43
non-optimal	84.79±0.26	84.15±4.67	85.17±4.14
non-optimal-CFS	84.79±0.26	**87.76**±5.10	85.73±4.01

However, it is still interesting to analyse what was learned in order to describe the wizards' attribute selection strategies. We investigate the rules learned by the best performing models for each data set (marked bold face in Table 11).

The reformulated set rule set as learned by JRip on the CFS data for the non-optimal model is shown in Figure 16. The best performing model for the optimal data set always predicts the majority class (“`yes`”) as a default.

The default option for the non-optimal model is also to the include the user’s attributes in generation (which reflects the fact that this was the preselected default option on the GUI). The only exception where the non-optimal model learned *not* to present the user’s attributes if for the case where the user has mentioned more than 2 attributes in the previous turn, and the selected NLG strategy is COMPARE.

In sum, based on this data there seems to be no major differences between low-ranked and high-ranked NLG behaviour for this task since the wizards mainly choose the option of always including the user’s attributes. This might be due to the fact that the user’s attributes were pre-selected on the GUI. We conclude that design decisions for the WoZ GUI are critical, as they significantly influence the wizards’ choices. We believe that a larger data collection on a different task may show significant improvements here, and the methods we have shown could be used to discover good NLG decisions for this task. Based on this data, the safest bet is to always include the user’s attributes in generation if one had to implement such a model, since none of the prediction models outperforms the majority baseline.

Note that simulation-based RL can discover unseen events which are not in the data. Therefore, RL has the potential to discover an optimal NLG decision for whether to include the user attributes or not. We can’t learn this decision using Supervised Learning on this data set, since the wizards mainly choose the option of always including the user’s attributes.

```
IF (nlgStrategy = compare) and (priceBinary = 1) and (noPrevUserSlots <= 2):
   THEN userSlotsSelected=no;
ELSE userSlotsSelected=yes;
```

Figure 16: Reformulation of the rules learned by JRip for the non-optimal model

5.2 Number of added attributes

Choosing the number of additional attributes (i.e. attributes not previously specified by the user) is the last decision step in our hierarchical NLG model. We don’t consider attribute type here, as our data has too few data points for Machine Learning techniques, see Figure 17.

In Section 4.2.1 we found that the frequencies are about equal for both data sets. One of the main differences is that wizards never added 4 attributes in the high ranked data set, see Figure 13.

Note that we treat this problem as a discrete classification task rather than numeric prediction, as the outcome variable is not continuous (e.g. you can’t present 3.5 features).

5.2.1 Feature ranking and feature selection

We first examine which context features influence the wizards’ decisions on how many new attributes to add. We use the same features as for user attribute selection (see Section 5.1.1) but

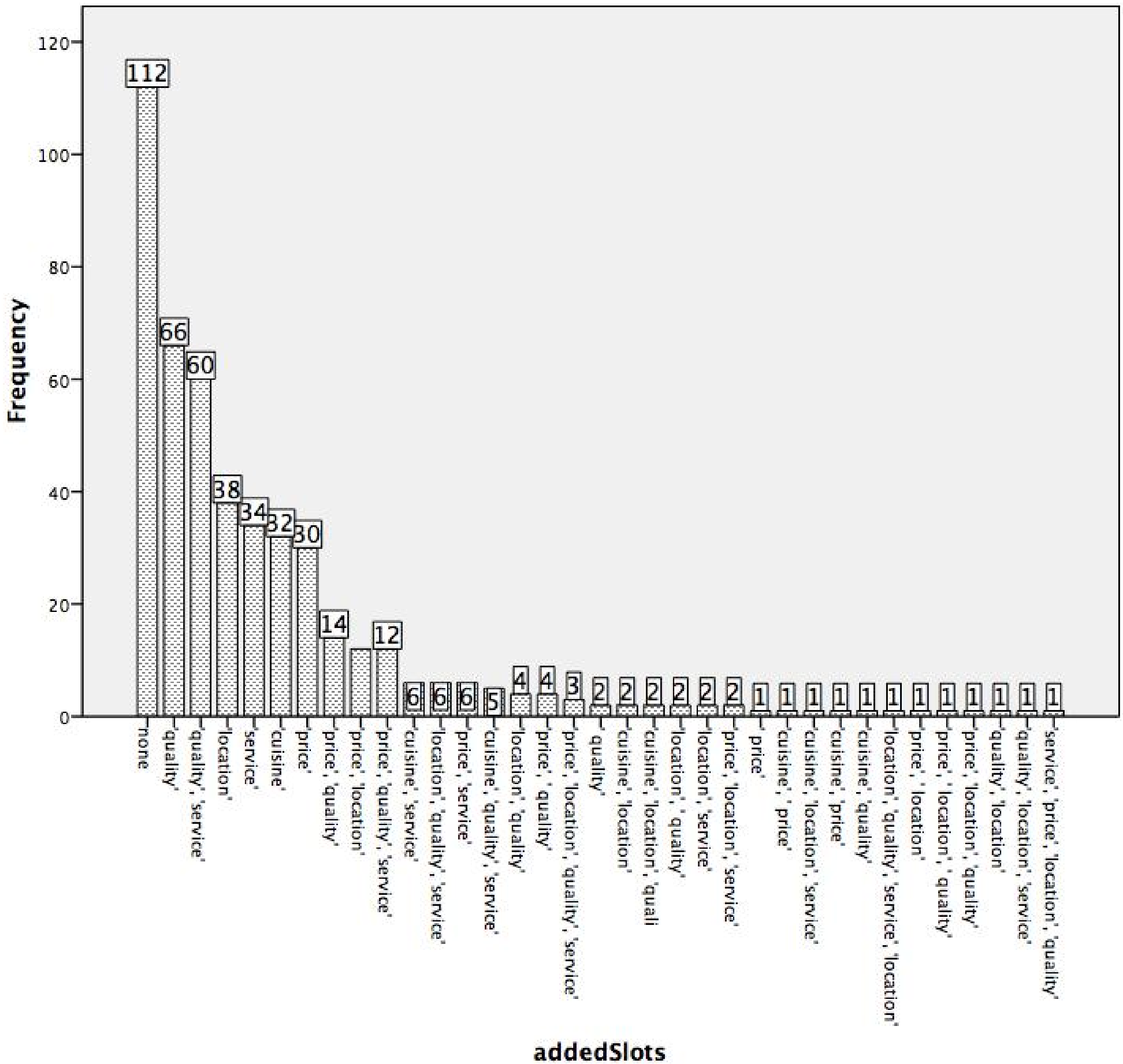


Figure 17: Frequencies of added slot types for the whole data set

this time we add `userAttrSelected` as an predictive feature to reflect the hierarchical approach (16 previous features plus `userSlotsSelected`; 17 features in total).

general model	optimal model
weights - features	weights - features
0.1729 nlgStrategy	0.12756 nlgStrategy
0.1083 prevNLG	0.09675 prevNLG
0.0806 generalNoiseCondition	0.08853 elapsedTime
0.0671 previousUserAct	0.0469 previousUserAct
0.0669 locationBinary	0.03481 userSlotsSelected
0.0638 foodNotBinary	0.03381 generalNoiseCondition
0.0623 priceNotBinary	0.03128 locationNotBinary
0.0562 cuisineNotBinary	0.03004 priceBinary
0.0494 locationNotBinary	0.01267 foodBinary
0.0388 userSlotsSelected	0.01163 cuisineBinary
0.0364 cuisineBinary	0.01059 locationBinary
0.0251 foodBinary	0.00558 foodNotBinary
0.0178 priceBinary	0.00456 cuisineNotBinary
0 db	0.00402 priceNotBinary
0 turnNo	0 turnNo
0 noPrevUserSlots	0 noPrevUserSlots
0 elapsedTime	0 db

Table 12: Feature ranking for number of added slots

The results for feature ranking are displayed in Table 12. The features `nlgStrategy`, `prevNLG`, and `previousUserAct` are ranked high for both models. Again, these results strengthen the hypothesis that human wizard NLG behaviour can be modelled as an hierarchical and sequential decision process.

The major difference we observe in the feature ranking for the optimal and non-optimal model is that `elapsedTime` is ranked high for the optimal model, but not for the non-optimal model. In general, wizards don't seem to pay attention to `turnNo`, `db`, and `noPrevUserSlots` for selecting number of attributes, as these features have zero weights for both of the models.

The results for feature selection are displayed in Table 13. Five features are selected for both of the data sets (`locationNotBinary`,`generalNoiseCondition`, `previousUserAct`, `prevNLG`, `nlgStrategy`). It is interesting to note that this time slots being filled matter more for the non-optimal model, whereas for the optimal model the `elapsedTime` and `userAttrSelected` are getting chosen. These results, again, favour an hierarchical (`nlgStrategy`) and sequential (`previousUserAct`, `prevNLG`) model, especially since the optimal model also takes the outcome of the previous decision process in our model (`userAttrSelected`) into account.

non-optimal model	optimal model
`locationNotBinary priceNotBinary, foodNotBinary, locationBinary, generalNoiseCondition, previousUserAct, prevNLG, nlgStrategy`	`locationNotBinary, generalNoiseCondition, previousUserAct, elapsedTime, prevNLG, nlgStrategy, userAttrSelected`

Table 13: CFS feature selection for number of added slots

5.2.2 Results and interpretation

Again, none of the models beats the majority baseline (where here the majority baseline is to add 1 additional attribute), see Table 14.

Dataset	majority	rules.JRip	tress.J48
optimal	45.36±1.89	48.34±7.46	41.11±8.80
optimal-CFS	45.36±1.89	**48.95**±6.64	43.13±8.64
non-optimal	41.84±0.72	42.67±4.86	39.44±8.47
non-optimal-CFS	41.84±0.72	**45.80**±6.05	41.77±7.71

Table 14: Accuracy for predicting number of added attributes

We investigate the best performing models (marked bold face in Table 14). The reformulated set rule set as learned by JRip is displayed in Figure 18 for the non-optimal part of the data, and in Figure 19 for the optimal part. The default rule for both models is adding “1” attribute. The model trained on the non-optimal data set generates three out of four possible cases, where `noAddedAttr`=“3” is missing (which accounts for 7.2% of the data). The model trained on the optimal data set generates 2 out of 3 cases (`noAddedAttr`=“4” never occurs in the optimal data). The optimal model never adds more than 2 attributes. It doesn’t add any new attributes if the dialogue gets longer. The missing case (noAddedAttr=“3”) only occurs in 5.9% of the data.

5.3 Summary

In Section 4 and 5 we showed differences between (what we defined to be) optimal and non-optimal wizard behaviour using feature ranking, feature selection, and Supervised Learning techniques. We found that good NLG strategy selection depends on the number of database hits and the previous NLG strategy generated, which indicates that NLG strategy selection is a sequential decision process (over several turns). Good attribute selection techniques, in contrast, depend on dialogue length and the total number of attributes generated. These differences between the

```
IF (nlgStrategy=summaryRecommend) and (prevNLG=summary) and (previousUserAct=requestMoreInfo):
   THEN noAddedSlots=4;
IF (generalNoiseCondition = nonoise) and (locationNotBinary = 1):
   THEN noAddedSlots=0;
IF (generalNoiseCondition=nonoise) and (previousUserAct=addInfo) and (nlgStrategy  Recommend):
   THEN noAddedSlots=0;
IF (nlgStrategy = compare) and (previousUserAct = requestMoreInfo):
   THEN noAddedSlots=2;
IF (nlgStrategy = summaryRecommend) and (prevNLG = summary):
   THEN noAddedSlots=2;
ELSE noAddedSlots=1;
```

Figure 18: Reformulation of the rules learned by JRip for the non-optimal model

```
IF (elapsedTime >= 00:03:00) and (nlgStrategy =  Recommend) and (previousUserAct = addInfo)
 and (locationNotBinary = 0) and (generalNoiseCondition = nonoise):
    THEN noAddedSlots=0;
IF (elapsedTime < 00:03:00) and (locationNotBinary = 1) and (prevNLG = summary):
   THEN noAddedSlots=2;
ELSE noAddedSlots=1;
```

Figure 19: Reformulation of the rules learned by JRip for the optimal model

optimal and the non-optimal part of the data confirm our hypothesis that good NLG strategies are context dependent.

Furthermore, results from feature ranking and selection provide evidence for our hypothesis that human wizards' NLG strategy and attribute selection can be modelled a sequential, hierarchical decision process, where NLG information is shared over several turns, and where higher level NLG decisions feed into the decision model for lower level NLG actions.

6 Discussion

In this section we discuss the models and results we obtained for investigating context-dependent optimal wizard NLG behaviour, for Information Presentation tasks.

6.1 Model accuracy

First of all, the accuracy of the Supervised Learning models is not very satisfying. Our data set is relatively small for statistical methods, and for some cases we don't have enough data. This is why some of our models over-predict the majority/ most frequent class in the data.

In addition, our data set has diverse patterns of wizard behaviour. We show in Section 3.4 that wizards behave significantly differently. Diverse behaviour is harder to detect for Supervised Learning methods, however for Reinforcement Learning we need data that explores many different ways to behave. We therefore suggest that the WoZ corpus we collected is a good starting point to "bootstrap" optimal policies using RL, following [Rieser and Lemon, 2008].

Also note that comparing against the majority baseline is more challenging than if we would have compared against a random baseline. The accuracy of the random baseline for NLG strategy prediction is 14.3%; for user slots added 50%; and for number of added attributes 25% for the optimal data set and 20% for the non-optimal data. Compared to the majority baseline, the accuracy of those random models is much lower and they would be easily outperformed by our Supervised Learning models.

In order to account for the low accuracy of our models we also take a descriptive approach and report on frequencies of NLG actions and sequences of NLG actions in a dialogue.

Also note that these numbers are affected by our definition of "optimal" (which is a consequence of the size of our data set).

6.2 GUI design

One of the lessons learnt is that WoZ GUI design is critical for the quality of the data. GUI design influences and limits the wizards' behaviour (as pointed out several times in this report). Alternatively, one could have a free talking wizards, as in [Rieser et al., 2005]. However, this method heavily relies on (manual) post-annotation and transcription work. Furthermore, the wizards would be less restricted in their choices, which is most likely to result in even more diverse behaviour (which SL models have problems to capture).

6.3 Influence of NLG surface realisation

An interesting question to ask for future work is whether the wizards' decisions were influenced by features of the lower level NLG prompt generation. As described in Section 2.1.1, the NLG prompt realisation includes some random variations of surface forms, such as sentence aggregation, referring expressions, lexical items, etc. Note that the overall quality of the generated output was high with respect to grammatical correctness.

The data gathered in this WoZ experiment is not suitable to answer this question, as this study was tailored to investigate higher level NLG content planning actions (i.e. the general IP strategy and the number of attributes selected). The wizards were not able to select any of the lower level features (sentence aggregation, referring expressions, lexical items etc), nor did they edit the automatically generated text (which was possible using the Wizard GUI).

We did observe that wizards experimented with different higher level NLG choices (e.g. selecting different strategies, number of attributes).

In 63% of cases the wizard changed between NLG actions before submitting the output to the final TTS panel. The majority of these changes are due to the fact that the user's attributes were pre-selected by default on the GUI and wizards changed that default in 58% of cases. Furthermore, wizards changed their top-level NLG strategy decisions in 5.1% of cases. No default top-level NLG strategy was pre-selected on the GUI – so in 94.9% of cases wizards chose an NLG strategy and stuck with it for that turn

However, there could be several reasons for this behaviour, and it cannot only be explained by features of lower level / surface generation output. A different study would be needed to investigate the influence of variations in the surface generator on Wizard decisions. For example, by allowing the wizards to ask for a new prompt to be generated (for the same content plan) if they don't like the current one generated.

6.4 "Optimal" human behaviour

In previous work we showed that humans perform non-optimally for choosing multimodal dialogue strategies [Rieser and Lemon, 2008]. However, work by [Yakushijn and Jacobs, 2009] shows that humans can be "trained" to perform close to optimally on a perceptual matching task. In order to investigate learning effects of wizards we compared the behaviour of each wizard in the first 3 sessions against the last 3 sessions. We found that wizards explore more different strategies in the beginning and show less variation for the last 3 sessions. However, user rankings are not significantly different. For one wizard, the user ranking even gets significantly ($p < .01$) worse. We therefore conclude that for this task, wizards did not become trained to perform optimally. In contrast to the perceptual matching task by [Yakushijn and Jacobs, 2009], our NLG task does not have a well-defined goal, i.e. the goal how to satisfy the user is "hidden". Furthermore, the "environment" the wizard operates in is stochastic, i.e. users behave differently. In [Yakushijn and Jacobs, 2009] the environment operates by an underlying causal relationship, which can be discovered by the subject during the training phase. Once the subject learns those causal relationships, it can perform near optimally on this task. We hypothesise that RL-based systems maybe able to perform better than humans on this task because they can learn to interact in a stochastic environment in an optimal way. We will test this hypothesis in future work.

7 Conclusion

In this report we have presented a method to analyse and describe human optimal wizard NLG behaviour in context. We first presented a Wizard-of-Oz experiment for data collection and then analysed the corpus.

We then used Machine Learning techniques to investigate human wizard NLG behaviour in context.

These methods allow us to discover differences between low-rated NLG behaviour and highly-rated NLG decisions. These differences between the low-rated and the highly-rated part of the data confirm our hypothesis that good NLG strategies are context dependent. We show that good NLG strategies are context-adaptive: wizards adapt to different context features and follow different strategies for highly-rated utterances, than in the case of the low-rated NLG behaviour. We discuss these differences for NLG Information Presentation strategy selection and attribute selection.

We also presented a method for learning NLG strategy selection and attribute selection in an hierarchical manner. Results from feature ranking and selection provide evidence for our hypothesis that human wizards' NLG strategy and attribute selection can be modelled a sequential, hierarchical decision process, where NLG information is shared over several turns, and where higher level NLG decisions feed into the decision model for lower level NLG actions in a top-down manner. Note that in previous work, [Rieser and Lemon, 2009b], we presented a Reinforcement Learning-based NLG decision model which can also automatically adapt bottom-up to lower level NLG decisions, e.g. from a surface realiser.

In future work we will also use these learned models of human wizard behaviour as a baseline for comparison with NLG policies learned using Reinforcement Learning.

The data-collection aspects of this work are to be further described in deliverable D6.1, and the corpus collected will be released publicly as part of the project's Annotated Data Archive, in deliverable D6.5.

References

[Bellman, 1957] Bellman, R. (1957). *Dynamic Programming*. Princeton University Press.

[Clark and Schaefer, 1989] Clark, H. and Schaefer, E. F. (1989). Contributing to discourse. *Cognitive Science*, 2(13):259–294.

[Cohen, 1995] Cohen, W. W. (1995). Fast effective rule induction. In *Proceedings of the 12th International Conference on Machine Learning (ICML)*.

[Crook and Lemon, 2009] Crook, P. A. and Lemon, O. (2009). Accurate probability estimation of hypothesised user acts for pomdp approaches to dialogue management. In *12th Annual Research Colloquium of the special-interest group for computational linguistics in the UK and Ireland (CLUKI)*.

[Dahlbäck et al., 1993] Dahlbäck, N., Jönsson, A., and Ahrenberg, L. (1993). Wizard of Oz-studies – why and how. In *Proc. of the Workshop on Intelligent User Interfaces*.

[Demberg et al., 2009] Demberg, V., Winterboer, A., and Moore, J. (2009). A strategy for information presentation in spoken dialogue systems. *submitted*.

[Fraser and Gilbert, 1991] Fraser, N. M. and Gilbert, G. N. (1991). Simulating speech systems. *Computer Speech and Language*, 5:81–99.

[Hall, 2000] Hall, M. (2000). Correlation-based feature selection for discrete and numeric class machine learning. In *Proc. of the 17th International Conference on Machine Learning (ICML)*.

[Halliday, 1985] Halliday, M. A. (1985). *Introduction to Functional Grammar*. Edward Arnold.

[Henderson et al., 2008] Henderson, J., Lemon, O., and Georgila, K. (2008). Hybrid reinforcement / supervised learning of dialogue policies from fixed datasets. *Computational Linguistics*, 34:4.

[Janarthanam and Lemon, 2009] Janarthanam, S. and Lemon, O. (2009). A Wizard-of-Oz Environment to study Referring Expression Generation in a Situated Spoken Dialogue Task. In *Proceedings of ENLG*.

[Koller and Petrick, 2008] Koller, A. and Petrick, R. (2008). Experiences with planning for natural language generation. In *ICAPS*.

[Koller and Stone, 2007] Koller, A. and Stone, M. (2007). Sentence generation as planning. In *Proceedings of ACL*.

[Lemon, 2008] Lemon, O. (2008). Adaptive Natural Language Generation in Dialogue using Reinforcement Learning. In *Proceedings of SEMdial*.

[Lemon et al., 2006] Lemon, O., Georgila, K., and Henderson, J. (2006). Evaluating Effectiveness and Portability of Reinforcement Learned Dialogue Strategies with real users: the TALK TownInfo Evaluation. In *IEEE/ACL Spoken Language Technology*.

[Liu et al., 2009] Liu, X., Rieser, V., and Lemon, O. (2009). A wizard-of-oz interface to study information presentation strategies for spoken dialogue systems. In *Proc. of the 1st International Workshop on Spoken Dialogue Systems*.

[Ng and Russell, 2000] Ng, A. Y. and Russell, S. (2000). Algorithms for inverse Reinforcement Learning. In *Proc. of the 17th International Conference on Machine Learning (ICML)*, pages 663–670. Morgan Kaufmann, San Francisco, CA.

[Polifroni and Walker, 2008] Polifroni, J. and Walker, M. (2008). Intensional Summaries as Cooperative Responses in Dialogue Automation and Evaluation. In *Proceedings of ACL*.

[Quinlan, 1993] Quinlan, R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

[Reiter and Dale, 2000] Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*. CUP.

[Rieser et al., 2005] Rieser, V., Kruijff-Korbayová, I., and Lemon, O. (2005). A corpus collection and annotation framework for learning multimodal clarification strategies. In *Proc. of the 6th SIGdial Workshop on Discourse and Dialogue*.

[Rieser and Lemon, 2006a] Rieser, V. and Lemon, O. (2006a). Cluster-based user simulations for learning dialogue strategies. In *Proc. of the 9th International Conference of Spoken Language Processing (Interspeech/ICSLP)*.

[Rieser and Lemon, 2006b] Rieser, V. and Lemon, O. (2006b). Using machine learning to explore human multimodal clarification strategies. In *Proceedings of ACL (Main conference poster sessions)*, pages 659–666.

[Rieser and Lemon, 2008] Rieser, V. and Lemon, O. (2008). Learning Effective Multimodal Dialogue Strategies from Wizard-of-Oz data: Bootstrapping and Evaluation. In *Proc. of ACL*.

[Rieser and Lemon, 2009a] Rieser, V. and Lemon, O. (2009a). Learning human multimodal dialogue strategies. *Journal of Natural Language Engineering (to appear)*.

[Rieser and Lemon, 2009b] Rieser, V. and Lemon, O. (2009b). Natural Language Generation as Planning Under Uncertainty for Spoken Dialogue Systems. In *Proc. of EACL*.

[Rieser and Lemon, subm] Rieser, V. and Lemon, O. (subm.). Natural Language Generation as Planning Under Uncertainty for Spoken Dialogue Systems. In *Empirical Methods in Natural Language Generation*.

[Stent et al., 2004] Stent, A., Prasad, R., and Walker., M. (2004). Trainable sentence planning for complex information presentation in spoken dialog systems. In *Association for Computational Linguistics*.

[Stuttle et al., 2004] Stuttle, M., Williams, J., and Young, S. (2004). A framework for dialogue data collection with a simulated ASR channel. In *Proc. of the International Conference of Spoken Language Processing (Interspeech/ICSLP)*.

[van Deemter, 2009] van Deemter, K. (2009). What game theory can do for nlg: the case of vague language. In *12th European Workshop on Natural Language Generation (ENLG)*.

[Walker et al., 2001] Walker, M., Passonneau, R., and Boland, J. (2001). Quantitative and qualitative evaluation of DARPA Communicator spoken dialogue systems. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

[Walker et al., 2000] Walker, M. A., Kamm, C. A., and Litman., D. J. (2000). Towards developing general models of usability with PARADISE. *Natural Language Engineering*, 6(3).

[Whittaker et al., 2002] Whittaker, S., Walker, M., and Moore, J. (2002). Fish or Fowl: A Wizard of Oz evaluation of dialogue strategies in the restaurant domain. In *Proc. of the International Conference on Language Resources and Evaluation (LREC)*.

[Winterboer and Moore, 2007] Winterboer, A. and Moore, J. D. (2007). Evaluating information presentation strategies for spoken recommendations. In *Proc. of the ACM Conference on Recommender Systems*.

[Witten and Frank, 2005] Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques (2nd Edition)*. Morgan Kaufmann.

[Yakushijn and Jacobs, 2009] Yakushijn, R. and Jacobs, R. A. (2009). Near-optimal learning on a perceptual matching task. *(submitted)*.

A Features

- General information
  - **wizard** : w1, w2
  - **userName** : from the pre-questionnaire
  - **userGender** : from the pre-questionnaire (f,m)
  - **userAge** : from the pre-questionnaire (4 age groups)
  - **userNative** : from the pre-questionnaire (y,n)
  - **userEatOut** : how often the user eats out in restaurants, from the pre-questionnaire (4 values)
  - **userSystemUsedBefore**: if the user has used a SDS before, and if so, whether he found it convenient to use; from the pre-questionnaire
  - **taskSet** : task set used (id 0-9)
  - **dialogueID** : unique ID for each dialogue (userName1-12)
- Turn information
  - **turnNo** : current turn number (numeric)
  - **currentTime**: current time stamp (in hh:mm)
  - **elapsedTime**: elapsed time so far (in hh:mm)
  - **finalUtterance** : utterance generated
- NLG related
  - **nlgStrategyRealisation** : NLG strategy realisation used, e.g. SummaryWithUM-CompareByItem; 0 if no NLG strategy was used, e.g. welcome prompt
  - **nlgStrategyCompressed** : NLG strategy summarised into broader categories, e.g. summariseCompareRecommend (14→7)
  - **previousNLG** : previous NLG strategy (compressed values) applied, e.g. summariseC-ompare; 0 if no NLG strategy was previously applied in this dialogue
  - **attributes** : number of attributes used (numeric)
  - **action1** : NLG action chosen to be generated first, e.g. SummaryWithUM
  - **action2** : NLG action chosen to be generated second, e.g. CompareByItem
  - **action** : NLG action chosen to be generated third, e.g. Recommend
  - **attributes1** : Attributes selected for the first NLG strategy, e.g. cuisine,service
  - **attributes2** : Attributes selected for the second NLG strategy
  - **attributes3** : Attributes selected for the third NLG strategy

- **selectedWozAtts** : set of all attributes selected by the wizard (list of strings)
- **userAttrSelected** : whether the wizard selected the 'default' slots provided by the user, in selectedWozAtts (binary)
- **addedSlots** : which attribute he added on to, in selectedWozAtts (list of strings)
- **noAddedSlots** : number of attributes added on top (numeric).

• Task-based/ Slot values

- **cuisineSlot** : value of the cuisine slot; "0" if empty
- **cuisineNotSlot** : whether the user negated the cuisine slot (0/1), e.g. "not Fish"
- **foodQualitySlot** : -"-
- **foodQualityNotSlot** : -"-
- **locationSlot** : -"-
- **locationNotSlot** : -"-
- **priceSlot** : -"-
- **priceNotSlot** : -"-
- **serviceSlot** : -"-
- **serviceNotSlot** : -"-
- **cuisineBinary** : whether cuisine is filled or not (binary) → domain independent
- **cuisineNotSlotBinary** : -"-
- **foodQualitySlotBinary** : -"-
- **foodQualityNotSlotBinary** : -"-
- **locationSlotBinary** : -"-
- **locationNotSlotBinary** : -"-
- **priceSlotBinary** : -"-
- **priceNotSlotBinary** : -"-
- **serviceSlotBinary** : -"-
- **serviceNotSlotBinary** : -"-
- **db** : number of database hits (numeric)
- **noMatches**: whether 0 DB results were retrieved (binary)

• Noise model

- **generalNoiseCondition** : whether the experiment was run under the noise condition (y/n)

- **noise** : how much noise was introduced in the course of this dialogue (accumulative feature)
- **Deletion** : whether the noise introduced is a Deletion (yes/no)
- **Insertion** : whether the noise introduced is an Insertion (yes/no)
- **Substitution** : whether the noise introduced is a Substitution (yes/no)
- **cuisineNoise** : -"-
- **cuisineNotNoise** : -"-
- **experimenterMessage** : message sent to the wizard, e.g. "final"
- **foodQualityNoise** : foodQuality slot after noise was introduced
- **foodQualityNotNoise** : -"-
- **locationNoise** : -"-
- **locationNotNoise** : -"-
- **priceNoise** : -"-
- **priceNotNoise** : -"-
- **serviceNoise** : -"-
- **serviceNotNoise** : -"-

- User reply
  - **userReply** : 7 Speech Act categories the user reply gets annotated with (addInfo, askRepeat, hangup, requestMore, select, silence, other)
  - **replySlots** : the slot(s) added in the following user reply
  - **previousUserAct** : user act from the previous turn, i.e. wizard reaction is following this user act.
  - **previousUserSlots** : user slots filled in the previous turns (i.e. all slots filled so far) NOTE: from the wizard's point of view, i.e. in noise condition this will be the noisy slots
- User ratings & objective measures
  - **question1** : The system's voice was easy to understand. (**TTS quality**)
  - **question2** : The way the system presented information was good. (**Information Presentation**)
  - **question3** : The system's utterances had the right length. (**Utterance length**)
  - **question4** : The system gave me a good overview of all the available options. (**Coverage**)
  - **question5** : The restaurant I chose was a good match for this task. (**task success**)

- **usersatisfaction** : normalised sum over questions in task questionnaire ($\frac{= question1 + question2 + question3 + question4 + question5}{5}$)
- **timeDelay** : user rating for final questionnaire ”The time delay of the system affected my ratings.”
- **TTSandTaskSuc** : user rating for final questionnaire ”I wasn’t able to choose a restaurant because I could not understand the system’s voice”
- **future** : user rating for final questionnaire ”I’d like to call the system in the future for restaurant information”
- **comments** : additional comments by the user from the final questionnaire
- **finalNoTurns** : total dialogue length in system turns (final turn)
- **totalDuration** : total dialogue duration in minutes (measured on final turn)

B User comments

- I would like to be more precise of finding the food quality of different dishes. (**w2, no-noise**)
- Since the system was talking instead of a human being, one can miss the human element in the voice. It didnt give me further information on anything. The voice was clear and was audible. The system sounded too mechanical and didnt have many options during the conversation. But the system had a good data on the all the restaraunts in Edinburgh. (**w2, noise**)
- Excellent service! Would be great if some minor problems were amended. Major issue was understanding the actual name of the restaurant, but could easily understand the location, price range and quality. Frustrating when the system needs to go back to the beginning or the query needs to be repeated. Sometimes the names of the restaurants are hard to understand. Also the system would not always remember my original preferences and would start the search from scratch. (**w2, noise**)
- The system's voice was easy to understand except when giving the names of many of the restaurants, which is probably the most important thing! (**w2, noise**)
- The system was very clear for all but one of the tasks. For one task it was not clear how to establish appropriate restaurants from the criteria given and that this had to be requested. (**w1, no-noise**)
- The system helps in narrowing down choices to the best quality and/or service, which are probably the higher priorities for most diners.however, information about specific types of food, e.g. haggis, spaghetti is limited to the type of cuisine the restaurant serves. the service can perhaps provide further contact details for the diner to find out if the specific dish is being served in the recommended restaurant. (**w1, no-noise**)
- better than other similar systems I have used. (**w1, no-noise**)
- Can the system give the address of the restaurant once you've chosen it? Or the telephone number to book a table? (**w1, noise**)
- On occasion, the system stopped talking and provided no prompts. It was difficult to know if it was checking something, or if you had to provide more specifications.Possibly, this would be easier to navigate if the system asked the user questions to specify the options, rather than the user providing the details. Alternatively, a skippable and simple introduction on how to phrase your request may be useful. (**w1, noise**)

C NLG sequences in the data

Sequence	Frequency	Percent	Cumulative Percent
summaryRecommend	23	11.1	11.1
summary-compare	18	8.7	19.7
summary-Recommend	14	6.7	26.4
Recommend	11	5.3	31.7
summaryCompare	11	5.3	37.0
summaryCompare-Recommend	9	4.3	41.3
summaryRecommend-Recommend	9	4.3	45.7
compare	8	3.8	49.5
summary-summaryRecommend	8	3.8	53.4
summaryCompare-compare	5	2.4	55.8
compare-compare	4	1.9	57.7
summary-compare-Recommend	4	1.9	59.6
summary-summary-summaryRecommend	4	1.9	61.5
compareRecommend	3	1.4	63.0
Recommend-Recommend	3	1.4	64.4
summary-summary-compare	3	1.4	65.9
compare-Recommend	2	1.0	66.8
summary	2	1.0	67.8
summary-compare-compare-compare	2	1.0	68.8
summary-summary	2	1.0	69.7
summary-summary-Recommend	2	1.0	70.7
summary-summary-summary-Recommend	2	1.0	71.6
summary-summary-summary-Recommend-Recommend	2	1.0	72.6
summary-summary-summaryCompare	2	1.0	73.6
summary-summaryCompare	2	1.0	74.5
summary-summaryRecommend-summaryCompare	2	1.0	75.5
summaryCompare-summaryCompare	2	1.0	76.4
summaryRecommend-summary-summaryRecommend	2	1.0	77.4
summaryRecommend-summaryRecommend	2	1.0	78.4
compare-compare-Recommend	1	.5	78.8
compare-Recommend-summary	1	.5	79.3
compare-summaryRecommend-Recommend	1	.5	79.8
Recommend-compare	1	.5	80.3
Recommend-compare-Recommend	1	.5	80.8
Recommend-Recommend-Recommend-summaryCompare	1	.5	81.2
Recommend-summary	1	.5	81.7
Recommend-summaryCompare	1	.5	82.2
Recommend-summaryCompare-summaryRecommend-compare	1	.5	82.7
Recommend-summaryRecommend	1	.5	83.2
summary-compare-compare	1	.5	83.7

	summary-compare-compare-summary-compare	1	.5	84.1
	summary-compare-compare-summary-Recommend-Recommend	1	.5	84.6
	summary-compare-compareRecommend	1	.5	85.1
	summary-Recommend-compare-summary-Recommend	1	.5	85.6
	summary-Recommend-Recommend	1	.5	86.1
	summary-Recommend-Recommend-Recommend	1	.5	86.5
	summary-Recommend-summary-Recommend	1	.5	87.0
	summary-Recommend-summary-summary-compare	1	.5	87.5
	summary-summary-compare-summary-summary-Recommend-Recommend	1	.5	88.0
	summary-summary-compareRecommend	1	.5	88.5
	summary-summary-Recommend-Recommend-Recommend	1	.5	88.9
	summary-summary-summary	1	.5	89.4
	summary-summary-summary-compare	1	.5	89.9
	summary-summary-summary-compare-Recommend	1	.5	90.4
	summary-summary-summary-summaryCompare	1	.5	90.9
	summary-summary-summary-summaryRecommend	1	.5	91.3
	summary-summary-summary-summaryRecommend-summaryCompare	1	.5	91.8
	summary-summary-summaryCompare-Recommend	1	.5	92.3
	summary-summary-summaryRecommend-summaryRecommend	1	.5	92.8
	summary-summaryRecommend-Recommend	1	.5	93.3
	summaryCompare-Recommend-Recommend	1	.5	93.8
	summaryCompare-summary-summary-summary-compare	1	.5	94.2
	summaryCompareRecommend	1	.5	94.7
	summaryRecommend-compare	1	.5	95.2
	summaryRecommend-compare-Recommend	1	.5	95.7
	summaryRecommend-Recommend-compare-summaryCompare	1	.5	96.2
	summaryRecommend-Recommend-summaryRecommend	1	.5	96.6
	summaryRecommend-summary-Recommend	1	.5	97.1
	summaryRecommend-summaryCompare	1	.5	97.6
	summaryRecommend-summaryCompare-compare-compare	1	.5	98.1
	summaryRecommend-summaryCompare-Recommend	1	.5	98.6
	summaryRecommend-summaryCompare-summary-summaryRecommend	1	.5	99.0
	summaryRecommend-summaryRecommend-Recommend-summary-summary-summaryRecommend-summaryCompare-summaryRecommend	1	.5	99.5
	summaryRecommend-summaryRecommend-summaryRecommend	1	.5	100.0
	Total	208	100.0	

D Highly rated NLG sequences

Sequence	Frequency	Percent	Cumulative Percent
summary-compare	10	10.8	10.8
summary-Recommend	9	9.7	20.4
summaryCompare	6	6.5	26.9
summaryRecommend	6	6.5	33.3
Recommend	5	5.4	38.7
compare	4	4.3	43.0
summary-summaryRecommend	4	4.3	47.3
summaryCompare-Recommend	4	4.3	51.6
summary-summary-summaryRecommend	3	3.2	54.8
summaryRecommend-Recommend	3	3.2	58.1
compareRecommend	2	2.2	60.2
summary-compare-compare-compare	2	2.2	62.4
summary-summary-compare	2	2.2	64.5
summary-summary-summary-Recommend	2	2.2	66.7
summary-summaryCompare	2	2.2	68.8
summary-summaryRecommend-summaryCompare	2	2.2	71.0
summaryCompare-compare	2	2.2	73.1
compare-compare	1	1.1	74.2
compare-Recommend	1	1.1	75.3
compare-summaryRecommend-Recommend	1	1.1	76.3
Recommend-compare-Recommend	1	1.1	77.4
Recommend-summaryCompare	1	1.1	78.5
Recommend-summaryRecommend	1	1.1	79.6
summary-compare-Recommend	1	1.1	80.6
summary-Recommend-Recommend	1	1.1	81.7
summary-Recommend-summary-Recommend	1	1.1	82.8
summary-Recommend-summary-summary-compare	1	1.1	83.9
summary-summary-compareRecommend	1	1.1	84.9
summary-summary-Recommend	1	1.1	86.0
summary-summary-summary	1	1.1	87.1
summary-summary-summary-compare-Recommend	1	1.1	88.2
summary-summary-summary-summaryRecommend	1	1.1	89.2
summary-summary-summaryCompare-Recommend	1	1.1	90.3
summaryCompare-Recommend-Recommend	1	1.1	91.4
summaryCompare-summary-summary-summary-compare	1	1.1	92.5
summaryCompareRecommend	1	1.1	93.5
summaryRecommend-compare-Recommend	1	1.1	94.6
summaryRecommend-Recommend-summaryRecommend	1	1.1	95.7
summaryRecommend-summaryCompare	1	1.1	96.8
summaryRecommend-summaryCompare-Recommend	1	1.1	97.8
summaryRecommend-summaryCompare-summary-summaryRecommend	1	1.1	98.9

```
|     |summaryRecommend-summaryRecommend                       |1        |1.1          |100.0            |
|     |--------------------------------------------------------|---------|-------------|-----------------|
|     |Total                                                   |93       |100.0        |                 |
|-----|--------------------------------------------------------|---------|-------------|-----------------|
```