

Where Data Meets Dinner: Building a restaurant recommendation dialog system

Group B1

Bas de Blok

b.deblok@students.uu.nl,

Mykola Chuprynskyy

m.chuprynskyy@students.uu.nl,

Melissa Rueca

m.rueca@students.uu.nl, 8463603

Luka van Hoeven

l.k.vanhoeven@students.uu.nl, 8888655

Dialog systems are machine-based systems designed to help users to solve specific tasks. The goal of this project was to design, develop and implement a dialog system capable of classifying dialog acts, interpreting user preferences and providing restaurant recommendations. We implemented two baseline classification systems: a majority-class baseline and a rule-based baseline. We also trained two machine learning classifiers: logistic regression and feed forward neural network. We evaluated the models using accuracy and they showed good results on both the original and the deduplicated data, although the latter performed slightly worse. We also conducted an error analysis to identify challenging dialog act and difficult utterances. The system uses a dialog manager to control how the system interacts with the user. It asks the user for missing information, understand the user's needs, provides suggestions based on the information collected and can offer additional recommendations if the user is not satisfied with the initial suggestion. Some possible improvements include using bigrams instead of unigrams to recognize pairs of words, applying a wider semantic analysis to help the model with negation sentences and adding a spell-checking during preprocessing to fix misspelling. In addition to this the system does not confirm user utterances, in combination with not handling negations consistently it could lead to situations where the user does not want a certain type of food, but the system thinks it does.

1 INTRODUCTION

Dialog systems are machine-based systems designed to help users solve specific tasks. They are commonly used in application such as controlling appliances, finding restaurants or making travel reservations [1].

The main goal of the project was to develop a dialog system able to provide restaurant recommendations based on user preferences. This required a system which could identify what the user is requesting from the system and provide appropriate answers for the user based on their utterance. To determine what the user is requesting we used a rule-based method and machine learning classifiers techniques, explained in **section 3**, trained with data explained in **section 2**. In addition to this the system needed to have access to restaurants it can recommend to the user and pick the appropriate one. It should be able to pick the appropriate one by extracting the preferences of the user from their utterance, the implementation of this part of the system is explained in **section 4**. The main interaction takes place in the terminal. Users interact with the system by asking questions, providing information or responding to the system's questions.

The system includes a dialog manager, two baseline models for comparison (rule-based and majority-class) and two machine learning classifiers (logistic regression model and feed forward neural network) to assign the correct dialog act to each user utterance. We evaluated them using accuracy and we conducted an error analysis to identify challenging dialogue acts and difficult utterances. The dialog manager guides the conversation according to the current state and user input. To achieve the result a state transition diagram was built (**section 2.2**). Anything possible conversation that could happen outside this diagram will not be covered by this system and is out of the scope of this project. This state transition diagram was a reference to how the interaction with the system was supposed to go. The implementation of the code was done by splitting the parts that needed to be coded amongst 4 group members. The code made by one group member was checked by the other group members.

This paper describes the dataset and preprocessing steps (**Section 2**), the implementation and evaluation of baseline and machine learning system (**Section 3**), the dialog manager (**Section 4**), the reasoning behind recommendations of the system (**Section 5**) and the configurability (**Section 6**) of the system.

2 DATA

The dataset will be used to identify the user's utterance. The dataset used in this project is derived from the second Dialog State Tracking Challenge [2], which focuses on restaurant recommendations dialogs.

2.1 Dataset

The dataset contains 3235 dialogs, each representing an interaction between a user and the system. The user expresses preferences such as type of cuisine, location and price range and the system responds either by giving recommendations, requesting additional preference or giving general communicative statements.

Analyzing the dataset, it can be observed that there are some utterances repeated across dialogs, accounting for approximately 79% of the total.

The dialogs were collected using automatic speech recognition, which can introduce transcription errors or linguistic inconsistencies. Specifically, we may encounter misspelled words, onomatopoeic expression representing sounds or noises, missing punctuation or colloquial expressions typical of spoken language. These issues can affect system performance, for example a misspelled word might not be classified correctly. Therefore, it was necessary to handle these cases to ensure proper recognition, for example the Levenshtein edit distance was used to handle misspellings.

The dataset includes 15 dialog acts to classify user and system the utterances they are explained in the following table:

Classification	Explanation
ack	The user acknowledges the system utterance
affirm	The user gives a positive confirmation
bye	The user says goodbye
confirm	The user asks for confirmation about the attribute of a restaurant
deny	The user rejects the system utterance
hello	The user says hello
inform	The user states a preference
negate	The user negates a statement that the system has made
repeat	The user asks for repeating the last utterance
reqalts	The user asks for an alternative suggestion

reqmore	The user asks for more suggestions
request	The user requests information about a restaurant
restart	The user requests to restart the dialogue
thankyou	The user says thank you/goodbye
null	The user utterance cannot be classified

Table 1: Classification and explanation of the data labels

2.2 State Transition Diagram

The dialog system is modelled as a state transition diagram, in which each state is a stage of the interaction. The state transition diagram contains 10 states:

1. Welcome
2. Ask area
3. Ask type of food
4. Ask for price range
5. Ask for additional requirements
6. Give suggestion
7. Give choice to pick the suggestion or restart conversation
8. Ask for restaurant details
9. Give restaurant details
10. End conversation

The system transitions between the states depends on user inputs and slot fulfilment.

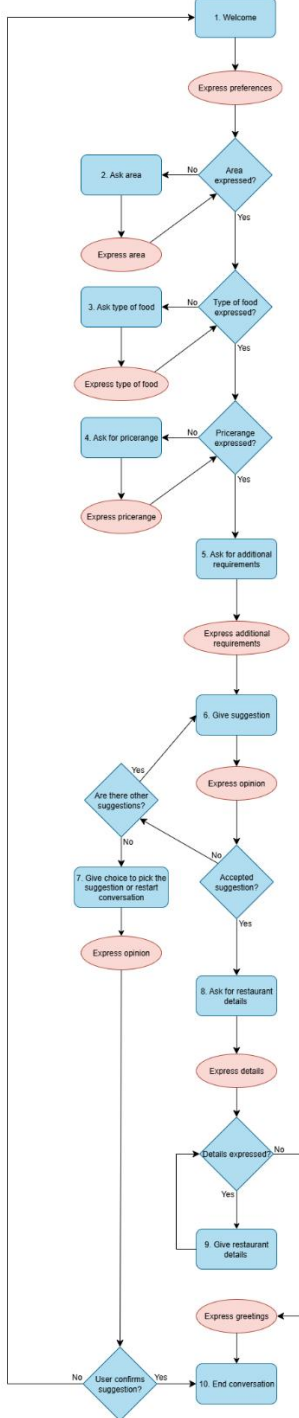
Here an example dialog snippet:

```

Task 01527: You want to find a moderately priced restaurant and it should serve bistro food.
If there is no such venue how about british type of food. Make sure you get the phone number.
system: Hello, welcome to the Cambridge restaurant system! You can ask for restaurants by
area, price range or food type. How may I help you?
user: im looking for a moderately priced restaurant
system: In which area would you like to eat?
user: south
system: What kind of food would you like?
user: british food
system: Do you have additional requirements?
user: no
system: I have found a restaurant that matches your preferences. It is saint johns chop
house food in the south area with a moderate price range. Do you like it?
system: Would you like more information about the suggested restaurant (phone, address or
postcode)?"
user: no

```

Figure 1: State transition diagram



system: Thank you for using the Cambridge restaurant system. Goodbye!

In this example the first system utterance corresponds to the “Welcome” state 1 because he’s greeting the user and asking him if it needs help. The system enters the “Ask area” state 2 when it asks “In which area would you like to eat?” and the “Ask type of food” state 3 when the system asks “What kind of food would you like?” because the system is trying to fill all the user preferences. The system “Do you have additional requirements?” corresponds to the “Ask for additional requirement” state 5. When the system provides the suggestion, we are in the “Give suggestion” state 6. After that, the system asks for restaurant details and we are in the “Ask for restaurant details” state 8”. The final “thank you good bye” corresponds to the “End conversation” state 10.

3 MACHINE LEARNING

In this section we explain the preprocessing of data, the implementation of the two baseline-systems, the machine learning classifiers used and the evaluation of the models.

3.1 Data preprocessing

Data preprocessing was a necessary step to prepare the dataset for all subsequent phases of the project.

The dataset was first subjected to a text cleaning, where all the utterances were normalized by converting them to lowercase and removing punctuation. This made the text more consistent and easier to analyse during classification.

Furthermore, two distinct versions of the dataset were constructed: in addition to the original dataset, which includes duplicate utterances, a deduplicated version was created, keeping only one instance of each utterance. Both the original and deduplicated datasets were then split into a training set (85% of the data) and a test set (15% of the data).

The original dataset contains a total of 25501 utterances, of which 20142 are duplicated. The average length of an utterance is 3.73 words. The label distribution is shown in table 2. There was a total of 59 out of vocabulary words in the test set for the original data. Some examples are ‘bristish, before, feed, venison’. The deduplicated test set contains 65 out of vocabulary words with things like ‘look, bat, po, int, bristish’. We can observe that some of these words are correct, but were not included in the training set, but some words were misspelled like ‘bristish’ meaning ‘british’, which is part of the training set.

Label	Train data original	Test data original	Train data deduplicated	Test data deduplicated
Inform	8638	1524	2603	459
Request	5520	974	817	144
Thankyou	2770	489	371	66
Reqalts	1485	262	141	25
Affirm	983	173	113	20
Negate	370	65	98	17
Bye	226	40	69	12
Confirm	146	26	51	9
Hello	79	14	35	6
Repeat	28	5	17	3
Ack	24	4	11	2
Deny	23	4	11	2
Restart	12	2	6	1
Reqmore	4	1	1	0

Table 2: Label distribution

3.2 Baseline systems

We implemented two baseline systems.

The first one was a majority-class baseline, which assigns the most frequent dialog act to every utterance, regardless of its content. In our dataset, the most frequent dialog act is ‘inform’, as such this baseline will predict this label for every utterance.

The second baseline was a rule-based system, which uses keyword matching to assign a dialog act to each utterance. For each dialog act we defined a set of keywords (single words or multiple words). The system searches for their presence in the user’s input: the multiple words must appear together to count as a match, and the single words can appear anywhere in the utterance. Once a match is detected, the utterance is immediately assigned to the corresponding dialog act.

An order of priority was defined among the dialog acts to prevent ambiguity and ensure that more distinctive acts prevail over more common ones. Thanking and goodbye acts are given the highest priority, as they typically dominate the communicative intent of an utterance even when additional information is present. These are followed by the dialog acts which request for repetition, restarting the conversation or asking for alternatives. Confirmation and denial acts come next because they allow the system to recognize then the user agrees or disagrees with the given information. The “null” dialog act is at the bottom of the hierarchy because it represents noise. If no keywords are matched, the utterance is assigned the “inform” dialog act. This mechanism ensures that each utterance is classified according to the most relevant dialog act for the interaction.

3.3 Machine learning classifiers

For the machine learning part, we decided to train two different machine learning classifiers on the dialog act data: Logistic regression and Feed Forward neural network.

3.3.1 Sequential model: Feed forward neural network

The feed forward neural network makes a vocabulary based on all the unique words in the training set. The system training data is then converted into a bag of words representation with the counts of each word in the sentence. The model is trained on the bag of words representation and the label. A validation set was made with 20% of the total training set, leaving 80% of the training set for actual training. The model is a sequential neural network with an input layer which has the number of words in the vocabulary + 1 (for any out of vocabulary words). It has 3 hidden layers with 128, 64 and 32 neurons respectively. In the output layer it uses SoftMax to assign probabilities to each of the 15 dialog acts, the highest probability is considered the label.

3.3.2 Logistic regression model

The Logistic Regression model uses TF-IDF features extracted from the utterances and it can also include character n-grams to capture patterns inside words. The model learns to predict the dialog act of each utterance from the training data and is evaluated on the test set using accuracy, macro F1-score and a confusion matrix.

3.4 Evaluation and error analysis

We evaluated the models' performance using accuracy on the test sets. We used accuracy to provide a global measure of performance on the test sets. As seen in table 2, the dataset is very imbalanced when it comes to label distribution. To compensate for this, we also evaluate on F1-score to show the balance between recall and precision. In this way we provided a global measure of performance. The results of this evaluation can be found in table 3.

We also analyzed difficult instances by writing our test cases. There were of two types: utterances affected by speech recognition error or misspellings (for example: "can you find an Italian restaurant") and utterances containing negation (for example: "i don't want Italian restaurant"). The system often failed to predict the correct dialog act for these utterances, and this highlights the areas where robustness could be improved. This could potentially be improved with using bigrams instead of unigrams, which may teach the model that 'didn't want' is not the same as 'want' [3].

There were some utterances which were mislabelled with interesting results. The utterance "thank you good bye" was mislabelled in the sequential models as bye, whilst it should be thankyou. However, this case is quite unique, since it could suffice for both the thankyou or the bye label. Moreover, the sequential model predicted "okay let me try this again" to be null and ack for the original and deduplicated model respectively. The correct label is repeat. From table 2 we observe that there are a limited number of examples for repeat in comparison to inform. The original data logistic regression model also had difficulty with this utterance, labelling it null. So, the conclusion is that misspellings cause issues as well as denying in sentences. A wider semantic analysis on the sentences might help, especially for negation sentences. For the errors with the misspelled words, we could involve spell checking in the preprocessing, this could lead to a more robust dataset and thus more robust output.

When looking at the F1-scores of individual class-labels, the sequential model with the original data scores a 0 on label 'ack' and after that a 0.8 on label 'repeat', the 0 in ack is caused by no examples being in the training set for ack, the examples are in the validation or test set. This could be improved in future models by gathering more data and a more distributed dataset. Or use data augmentation to generate data for the classes which do not have a lot of data.

Model	Accuracy	F1-Score
Sequential, original	98.07%	98.20
Sequential, deduplicated data	89.41%	89.91
Logistic regression, original	97.12%	97.20

Logistic regression, deduplicated data	87.55%	88.16
Rule based model, original	85.22%	84.38
Rule based model, deduplicated data	82.44%	81.38
Majority baseline, original	39.84%	22.70
Majority baseline, deduplicated data	57.16%	41.58

Table 3: Accuracies and f1-scores for the models

An interesting observation we can make from the removal of duplicate data points is a decrease in accuracy over the test set for all different models.

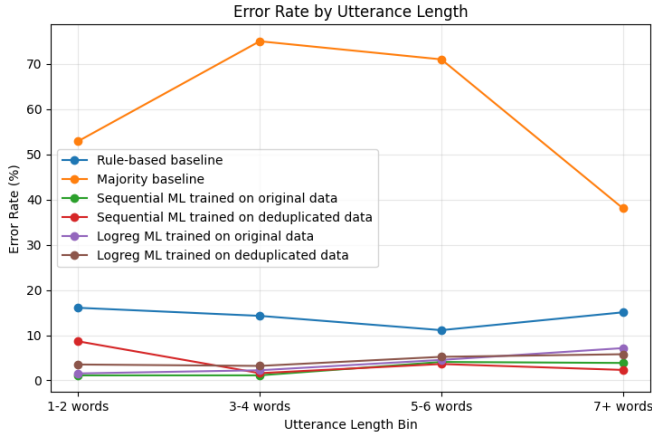


Figure 2: Error rate by utterance length on the test set

The above figure shows the error rate by utterance length for the different models. The graph shows that the sequential and logistic regression model perform about the same, the rule-based model performs slightly worse and the majority baseline performs the worst by far.

An interesting observation is the fact that the sequential model with original data had 70 incorrect predictions with an average length of 4.9 words, whilst the model with deduplicated data had 188 incorrect predictions with an average length of 2.27 words. It seems that the model with original data struggles to identify longer sentences, whilst the deduplicated model struggles to identify shorter sentences. This is also reflected in the logistic regression model, where the model with original data had 110 incorrect predictions with an average length of 5.21, whilst the deduplicated data model had 150 incorrect predictions with an average length of 3.93. The effect on the logistic regression model is less than on the sequential model.

4 DIALOG MANAGER

The dialog manager implements the state transition diagram. The system starts in the "Welcome" state (state 1). The system can transition from state to state every time it receives a user utterance. The state transition function of the dialog manager uses this utterance first by finding its corresponding dialog using one of the classification models. It uses this dialog acts to determine how it should move through the state transition diagram. Additionally, whenever the predicted dialog act is "inform" the system attempts to extract preferences for food, area and price range from the user utterance.

A short example dialog illustrating state transitions:

system: Hello, welcome to the Cambridge restaurant system? You can ask for restaurants by area, price range or food type. How may I help you? STATE 1

user: im looking for a cheap restaurant that serves seafood

system: In which area would you like to eat? STATE 2

user: north

system: What kind of food would you like? STATE 3

user: seafood

system: Do you have additional requirements? STATE 5

user: no

system: I have found a restaurant that matches your preferences. It is loch fyne food in the north area and it serves seafood food. Do you like it? STATE 6

user: yes!

system: Would you like more information about the suggested restaurant (phone, address or postcode)?" STATE 8

user: address

system: The address of loch fyne is the little rose 37 trumpington street. Do you need other information or exit? STATE 9

user: exit

system: Thank you for using the Cambridge restaurant system. Goodbye! STATE 10

Extraction of the user preferences is done by analyzing all the words in the user utterance. There is a set of valid words, words that the system understands as preference. For instance, cheap, expensive, Asian, Spanish etc. The system matches every word of the utterance against every valid word using Levenshtein distance. When the distance is smaller than a threshold, the system assumes that the word is a misspelling of a valid word. If a valid word is mentioned, the preference is kept in the system. So, if the user mentions ‘Asian’, it is registered as a preference for Asian food in the system. The system also is able to handle if the user does not mind. In the step of finding a matching restaurant, it will not filter for user input on the attribute where the user does not mind. The system will not be able to handle negations very well in this regard, which could be improved in the future.

4.1 System utterance templates

In this table, we present all system utterance templates with examples of sentences.

A full example dialog that traverses the diagram from the start state to the end state is provided in appendix A.1.

<i>State</i>	<i>Utterance template</i>	<i>Example</i>
Welcome (state 1)	"Hello, welcome to the Cambridge restaurant system! You can ask for restaurants by area, price range or food type. How may I help you?"	"Hello, welcome to the Cambridge restaurant system! You can ask for restaurants by area, price range or food type. How may I help you?"
	"I am sorry, I did not understand that. How may I help you?"	"I am sorry, I did not understand that. How may I help you?"
Ask area (state 2)	"In which area would you like to eat?"	"In which area would you like to eat?"
	"I am sorry, I did not understand that. In which area would you like to eat?"	"I am sorry, I did not understand that. In which area would you like to eat?"
	"What kind of food would you like?"	"What kind of food would you like?"

Ask type of food (state 3)	"I am sorry, we do not have that kind of food. What kind of food would you like?"	"I am sorry, we do not have that kind of food. What kind of food would you like?"
Ask for price range (state 4)	"What price range are you looking for?"	"What price range are you looking for?"
	"I am sorry, I did not understand that. What price range are you looking for (cheap, moderate or expensive)?"	"I am sorry, I did not understand that. What price range are you looking for (cheap, moderate or expensive)?"
Ask for additional requirements (state 5)	"Do you have any additional requirements? For example do you want the restaurant to be romantic, family-friendly, or quick?"	"Do you have any additional requirements? For example do you want the restaurant to be romantic, family-friendly, or quick?"
	"I am sorry, I did not understand that. Do you have any additional requirements? For example do you want the restaurant to be romantic, family-friendly, or quick?"	"I am sorry, I did not understand that. Do you have any additional requirements? For example do you want the restaurant to be romantic, family-friendly, or quick?"
Give suggestion (state 6)	"I have found a restaurant that matches your preferences. It is {suggestion['restaurantname']} food in the {self.area} area with a {self.pricerange} price range. Do you like it?"	"I have found a restaurant that matches your preferences. It is saint johns chop house food in the west area with a moderate price range. Do you like it?"
	"I am sorry, but I could not find any restaurants that match your requirement. Do you want to start over or exit?"	"I am sorry, but I could not find any restaurants that match your requirement. Do you want to start over or exit?"
	"I am sorry, I do not have any more suggestions that match your criteria. Do you want to start over or exit?"	"I am sorry, I do not have any more suggestions that match your criteria. Do you want to start over or exit?"
	"I have found another restaurant that matches your preferences called {suggestion['restaurantname']}. It is {self.food} food in the {self.area} area with a {self.pricerange} price range. Do you like it?"	"I have found another restaurant that matches your preferences called the oak bistro. It is british food in the centre area with a moderate price range. Do you like it?"
	"I am sorry, I did not understand that. Unfortunately there are no other restaurants matching your criteria. Do you want to start over or exit?"	"I am sorry, I did not understand that. Unfortunately there are no other restaurants matching your criteria. Do you want to start over or exit?"
	"I found a restaurant which is very {user_requirement}, it is called {restaurant_name}. {Optional: The food is {good/average} and} {Optional: it is {not crowded/somewhat crowded} and} {Optional: the length of stay is {appropriate/long}.} Overall, it is a great choice for a {user_requirement} restaurant. Do you like it?"	"I found a restaurant which is very romantic, it is called pizza hut fen ditton. the food is good and it is not crowded and the length of stay is appropriate. Overall, it is a great choice for a romantic restaurant."
	"I found a restaurant which is somewhat {user_requirement}, it is called {restaurant_name}. {Optional: The food is {good/average} and} {Optional: it is {not crowded/somewhat crowded} and} {Optional: the length of stay is {appropriate/long}.} It is {user_requirement} because of the {positive_aspects}, but it is not because of the {negative_aspects}. Do you like it?"	"I found a restaurant which is somewhat romantic, it is called cotto. the food is good and it is not crowded and the length of stay is appropriate. It is romantic because of the crowdedness, but it is not because of the food quality, length of stay."
	"Unfortunately there are no other restaurants matching your criteria. You can either pick the suggested restaurant or start over. Would you like to pick the suggested restaurant or start over?"	"Unfortunately there are no other restaurants matching your criteria. You can either pick the suggested restaurant or start over. Would you like to pick the suggested restaurant or start over?"
Give choice to pick the suggestion or restart conversation (state 7)		

	"I am sorry, I did not understand that. Unfortunately there are no other restaurants matching your criteria. Would you like to pick the suggested restaurant?"	"I am sorry, I did not understand that. Unfortunately there are no other restaurants matching your criteria. Would you like to pick the suggested restaurant?"
Ask for restaurant details (state 8)	"Would you like more information about the suggested restaurant (phone, address or postcode)?"	"Would you like more information about the suggested restaurant (phone, address or postcode)?"
Give restaurant details (state 9)	"The phone number of {self.picked_suggestion['restaurantname']} is {self.picked_suggestion['phone']}. Do you need other information or exit?"	"The phone number of chiquito restaurant bar is 01223 400170. Do you need other information or exit?"
	"The address of {self.picked_suggestion['restaurantname']} is {self.picked_suggestion['addr']}. Do you need other information or exit?"	"The address of chiquito restaurant bar is 2g cambridge leisure park cherry hinton road cherry hinton. Do you need other information or exit?"
	"The postcode of {self.picked_suggestion['restaurantname']} is {self.picked_suggestion['postcode']}. Do you need other information or exit?"	"The postcode of chiquito restaurant bar is c.b 1. Do you need other information or exit?"
End conversation (state 10)	"Thank you for using the Cambridge restaurant system. Goodbye!"	"Thank you for using the Cambridge restaurant system. Goodbye!"
	"I am sorry, I did not understand that. Thank you for using the Cambridge restaurant system. Goodbye!"	"I am sorry, I did not understand that. Thank you for using the Cambridge restaurant system. Goodbye!"

Table 4: System utterance template

5 REASONING

After collecting the user's preferences and any additional requirements, the system applies reasoning rules to analyse the possible restaurants and decide which one is the best option.

Each restaurant has properties such as crowdedness, length of stay and food quality and the system uses specific rules to infer additional characteristics from these properties. For example, a busy restaurant is considered not romantic, while a restaurant that allows for a long stay is considered romantic.

During the recommendation process, the system applies these rules and assigns the inferred properties to each restaurant when applicable. It also handles possible contradictions: if one rule indicates that a restaurant is romantic while another indicates it is not, the system chooses the restaurant which has the properties that most align with the rule by counting. It then reasons why it picked what it picked and explains which part of, for instance, 'romantic' suffices and which part does not. It is quite hard to meet all requirements for the given user input, thus many times it will be the case that sacrifices need to be made.

###Before the user has managed to give its preferences (British, moderate, centre)

System: Do you have any additional requirements? For example do you want the restaurant to be romantic, family-friendly, or quick?

User: i would like the restaurant to be romantic

System: I found a restaurant which is somewhat romantic, it is called cotto. the food is good and it is not crowded and the length of stay is appropriate. It is romantic because of the crowdedness, but it is not because of the food quality, length of stay.

One can observe that the system tries to explain its reasoning for picking this restaurant for the user. The requirements and antecedents (romantic, touristy, family-friendly, trashy etc.) can be dynamically expanded in the system.

If a requirement is fully met the system lists the attributes that make up the requirement:

System: I found a restaurant which is very romantic, it is called pizza hut fen ditton. the food is good and it is not crowded and the length of stay is appropriate. Overall, it is a great choice for a romantic restaurant.

If none of the requirements are met the system apologizes and this results in there being no suggestion. The dynamic list can easily be expanded, so over time the system can easily handle more types of extra requirements.

6 CONFIGURABILITY

The dialog system includes different options that let the user adjust how it behaves. These options allow the system to be more flexible and adapt to different preferences.

We implemented four configuration options. First, the system can be configured to allow or disallow changing the preferences of the user after one has already been found in previous user utterances. For example, if the user initially specified a price range, the system may either accept a new incoming price range later in the conversation or ignore the new input. Second, the system can display its output in all capital letters or not. Third, it is possible to have a one-second delay before the system replies to the user. Finally, the user can decide to use a baseline model instead of a machine learning model for its classification of user utterances.

These options are set by the user through command line arguments when starting the system. In the code, they are implemented as boolean flags within the dialog management class, which are checked during execution to determine the system's behaviour.

7 CONCLUSION

This paper aimed at developing a dialogue system to provide recommendations for restaurants to the user. The dataset was carefully analyzed and parsed, used to train 2 machine learning models as well as creating 2 baseline models to evaluate the performance. The models were aimed at recognizing what type of utterance the user has given to the system. A system diagram was built, which guided the implementation in code. The diagram was updated constantly to align with the requirements and current state of the codebase. The system extracted preferences from the user, correcting errors using Levenshtein distance. It had a reasoning component which could also handle different extra requirements, for example a restaurant could be 'romantic', 'family-friendly' etc. Overall, the system is flexible, letting the user input multiple wishes at the same time. Currently the system cannot book a table or handle live availability of seats in restaurants. In addition to this the system does not confirm user utterances, in combination with not handling negations consistently it could lead to situations where the user does not want a certain type of food, but the system thinks it does. This could be improved in future iterations. The system is configurable to allow or disallow changing preferences, display output in capital letters and have a delay before uttering a response. These elements will be tested to answer questions about the user-friendliness of the system.

The systems classification had trouble classifying utterances with negations in them, as well as classifying labels which had a limited number of examples in the training set (ack, repeat). For both implementations of the machine learning model the accuracy of the model trained on deduplicated data performed worse than the model trained on the original data. For the model for logistic regression the performance went from 97.12% to 89.41%, for the sequential model it went from 98.07 to 89.41. From the data it seems that all models perform better on utterances with 3-4 words, the effect size could be investigated in future research. Data augmentation can be used to make a more even distribution of class examples, which

could improve the performance of the models. In addition to this training the models with bigrams or potentially add more context about the conversation may improve the model's performance. Moreover, future research can investigate models with more contextual understanding, for example LSTM or GRU, and investigate the effects of these types of models on performance.

8 TEAM MEMBER CONTRIBUTIONS

For this project the team members who participated carefully read through the assignment before the assigned work hours. These assigned work hours we mainly took to discuss the approach we would take for the assignment, after which we drafted initial documents and todo-lists. For example, we made the initial version of the state transition diagram together, after which we assigned each group member tasks and deadlines. We could complete these tasks asynchronously by using a git-repository and shared documents. When tasks were dependent on other tasks, we communicated and planned deadlines to help each other complete them. This resulted in participating group members putting the required hours and effort into the project.

<i>task</i>	<i>Bas</i>	<i>Mykola</i>	<i>Melissa</i>	<i>Luka</i>
Raw data cleaning		3		
Majority class baseline	1	0.5		
Keyword matching baseline			2.5	
Machine learning classifiers		3.5		4
Combining models into 1 cli	5			
Evaluation	4	2	3	2
State transition diagram	4	4	4	4
State transition function	6	3.5	3.5	2
Preference prediction			1	5
Impl. rules and contradictions		6		9
Configurability	4		1.5	
Report	12	11	17	12
Cleanup and doctstrings	6			1
Refactoring and testing	3	4	3	2
Reading, preparing assignments etc.	6	5.5	8	8
Total	51	43	43.5	49

REFERENCES

- [1] S. Arora, K. Batra, and S. Singh, 'Dialogue System: A Brief Review', Jun. 18, 2013, *arXiv*: arXiv:1306.4134. doi: 10.48550/arXiv.1306.4134.
- [2] Microsoft, '2011 Dialog System Technology Challenge (DSTC)'. [Online]. Available: <https://www.microsoft.com/en-us/research/event/dialog-state-tracking-challenge/>
- [3] C.-M. Tan, Y.-F. Wang, and C.-D. Lee, 'The use of bigrams to enhance text categorization', *Information Processing & Management*, vol. 38, no. 4, pp. 529–546, Jul. 2002, doi: 10.1016/S0306-4573(01)00045-0.
- [4] 'Methods in ai research'.

A APPENDICES

A.1 Full Example Dialog

system: Hello, welcome to the Cambridge restaurant system! You can ask for restaurants by area, price range or food type. How may I help you? STATE 1

user: i'm looking for an italian restaurant

system: In which area would you like to eat? STATE 2

user: in the south

system: What price range are you looking for? STATE 4

user: expensive

system: Do you have any additional requirements? For example do you want the restaurant to be romantic, family-friendly, or quick? STATE 5

user: romantic

system: I found a restaurant which is very romantic, it is called frankie and bennys. the food is good and it is not crowded and the length of stay is appropriate. Overall, it is a great choice for a romantic restaurant. Do you like it? STATE 6

user: no

system: Unfortunately there are no other restaurants matching your criteria. Would you like to pick the suggested restaurant or start over? STATE 7

user: start over

system: Hello, welcome to the Cambridge restaurant system! You can ask for restaurants by area, price range or food type. How may I help you? STATE 1

user: i'm looking for british food

system: In which area would you like to eat? STATE 2

user: centre

system: What price range are you looking for? STATE 4

user: expensive

system: Do you have any additional requirements? For example do you want the restaurant to be romantic, family-friendly, or quick? STATE 5

user: romantic

system: I found a restaurant which is very romantic, it is called midsummer house restaurant. the food is good and it is not crowded and the length of stay is appropriate. Overall, it is a great choice for a romantic restaurant. Do you like it? STATE 6

user: yes

system: Would you like more information about the suggested restaurant (phone, address or postcode)?
STATE 8

user: address

system: The address of midsummer house restaurant is midsummer common. Do you need other information or exit? **STATE 9**

user: postcode

system: The postcode of midsummer house restaurant is nan. Do you need other information or exit? **STATE 9**

user: no

system: Thank you for using the Cambridge restaurant system. Goodbye! **STATE 10**