Find your Dog – FiDo

a dialogue system to help its user to decide which dog to get

Project report, Dialogue systems 2, LT2319, Autumn 2022

Max Boholm

Introduction

- Which dog to get might be a complex matter due to the great variability of dog breeds, e.g., level of energy, capacity for being trained and protectiveness
- FiDo is a dialogue system designed to help its user with this problem
- *Incremental search*: asking the user a series of questions about preferences for certain traits (e.g., trainability and energy), thereby filtering out preferred dogs in a database
- Also, dogs can be *described* and *compared*

API-Ninjas Dog API

- Information about more than 200 dogs
- Parameters of requests include `shedding`, `barking`, `energy`, `protectiveness`, `trainability` (values `0` to `5`)
- FiDo also work with a local database (*n*=177)
- Both API and local mode have limitations


```
[
  {
    "image_link": "https://api-ninjas.com/images/dogs/golden_retriever.jpg",
    "good_with_children": 5,
    "good_with_other_dogs": 5,
    "shedding": 4,
    "grooming": 2,
    "drooling": 2,
    "coat_length": 1,
    "good_with_strangers": 5,
    "playfulness": 4,
    "protectiveness": 3,
    "trainability": 5,
    "energy": 3,
    "barking": 1,
    "min_life_expectancy": 10,
    "max_life_expectancy": 12,
    "max_height_male": 24,
    "max_height_female": 24,
    "max_weight_male": 75,
    "max_weight_female": 65,
    "min_height_male": 23,
    "min_height_female": 23,
    "min_weight_male": 65,
    "min_weight_female": 55,
    "name": "Golden Retriever"
  }
]
```


(example response)

Data

- 10 dialogues were recorded, transcribed and distilled
- 3 phases:
  - Phase 1: naive (Lab 1)
  - Phase 2 and 3: simulated incremental search
  - Different system questions in phase 2 and 3
  - Phase 3 simulated additional functions (comparisons and descriptions of dogs)

Fundamental issues

- Different outcomes of incremental search
- How the system should ask questions and how to interpret user answers?

Incremental search

- Process whereby the answer, ***A***, to a question, ***Q***, is searched by finding answers, $\boldsymbol{a_1, \ldots a_n}$, to a series of sub-questions, $\boldsymbol{q_1, \ldots, q_n}$, such that those answers restrict the set ***R*** of possible values of A, $\{e_1, \ldots, e_n\}$
- Three outcomes:
  - **Success: $|\boldsymbol{R}| = 1$**; i.e. R contains a single element e_i (then $A = e_i$)
  - **Over-populated (OP)**: after q_1 to q_n have been answered, $|\boldsymbol{R}| > 1$
  - **Under-populated (UP)**: an answer a_i (to q_i) result in $\boldsymbol{R} = \varnothing$

How to ask questions?

- A problem:

  `S> On a scale from zero to five, what value should your dog have on F?`

  `U> three`

  interpretation: {*dog*: *dog*'s value for *F* = 3}

- Assumes preferences as isolated from other values of the scale: "three, *not more not less*"
- But some preferences rather covers a range of values
  - E.g., one who prefers dogs where *barking* = 1, might as well accept dogs where *barking* = 0.
- … but how to implement that?

The (preliminary) solution in FiDo

1. Transform `[0, …, 5]` to [*-3, …, +3*]
2. Interpretation: the preference for *F* covers the range from the score given, *s*, to the ...
   - maximum of the scale, when *s* > 0 (positive)
   - minimum of the scale, when *s* < 0 (negative)

- Example:

```
     S> One a scale from minus three to plus three, to what degree should
your dog (be) F?

(i)  U> two → {dog: dog's value for F ≥ 4} (positive score)

(ii) U> minus two → {dog: dog's value for F ≤ 1} (negative score)
```

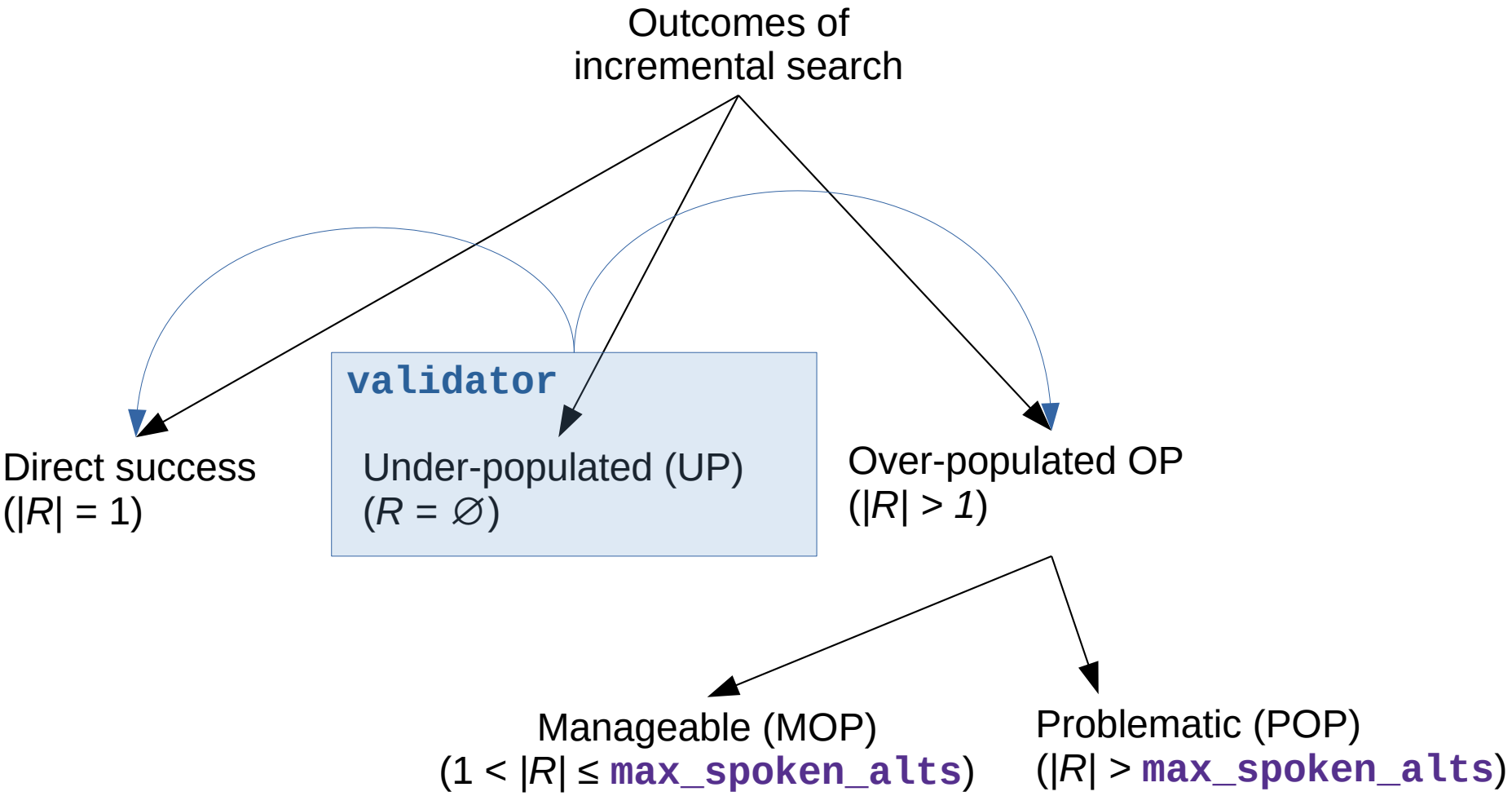
- Indeed, this “solution” has several problems – which we ignore for now!

Implementation

Built-in support for incremental search in TDM

```
<goal type="perform" action="suggest_dog">
   <plan>
      <inform>
        <proposition predicate="explain_procedure" value="procedure"/>
      </inform>
      <findout type="wh_question" predicate="what_dog_to_get"/>
      <invoke_service_action name="SuggestDog" postconfirm="true"/>
   </plan>
  </goal>

  <parameters question_type="wh_question" predicate="what_dog_to_get" incremental="true">
    <ask_feature predicate="pf_trainability"/>
    <ask_feature predicate="pf_shedding"/>
    <ask_feature predicate="pf_energy"/>
    <ask_feature predicate="pf_barking"/>
    <ask_feature predicate="pf_protectiveness"/>
  </parameters>
```


Outcomes of
incremental search
validator
Direct success
(|R| = 1)
Under-populated (UP)
(R = ∅)
Over-populated OP
(|R| > 1)
Manageable (MOP)
(1 < |R| ≤ max_spoken_alts)
Problematic (POP)
(|R| > max_spoken_alts)

MOP outcomes

In order to help the user with additional information in cases of MOP outcomes FiDo has for sub-goals for:

- *describe dog*: the system describes key characteristics of a dog
- *compare dogs*: the system provides a comparison of a target dog and another dog with regard to some trait
- *tell most dog*: the system tells which dog (or dogs) of the remaining ones in *R* that has (have) the highest value for some trait
- *tell least dog*: the system tells which dog (or dogs) of the remaining ones in *R* that has (have) the lowest value for some trait

Dialogues handled by FiDo

Dialogue type	`tdm`	`pipeline`
Success, along the way (asking q_1–q_j, where $j < 5$)	✓	✓
Success, all the way (asking q_1–q_5)	✓	✓
MOP Comparison	✓	✘
MOP Description	✓	✓
MOP Tell least	✓	✓
MOP Tell most	✓	✓
System induced answer-revision (anticipating and avoiding UP outcomes)	✓	✓
POP (minimal solution)	(✓)	(✘)

demo ...

Discussion

- A better understanding of “preference probing”
- Remembering *R*
- A better approach for POP outcomes
- Guidance to the user in case of MOP outcomes
- What predicates?
- Is incremental search really the best approach for this problem?

…. for listening!