Voice Recognition RPG

Baron Khan

Adding Voice Commands to a Game

- Commands to attack with a sword:
  - "attack with a sword"
  - "hit with something sharp"
  - "use a sword to fight"
  - "launch an assault with the sword"
  - "obliterate the enemy with a long weapon“
- Commands to heal the player:
  - "heal"
  - "recover"
  - "rest"
  - "heal with a potion"
  - "regenerate using an elixir"

3 years ago…

```
switch(input) {
  case "attack with a sword":
  case "hit with something sharp":
  //...
  case "obliterate the enemy with a pointy weapon":
     attackWithWeapon();
     break;
  case "heal":
  case "recover":
  //...
  case "rest":
     heal();
     break;
  case "heal with a potion":
  case "recover with a potion":
  //...
  case "regenerate using an elixir":
     healWithPotion();
     break;
  //...
```

Ad infinitum…

* Strings can be evaluated with a switch statement since Java 7.

1 year ago…

("attack" | "hit") . "with" . ["a"] . ("sword" | "blade")

1 year ago…

("attack" | "hit") . "with" . ["a"] . ("sword" | "blade")

Several minutes later…

("attack" | "hit" | "obliterate" | ("launch" . "an" . "assault")) . ("with" | "using") . ["a"] . ("sword" | "blade" | ("something" . ("pointy" | "sharp")))

An expression for each intent in the game

Now…

- CSV file:

	attack	**heal**
default	AttackDefault	HealDefault
weapon	AttackWeapon	
weapon-sharp	AtkWeaponSharp	
weapon-blunt	AtkWeaponBlunt	
healing-item		HealWithItem

Voice Recognition RPG project

- Create a text-based role-playing game controlled using voice commands
- Reduce developer workload as much as possible

- Three key areas for reducing workload:
  1. **Adding voice commands without hard-coding every acceptable phrase**
  2. Automatically assign physical properties to objects
  3. Generating new rooms in the game without manually placing objects

Motivation

- Online APIs such as Dialogflow and IBM’s Watson Conversation can be used to easily add commands

A Star Trek VR game using IBM’s Watson Conversation

Motivation

- Online APIs such as Dialogflow and IBM's Watson Conversation can be used to easily add commands

☹ 1 request = $$$$$

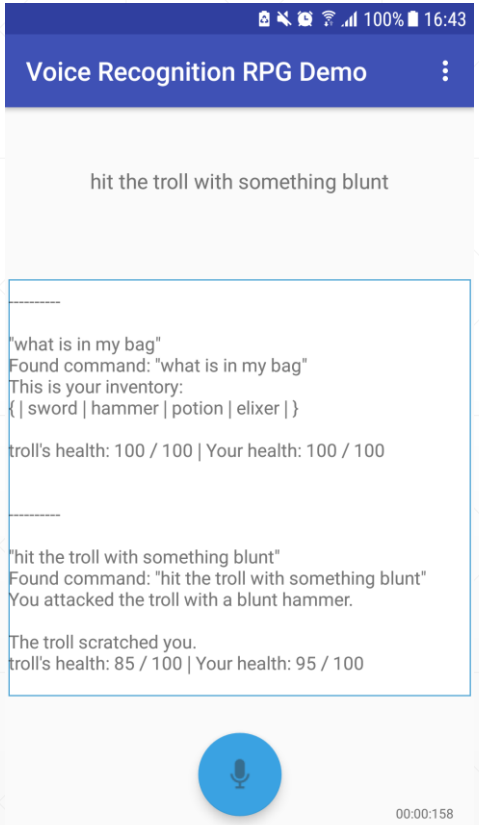
☹ Internet connection required

☹ Privacy?

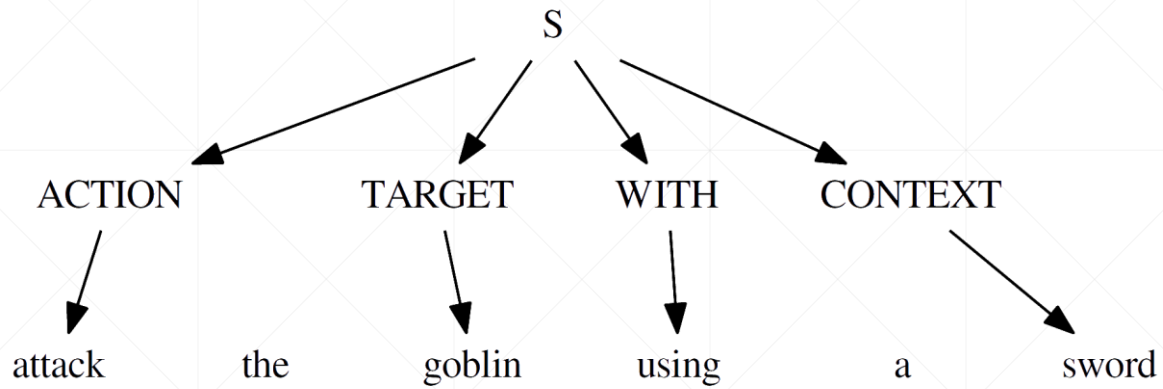
A Star Trek VR game using IBM's Watson Conversation

RPG Demo

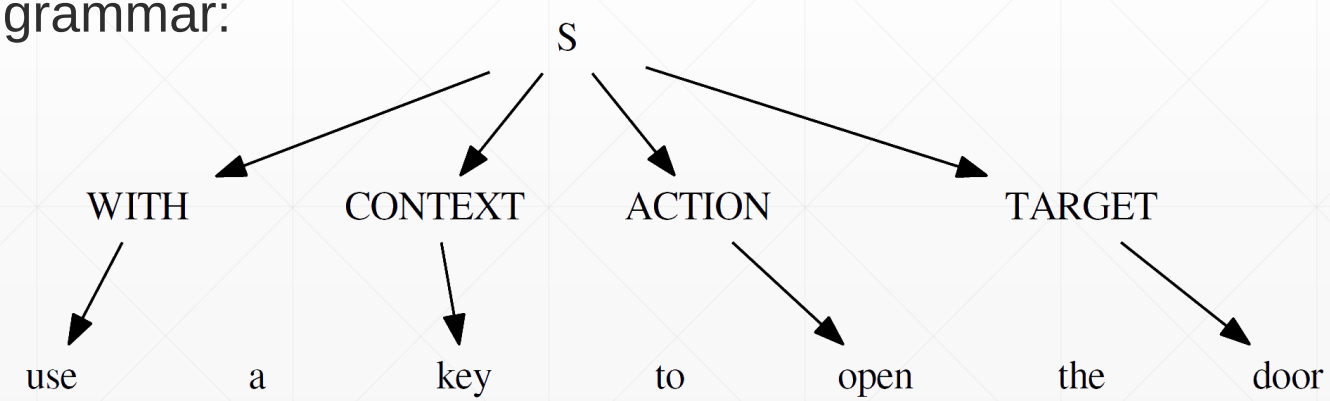
- Two different gameplay styles:
  - *Overworld* Mode
    - Exploration – interacting with objects
    - Examples: Zork, point-and-click adventure games
  - *Battle* Mode
    - Turn-based – fighting enemies
    - Examples: Pokémon, Final Fantasy



How it Works: Slot-Filling



- Alternative grammar:



How it Works: Context-Action Maps

- Overworld Mode:

action name

context type

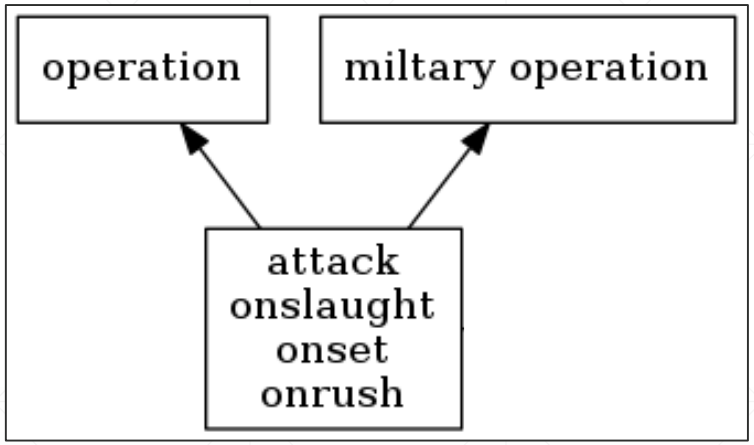
	look	**show**	**cut**	**break**	**grab**	**open**
default	LookDefault	ShowDefault	CutDefault	BreakDefault	GrabObject	OpenObject
weapon			CutWeaponNotSharp	BreakWeaponNotBlunt		
weapon-sharp			CutWeaponSharp	BreakWeaponNotBlunt		
weapon-blunt			CutWeaponNotSharp	BreakWeaponBlunt		
vision-item	LookDefault					

- Battle Mode:

	attack	**heal**	**show**	**look**
default	AttackDefault	HealDefault	ShowDefault	LookDefault
weapon	AttackWeapon			
weapon-sharp	AttackWeaponSharp			
weapon-blunt	AttackWeaponBlunt			
healing-item		HealItem		

Heart of the System: WordNet

- Created by Princeton University
- Large lexical database of English words
- Forms tree of words
  - Each node is a set of synonyms (*synset*)
  - Parent nodes: hypernyms



- Used to calculate semantic similarity between two words

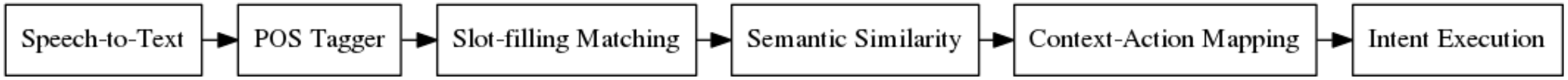
Semantic Similarity Methods

- Algorithms to calculate similarity of two words using WordNet
- Numerous algorithms
  - Path-based (Wu and Palmer, Leacock and Chodorow)
  - Information Content (Lin, Resnik)
  - Overlaps in Definitions (Lesk)
- Most implementations provided by WS4J library
  - Requires ILexicalDatabase interface to be implemented
- Some implemented manually (COS, FAST LESK)

$$sim_{wup} = \frac{2 * depth(LCS(w_1, w_2))}{depth(w_1) + depth(w_2)}$$

How it Works: Pipeline

- Input: user’s audio
- Output: text response (+ side effect in application)



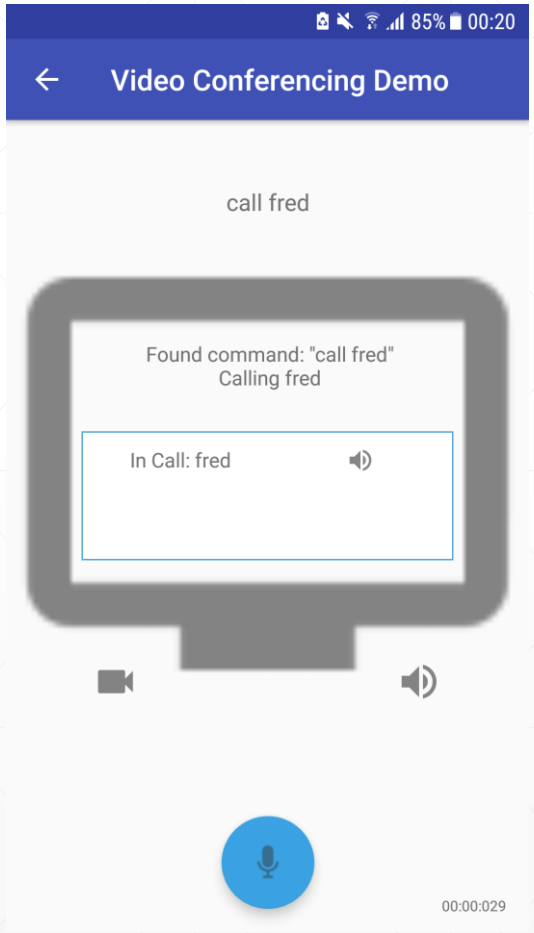
Applied to Other Domains

- Video Conferencing commands

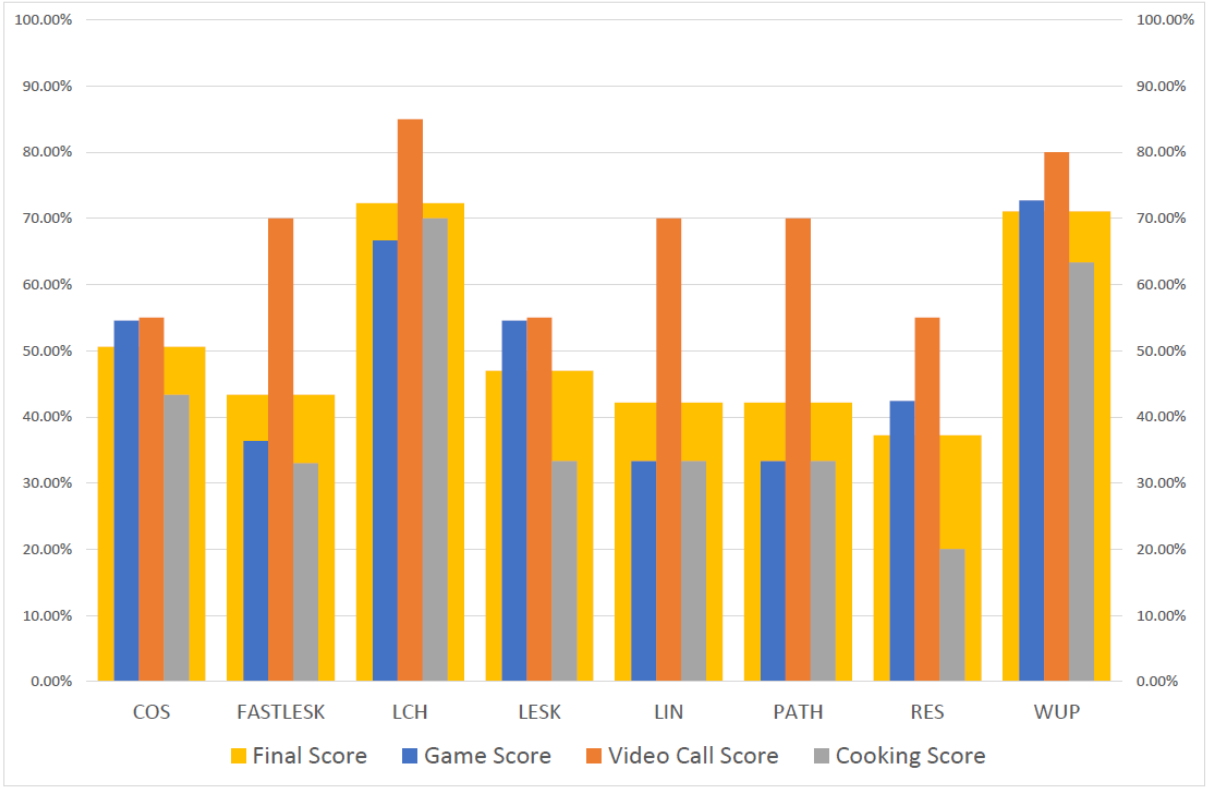
	phone	**stop**	**mute**
default	PhoneContact	StopCall	Mute
video	PhoneContact		
audio	PhoneContact		
contact	PhoneContactAudio	StopCallContact	

- Cooking commands
  - “boil the eggs”
  - “use a spoon to stir the soup”

	make	**stir**	**boil**	**pour**	**serve**
default	Make	Stir	Boil	Pour	Serve
spoon	Make	StirSpoon			
cooker	Make		BoilCooker		
food	MakeFood				



Correctness of Semantic Similarity Methods



Performance of Semantic Similarity Methods

- Near-instantaneous response (<= 0.1s) required

Method	**Average Time per Command / s**		
	PC	*Phone*	*Raspberry Pi*
COS	0.023	0.85	6.40
FASTLESK	0.012	1.5	5.00
LCH	0.004	0.099	3.29
LESK	1.35	280.02	2068.81
LIN	0.009	0.29	5.19
PATH	0.004	0.088	2.96
RES	0.007	0.12	3.29
WUP	0.015	0.16	5.15

Evaluation of Semantic Similarity Methods

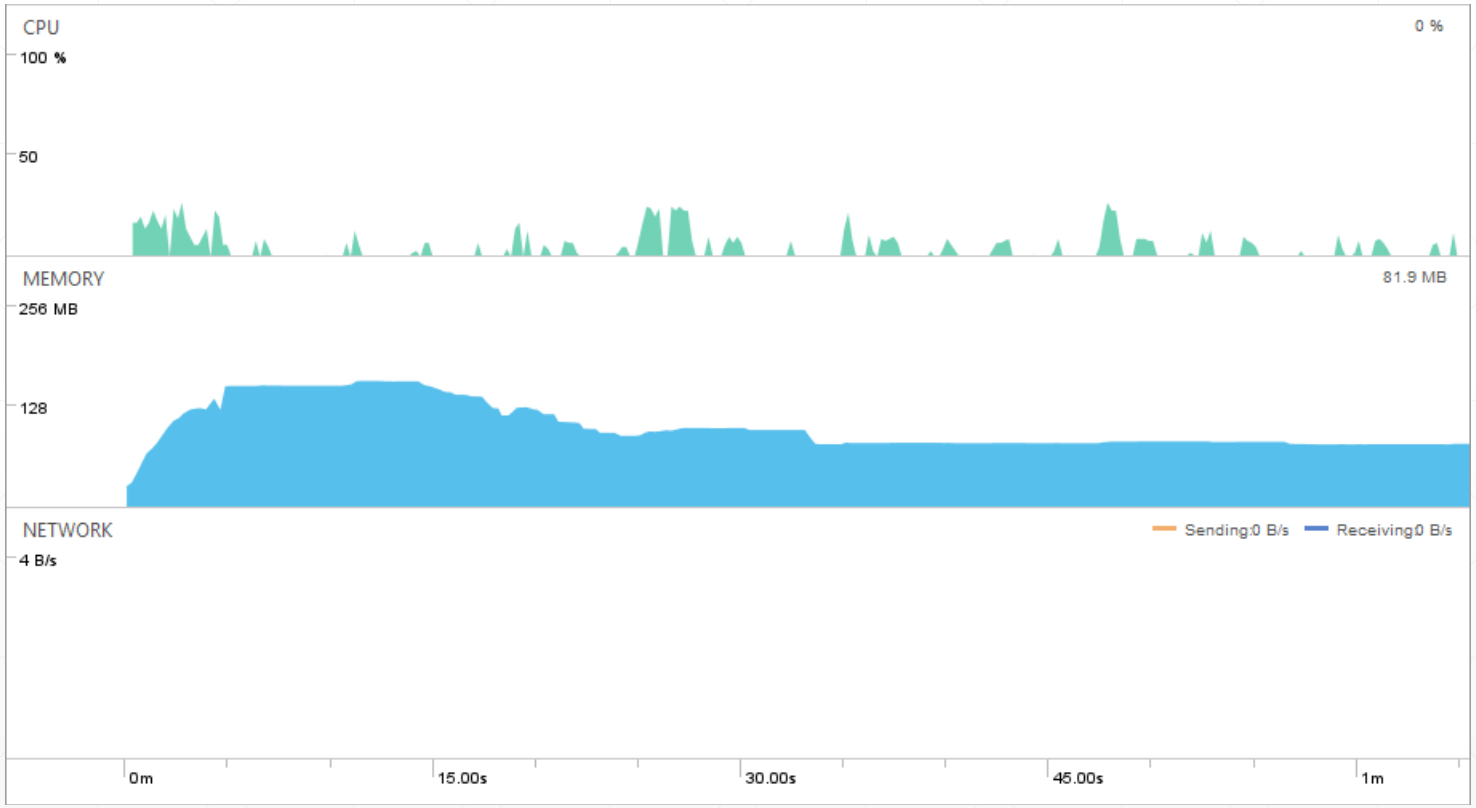
- Path-based methods have overall highest accuracy and speed
  - Wu and Palmer (WUP)
  - Leacock and Chodorow (LCH)
- Different methods perform better in different domains
- WUP method chosen for RPG demo

- ~70% correctness for the best methods in tests

System Features Improve Accuracy

- Confirmation and suggestions on ambiguous intents
- Chaining multiple commands in one utterance
- Detect multiple targets or contexts
- Synonym-mapping
- Ignoring incorrect matches
- Sentence-matching

Application Performance



- Three key areas for reducing workload:
  1. Adding voice commands without hard-coding every acceptable phrase
  2. **Automatically assign physical properties to objects**
  3. **Generating new rooms in the game without manually placing objects**

Room Generation from Text

- Text description of room → Java source file for room
- Use semantic similarity engine to find similar objects in text description
- Binary relationships between two objects as conditionals

There is a *table in the middle of the room.
An *armchair is underneath the table.
A *potion is on the table.
A *knife is with the potion.


```java
package com.khan.baron.voicerecrpg.game.rooms;
/* TODO: insert object imports */

public class RoomPuzzle extends Room {
    public RoomPuzzle() {
        super();
        addDescriptionWithObject(
            "There is a table in the middle of the room.",
            new GlassTable());
        addDescriptionWithObjectCond(
            "An armchair is underneath the table.",
            "An armchair is in the room.",
            new Chair(),
            () -> getRoomObjectCount("table") > 0);
        addDescriptionWithObjectCond(
            "A potion is on the table.",
            "A potion is now on the floor.",
            new Potion("potion"),
            () -> getRoomObjectCount("table") > 0);
        addDescriptionWithObjectCond(
            "A knife is with the potion.",
            "A knife is in the room.",
            new Weapon("knife"),
            () -> getRoomObjectCount("potion") > 0);
    }
}
```

Summary of Voice Recognition RPG Project

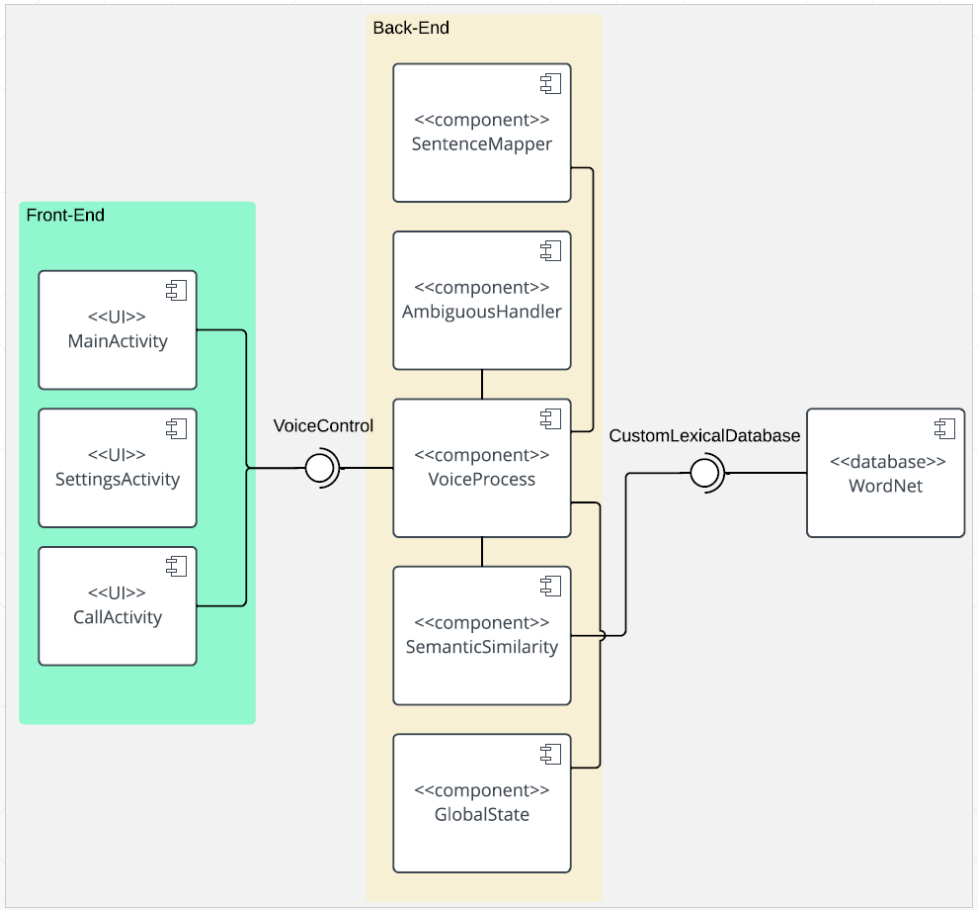
- Created prototype for a voice-controlled, text-based RPG on Android
- Created offline voice recognition system using WordNet
  - Good performance with near instantaneous processing on modern devices
  - Provides good foundation that can be improved upon
  - Applied to different domains (games, video conferencing, etc)
  - Created standalone Java library
- Evaluation of different semantic similarity methods
- Explored other areas for improving development of RPG

Voice Recognition RPG

Supplementary Slides

Baron Khan

System Architecture



Part-of-Speech (POS) Tagging

- Label each word in text with its part-of-speech
- Using Stanford POS tagger for Java

CC	Coordinating conjunction	TO	to
CD	Cardinal number	UH	Interjection
DT	Determiner	VB	Verb, base form
EX	Existential there	VBD	Verb, past tense
FW	Foreign word	VBG	Verb, gerund or present participle
IN	Preposition or subordinating conjunction	VBN	Verb, past participle
PRP$	Possessive pronoun	NNS	Noun, plural
RB	Adverb	NNP	Proper noun, singular
RBR	Adverb, comparative	NNPS	Proper noun, plural
RBS	Adverb, superlative	PDT	Predeterminer
RP	Particle	POS	Possessive ending
SYM	Symbol	PRP	Personal pronoun
JJ	Adjective	VBP	Verb, non-3rd person singular present
JJR	Adjective, comparative	VBZ	Verb, 3rd person singular present
JJS	Adjective, superlative	WDT	Wh-determiner
LS	List item marker	WP	Wh-pronoun
MD	Modal	WP$	Possessive wh-pronoun
NN	Noun, singular or mass	WRB	Wh-adverb

Penn Treebank tagset

Context-Action Map Table Generator

	attack	*heal*	*move*
default	Attack	Heal	Move
weapon	AtkWeapon		
potion		HealPotion	

```
python generateTable.py game-map.csv GameContextActionMap
```

```
setActionList(                  "attack",          "heal",              "move");
addDefaultContextActions(   new Attack(),      new Heal(),          new Move());
addContextActions("weapon", new AtkWeapon(), null,                null);
addContextActions("potion", null,            new HealPotion(), null);
```

Entity

- Object which can be a potential target / context

```
public class Sword extends Entity {
    public Sword() {
        super("sword");
        setContext("weapon");
    }
}
```

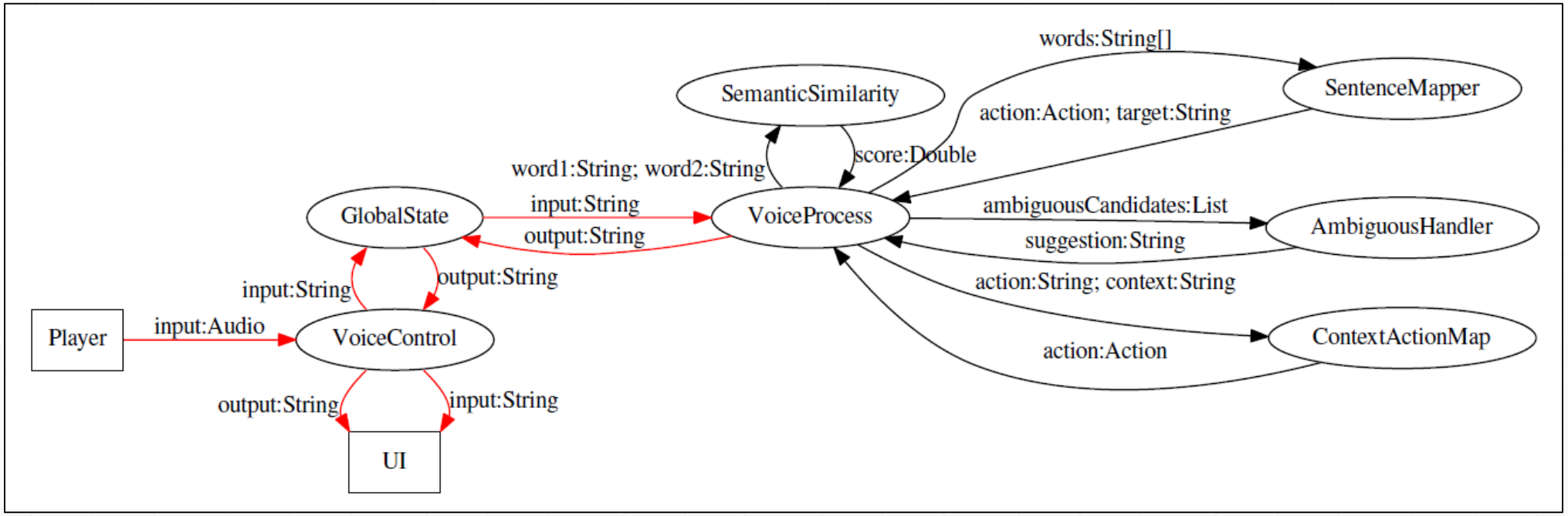
- Build up list of potential targets / contexts in ContextActionMap

Actions

- Wrappers for methods which change the `GlobalState` of the application.

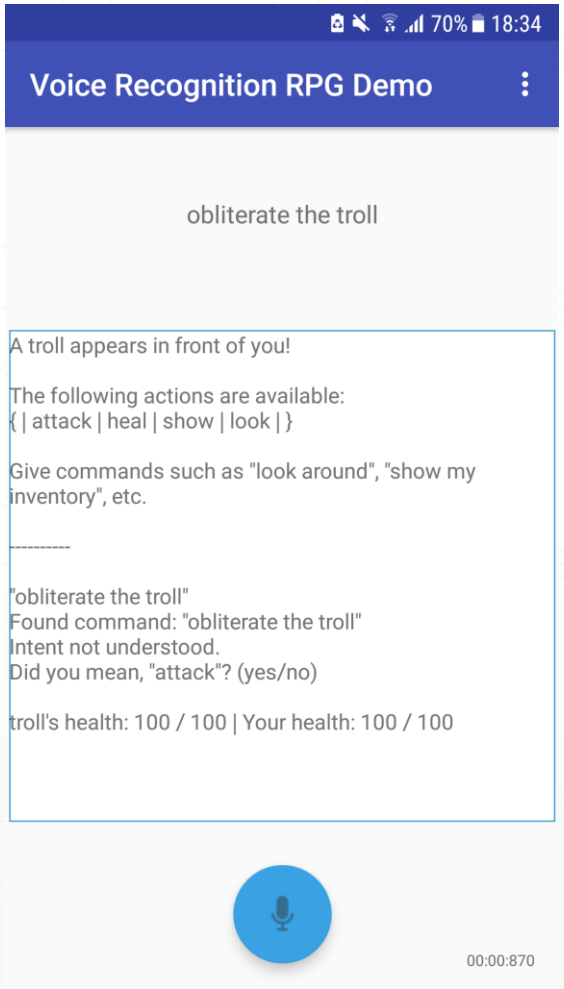
```
1  public class AtkWithWeapon extends Action {
       public String execute(GlobalState state, Entity currentTarget) {
3          //Insert code to attack with weapon (e.g. decrease enemy health)
           //Access current context using Action.getCurrentContext()
5          return "Return a response to the user.";
       }
7  }
```

Data Flow

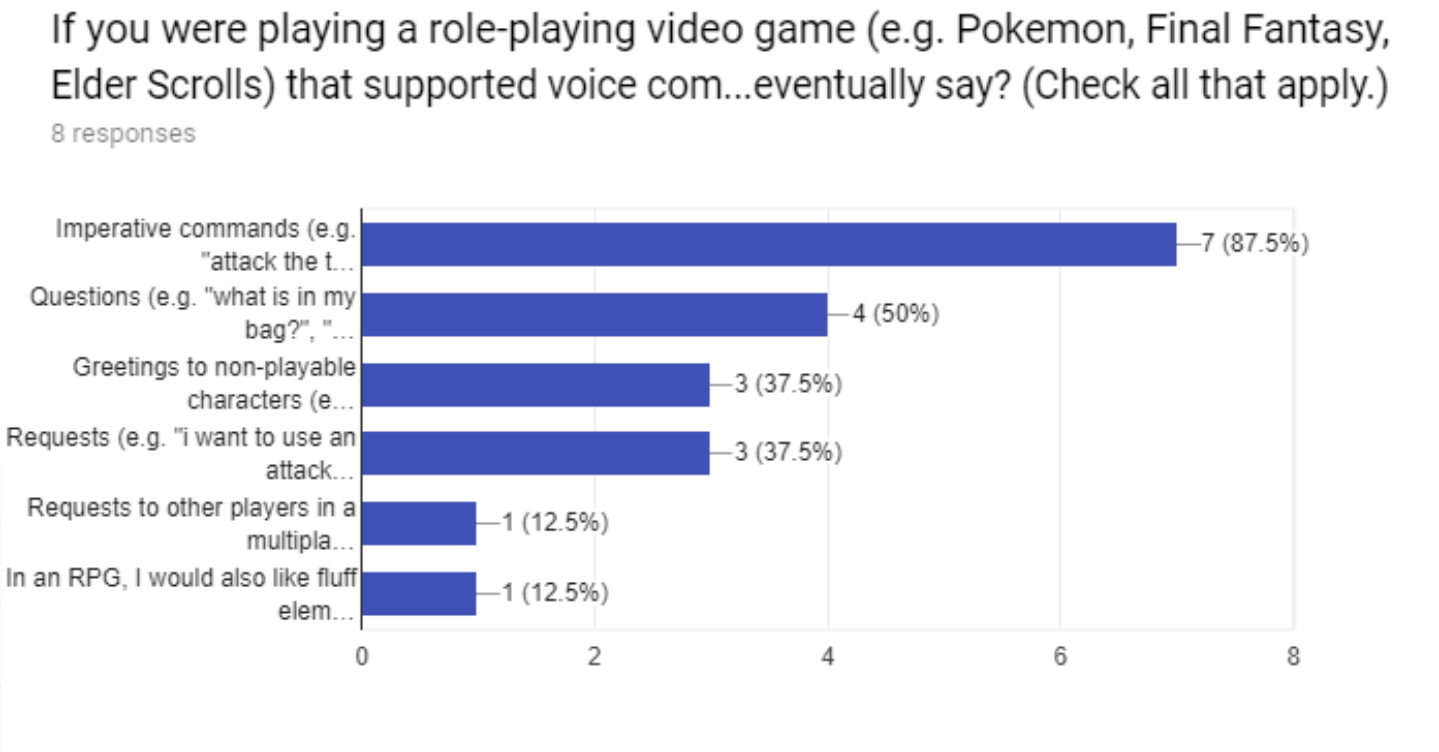

words:String[]
SemanticSimilarity
SentenceMapper
action:Action; target:String
score:Double
word1:String; word2:String
GlobalState
input:String
VoiceProcess
ambiguousCandidates:List
AmbiguousHandler
output:String
suggestion:String
input:String
output:String
action:String; context:String
Player
input:Audio
VoiceControl
ContextActionMap
action:Action
output:String
input:String
UI

Ambiguous Handler

- If action/target/context candidate just below the threshold, marked as ambiguous
- If best candidate below threshold, all ambiguous candidates queried
- Suggestions given to player in order of score
  - Until they say, “yes”
- > 4 suggestions → show all at once



Most RPG Voice Commands are Imperative



Hybrid Methods

Method 1	Method 2	Final Score
FASTLESK	LCH	**74.70%**
LESK	LCH	**74.70%**
WUP	COS	73.49%
LCH	PATH	72.29%
WUP	LCH	71.08%

Sentence-Matching

- If slot-filling grammar fails (i.e. not imperative)
- For questions, greetings, etc.
  - “What actions can I do?”
  - “Hello, how are you?”
- Can add examples of sentences

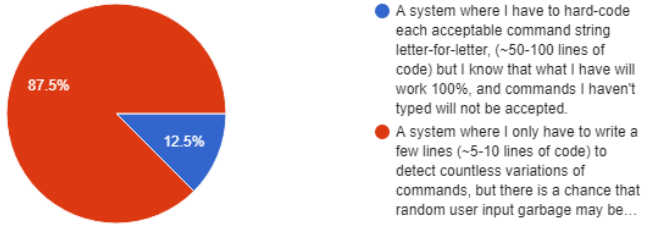
```
addSentenceMatch(new ShowDefault(), "inventory",
        "what is in my inventory",
        "what are the contents of my bag",
        "what items do i have"
);

addSentenceMatch(new ShowDefault(), "actions",
        "what can i do",
        "what are my actions",
        "what are the commands",
        "what action can i do",
        "what are my options"
);
```

Survey Results

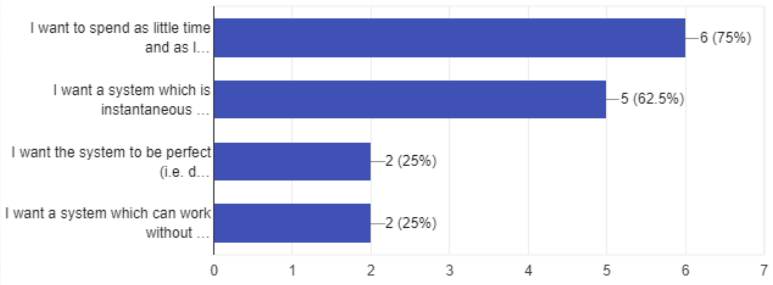
As a developer, when choosing a system for adding voice commands to an application (assuming you have the use...s it), which system would you prefer?

8 responses



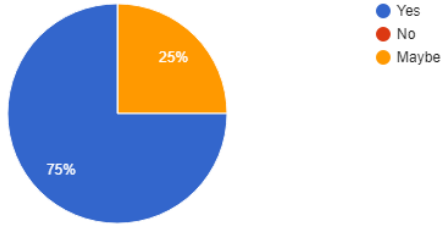
As a developer that wants to add voice commands to an application, which of the following are the most important to you? (Select up to two.)

8 responses



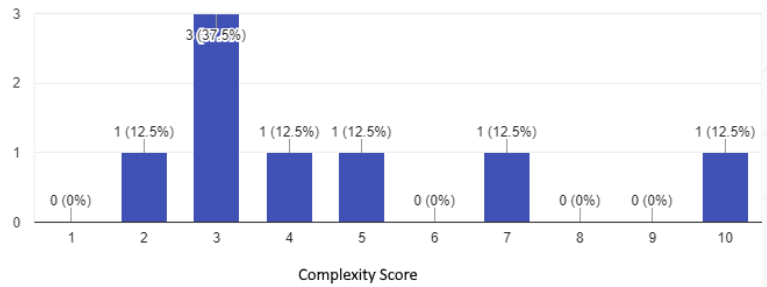
Read the following README (link below) for a Java library that adds voice commands to a project. Is it clear what the library is aiming to do?

8 responses



Consider the following user guide (link below) for the above Java library. How complex are the instructions?

8 responses



Challenges

- Creating the voice recognition system from scratch
  - Ensuring it works completely offline
  - Processing the user’s input in the most efficient and correct way
  - Ensuring it is easy to add voice commands to the game
- Android development issues
  - Issues with compatibility with Java 8