# Discourse and Dialogue Models

James Pustejovsky

August 27, 2020

Slides thanks to Staffan Larsson

# Why build dialogue systems?

- Theoretical purpose: test theories
  - e.g. what kind of information does an agent need to keep track of in order to be able to participate in a dialogue?
  - However, complex system with many components – how to evaluate
- Practical purpose: human-computer interaction

# Why spoken interaction?

- Spoken interaction is the natural way for humans to interact
  - computers should adapt to humans rather than the other way around
  - important to enable systems to interact in a natural way
- Language can be used to convey any message, at any time
  - On a screen, you can only push the buttons shown
  - Less effort for user, who can just say what's on her mind...
  - ...but system then needs to be able to deal with most of the ways that the dialogue may unfold
- Users want hands-free and/or eyes-free use
  - Especially in in-vehicle situations

# History of dialogue systems

- ELIZA (Weizenbaum 1966)
  - text dialogue
  - simulated psychoanalyst
- SHRDLU (Winograd 1972)
  - written dialogue
  - control simulated robot in a blocks world
- TRAINS (Allen et al 1991)
  - spoken dialogue
  - joint planning task
- CSLU Toolkit (McTear 1993)
  - platform for implementing dialogue system applications
  - simple dialogue manager
- Philips train timetable system (Aust et al 1994)
  - speech over phone
  - first deployed system
- Linguatronics (1996)
  - in-car spoken dialogue
  - dialing etc
- VoiceXML (W3C 2000)
  - general platform
  - form-filling dialogue
- Siri (Apple 2009)
  - smartphone-based
  - multimodal
- API.AI, Amazon Alexa (2015)
  - proprietary platforms open for third party development

# Two types of methods in Computational Linguistics

- Rule-based
- Statistical/Machine Learning

# Rule-based methods

*Example:* Interpret English commands in infotainment system

- create a lexicon for English
- write grammar rules for English in the infotainment domain
- write rules relating English sentences to a semantic representation (intents and entities)

# Statistical/Machine Learning methods

*Example:* Interpret English commands in infotainment system

- collect lots of examples of English sentences from the infotainment domain
- annotate sentences with their meanings (intents and entities)
- use machine learning techniques to produce statistical models correlating English sentences with intents and entities

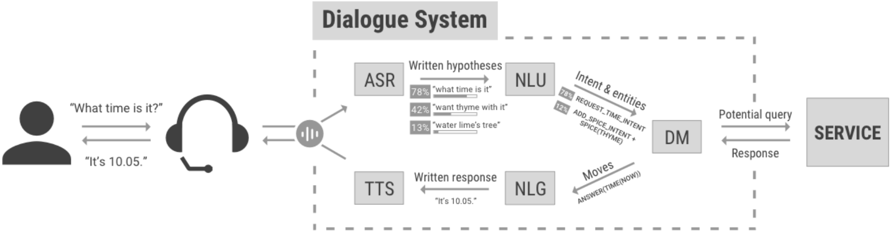
# Comparing rule-based and statistical methods

- Rule-based methods get more exact and correct results, but it can take a lot of work to get them to cover enough data
- Statistical methods cover a lot more data, but they sometimes get things very wrong, in ways that we do not understand

# Hybrid systems

- Hybrid systems attempt to combine both rule-based and statistical methods
- . . . but there are many open research questions concerning the best way to combine the two approaches

# Dialogue systems architecture



# Natural Language Understanding (NLU)

- Extract relevant meaning from text
  - In many systems, meaning consists of "intents" (requested actions) and entities
  - In general in natural language, much more complex meanings can be conveyed: relations, negation, modality, counterfactuals, ...
- Until around 2000, NLU was mostly rule-based
  - A single grammar often used both to govern ASR and to extract meaning from text
- NLU is increasingly based on machine learning, generalising from examples

# Dialogue Management (DM)

- Over the last 5-10 years there has been a focus in academia on statistical methods for dialogue management
- However, the complexity of dialogue management have lead to doubts about the prospects of such methods
- All commercial dialogue managers are more or less rule-based

# Natural Language Generation (NLG)

- Convert output from DM into text
- NLG has so far received much less attention that ASR and NLU
- Many current commercial systems conflate DM and NLG, using simple language-templates with slot values filled in
  - "Calling $NAME's $NUMTYPE number"
- Research has produced more powerful generation techniques that are not being used commercially yet.
- Current approach works okay for simple kinds of dialogue and for syntactically simple languages such as English
- When moving into more complex domains and when localising to more complex languages (e.g. Turkish), NLG will become an issue

# Text-To-Speech (TTS)

- TTS has improved significantly over the last 30 years, reaching almost natural voice quality
- However, there is still plenty of room for improvement
- For example, control over intonation is still a problem
- Example
  - "What **city** do you want to go to?"
  - "London"
  - # "What **city** do you want to go from?"

# Text-To-Speech (TTS)

- TTS has improved significantly over the last 30 years, reaching almost natural voice quality
- However, there is still plenty of room for improvement
- For example, control over intonation is still a problem
- Example
  - "What **city** do you want to go to?"
  - "London"
  - "What city do you want to go **from**?"
- Generating correct intonation often requires some level of understanding of what is being said, and of what has been said before

# Multimodality

- For practically useful dialogue systems, the connection between traditional touch-screen interaction and spoken interaction is important
- Current state of the art in industry is that the user has to choose between "normal" touch-screen interaction and spoken interaction (with a different GUI)
- Problems with this approach:
  - Forces users to abandon what they know for something less known
  - Not possible to mix spoken interaction and touch-screen interaction freely
  - Sometimes, you have to look at the screen
- Instead, systems should enable
  - The same touch-screen interaction regardless of whether speech is enabled or not
  - Users can switch modality anytime
  - Never necessary to look at the screen

# Why is dialogue management important?

- Without a DM, there is no dialogue.
- The user has to give all information that the system needs in a single utterance, which in some cases may be very difficult and cognitively demanding
  - "I want to book a flight from Gothenburg to London on September 2 in the afternoon, coming back on the 10th in the morning, for 2 adults and 2 children aged 5 and 8, with no stopovers and preferably going to Heathrow airport, economy class."
- If any information is left out, there is no way to supply it later.

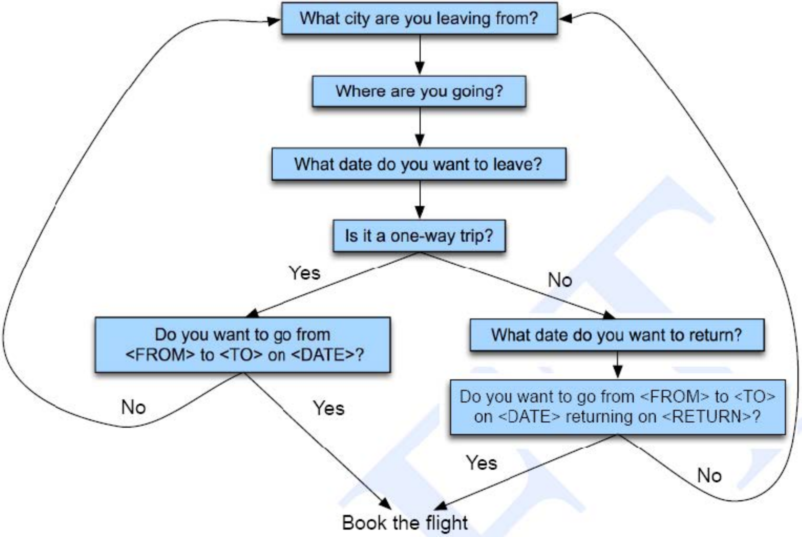
# Why is dialogue management important?

- A dialogue manager makes it possible to have coherent exchanges consisting of several turns
- This means that the user does not have to say everything at once ("the truth, the whole truth and nothing but the truth")
- Instead, the user can say what's on her mind, and the system will ask for additional needed information

# Dialogue Management methods

- Four types of dialogue managers:
  - Finite state-based
  - Form-filling
  - Plan-based
  - Information State

# Finite state-based DM



# Finite state-based DM

- Represents dialogue flow using a finite state machine
    - States: questions to the user
    - Transitions: user responses and resulting actions
    - Also stores answers in variables (<DATE> etc) (not pure finite state)
- Works for system initiative ("single initiative") dialogue
    - System has all the initiative
    - Tends to ignores or misinterpret anything which is not a direct answer to a system question

# Finite state-based DM

- However, human-human conversation is very often "mixed initiative"
  - User may provide unrequested information
  - User may ask a question in response to a question
  - ...
- To deal with mixed initiative for $n$ questions, $\sim 7n^2$ transitions are needed (for $n = 20$, 2800 states)
- These all need to be created and maintained by the dialogue developer

# Form-based dialogue management

- Form = slots and values
- Relies on the structure of a form to guide the dialogue.
- Provides some aspects of mixed initiative dialogue
- Asks the user questions to fill slots in the frame
  - but allow the user to guide the dialogue by giving information that fills other slots in the frame
- Each slot may be associated with a question to ask the user, following type:
  - ORIGIN CITY "From what city are you leaving?"
  - DESTINATION CITY "Where are you going?"
  - DEPARTURE TIME "When would you like to leave?"
  - ARRIVAL TIME "When do you want to arrive?"

# Form-based dialogue management

- DM asks questions to the user, filling any slot that the user specifies...
- ...until it has enough information to perform a data base query, and then return the result to the user
- If the user happens to answer two or three questions at a time, the system has to fill in these slots and then remember not to ask the user the associated questions for the slots.
- Does away with the strict constraints that the finite-state manager imposes

# Form-based dialogue management

- VoiceXML
  - Voice Extensible Markup Language
  - an XML-based dialogue design language released by the W3C,
  - very simple mixed-initiative
  - form-based architecture
  - grammar-based ASR and NLU
- Most if not all systems on the market are more or less form-based (Siri, Google Assistant, etc.)
  - Statistical NLU has replaced grammars

# Plan-based DM

- Popular 1980's-1990's
- View dialogue as planning and plan-recognition
- Highly general approach, can handle very complex dialogues (in principle)
- However:
  - Adapting such approaches to individual domains is very labour-intensive
  - Systems are very brittle and tend to break easily

# Information State approach

- Goal: explore the space between finite-state/form-filling approaches (robust but limited) and plan-based approaches (capable but brittle and labour-intensive)
- Key component: a rich Information State, representing the state of the dialogue so far
- Deal with dialogue beyond form-filling in a robust way:
  - Dealing with multiple forms
  - Comparing alternatives ("negotiative dialogue")
  - General and versatile approaches to confirmation, turn-management and other basic dialogue phenomena
  - Instructional dialogue (e.g. technical manuals)
  - Problem-solving dialogue (e.g. putting together an itinerary)
- Important principle: "Separation of concerns"

# Information State approach: separation of concerns

- Keep the following types of knowledge separate:
  - How to deal with the domain (domain knowledge)
  - How to speak about the domain (linguistic knowledge)
  - How to deal with dialogue (DM)
- Advantages
  - Simpler and faster development of new applications/domains, since only domain knowledge needs to be added
  - Simpler and faster localisation of applications to new languages, since only language knowledge needs to be added
  - Cumulative development of dialogue management since all DM improvements become available in future applications $\Rightarrow$ high quality DM across applications

# Information State approach: Multiple forms

- Some domains require the ability to deal with multiple forms, e.g. for a travel agency application:
  - general route information ("Which airlines fly from Boston to San Francisco?")
  - information about airfare practices ("Do I have to stay a specific number of days to get a decent airfare?")
  - questions about car or hotel reservations
- Since users may want to switch between forms (in principle at any time), the system must be able to
  - disambiguate which slot of which form a given input is supposed to fill
  - switch dialogue control to that form
  - return control to previous form once the "embedded" form is done

| | Industry | Academia |
|---|---|---|
| 1990 | Interactive Voice Response<br>▶ Finite state automata (FSA) | Rule-based systems<br>▶ Finite State-based, Form-filling, Plan-based DM<br>▶ Rule-based NLU<br>▶ Low quality ASR |
| 2000 | VoiceXML<br>▶ Finite-state-based, form-filling dialogue<br>▶ Rule-based NLU<br>▶ Grammar-based ASR | Information State Approach to DM<br>▶ Explore middle ground between form-filling and plan-based DM<br>▶ E.g. negotiative dialogue<br>▶ Separation of concerns |
| 2010 | Conversational assistants<br>▶ Form-filling dialogue<br>▶ Rule-based DM<br>▶ ASR gets a lot better | Machine learning approaches<br>▶ POMDP<br>▶ Reinforcement learning<br>▶ Back to form-filling dialogue<br>▶ Hardware advances for ML |
| 2017 | Development platforms<br>▶ Form-filling dialogue<br>▶ Rule-based DM<br>▶ ML for NLU, increased robustness | The pendulum swings back?<br>▶ Increased interaction with Industry<br>▶ Trend: need to move beyond form-filling |

# Machine learning vs. rule-based methods for dialogue systems

- Machine learning has proven useful for ASR and NLU, which are about extracting a meaningful message from a noisy signal
- Less useful for producing coherent responses (DM, NLG)
  - Machine learned methods are inherently unpredictable, but we often want the output from the system to be predictable (and debuggable)

# Machine learning vs. rule-based methods for DM?

- Dialogue management has a huge state space compared to ASR and NLG, so a lot of (expensive) data is needed for machine learning
- Has proven very hard to get beyond form-based DM
- Keynotes at recent major conferences (SigDIAL, Interspeech) have made a case for revising rule-based DM and try to combine with ML, rather than trusting ML to solve everything

# The future: Academia

- The pendulum is swinging back from purely ML approaches to DM, and there will be more work on hybrid approaches combining rule-based and ML methods for DM
- Theoretical work on human-human dialogue has made progress, and this needs to feed into DM research
- With more complex dialogue types comes higher demands on NLG and information presentation
- Work on robotics and dialogue will move towards embodied and situation-aware dialogue systems that can see what the user can see, and talk about it
- As systems become exposed to more diverse and less predictable environments, they will need to be able to learn language from users; foundational research is underway

# The future: Industry

- Dialogue is coming into view, but has so far not received a lot of attention compared to ASR and NLU; this will eventually change
  - To some extent, dialogue can help with NLU problems, but this has yet to be exploited
- There will be a race to handle more complex types of dialogue
- Progress has been made on tools for building simple apps/skills; these need to be extended to work with more complex dialogue types
- For in-vehicle systems, managing cognitive load will be important
  - There is relevant academic research, e.g. about interrupting and resuming dialogue, and system-initiated dialogue

# Natural Language Understanding For Dialogue Systems

User: ”I need a train ticket to Copenhagen.”

*[ intent: book_travel,*
*slots: {*
*destination: ”Copenhagen”,*
*means_of_transport: train*
*} ]*

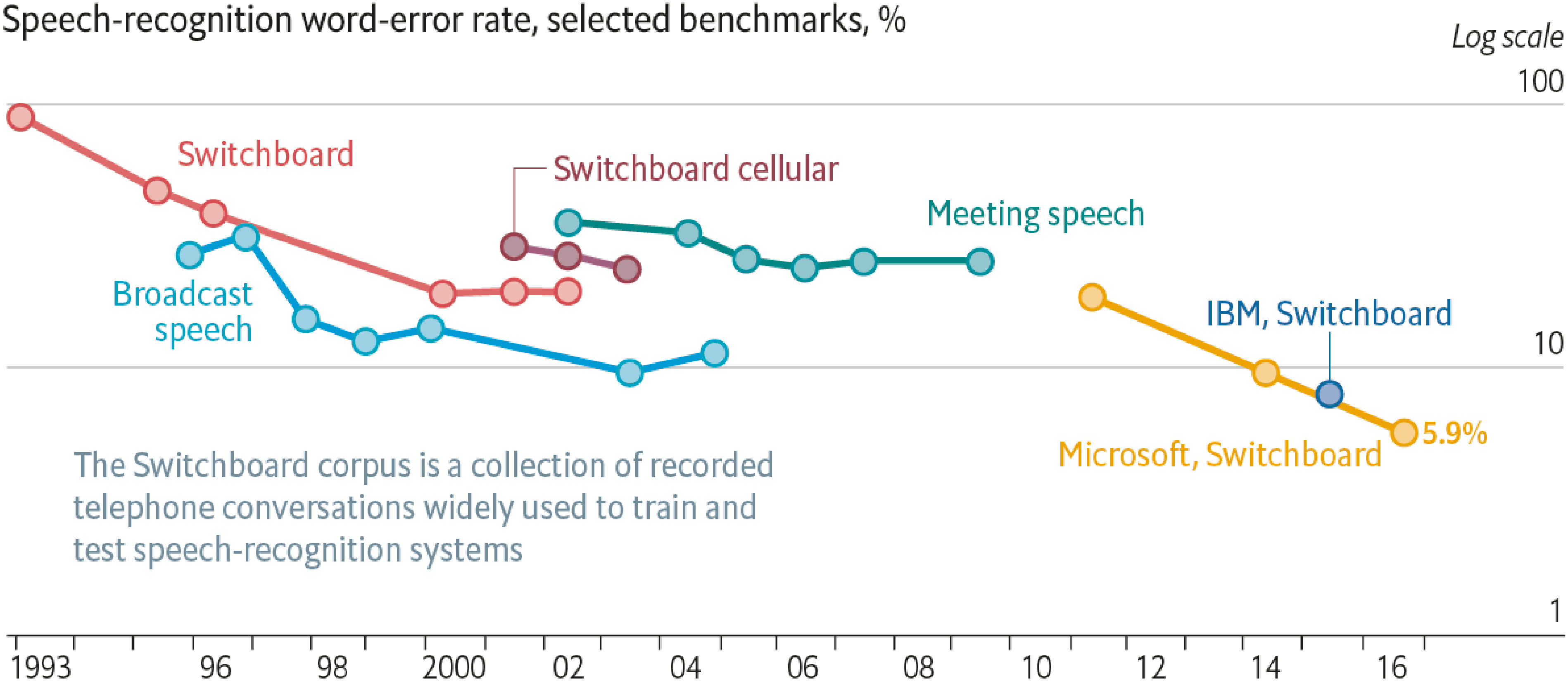
System: ”Okay, at what time?”

# Outline

- NLU vs ASR
- NLU in relation to NLP in general
- Desired properties of NLU for DS
- Implementing an NLU component
- Evaluating and improving performance
- Current research challenges

# NLU vs ASR: State of the Art

Speech-recognition word-error rate, selected benchmarks, %




Sources: Microsoft; research papers

# How about NLU?

- No established benchmark
- … but various benchmarks for related NLP problems
- In existing dialog systems, NLU often performs worse than ASR

# Example of state-of-the-art NLU

# Related NLP problems

Sentiment analysis

Intent classification

Semantic role labeling

Semantic similarity estimation

POS tagging

Coreference resolution

Parsing

Entity extraction

Irony detection

# Desired properties of NLU for DS

- Output mappable onto dialog manager’s input representation (e.g. as intents and slots)
- Tolerates noise (e.g. disfluencies, ASR misrecognitions)
- Estimates confidence / probability
- Handles semantic ambiguity
- Can generalize from given examples to unseen input

# Example: Noisy input

User: ”is the train from Göteborg late?”
ASR: ”the train from Göteborg yet”

*[*
*{ intent:* ***book_travel****,*
*slots: { departure: ”Göteborg” },*
*confidence:* ***0.64*** *},*

*{ intent:* ***get_delay_info****,*
*slots: { departure: ”Göteborg” },*
*confidence:* ***0.47*** *},*
*]*

# Example: Ambiguity

User: ”next”

*[*
*{ intent:* ***next_audio_track****,*
*slots: {},*
*confidence:* ***0.64*** *},*

*{ intent:* ***next_cooking_instruction****,*
*slots: {},*
*confidence:* ***0.36*** *}*
*]*

**Uncertainty can be correctly disambiguated by dialogue manager, e.g. by using dialogue context.**

# Semantic representation

- Current paradigm:
  - Intents (requests and questions)
  - Slot values (answers)
- May support multiple hypotheses
- May contain confidence/probability

# Semantic representation

”call John”


```
{
  'entities': [
    {'entity': 'predicate:selected_contact_to_call',
     'value': 'John',
     'confidence': 0.92}
  ],
  'intent_ranking': [
    {'confidence': 0.65, 'name': 'action::call'},
    {'confidence': 0.10, 'name': 'question::phone_number'},
  ]
}
```

# Semantic representation

- Not supported by current paradigm:
  - Other kinds of dialogue acts, e.g. feedback (”**okay**”)
  - Polarity / negations (”**not** Paris”)
  - Combined intents (”turn off the lights **and** play some disco music”)
  - Anaphora (”call **him**”)
- Can be worked around to some extent
  - E.g. special intents for other dialogue acts and negations

# Implementing an NLU component

- Use existing service
  - DialogFlow, Wit.ai, IBM Watson Assistant, Amazon Lex, Microsoft Luis, Recast.ai …
- Use software library
  - NLTK, Rasa NLU, Spacy, Duckling, scikit-learn ...
- Build from scratch

# Implementing an NLU component

- Additional option: Combine NLU with DM and NLG in a trainable end-to-end DS
- Typical approach: Train neural network on input and output utterances
- Examples: Wen et al (2016), Google Duplex
- Very difficult to design or control
- May be feasible for very small domains or social conversation

# Existing NLU service: Demo

- https://wit.ai

# Using existing NLU services

- Pros:
  - Easy to get started
  - Developer-friendly interfaces
- Cons:
  - Black boxes: Unclear how the NLU works
  - Difficult to improve / extend
  - Limited semantic representation
  - Behaviour may suddenly change

# Building an NLU: Approaches

- Rule-based
  - Context-free grammar
  - Regular expressions
- Statistical
  - Bag of words
  - Support vector machine
  - Neural network (recurrent/convolutional)
  - Word/sentence embedding

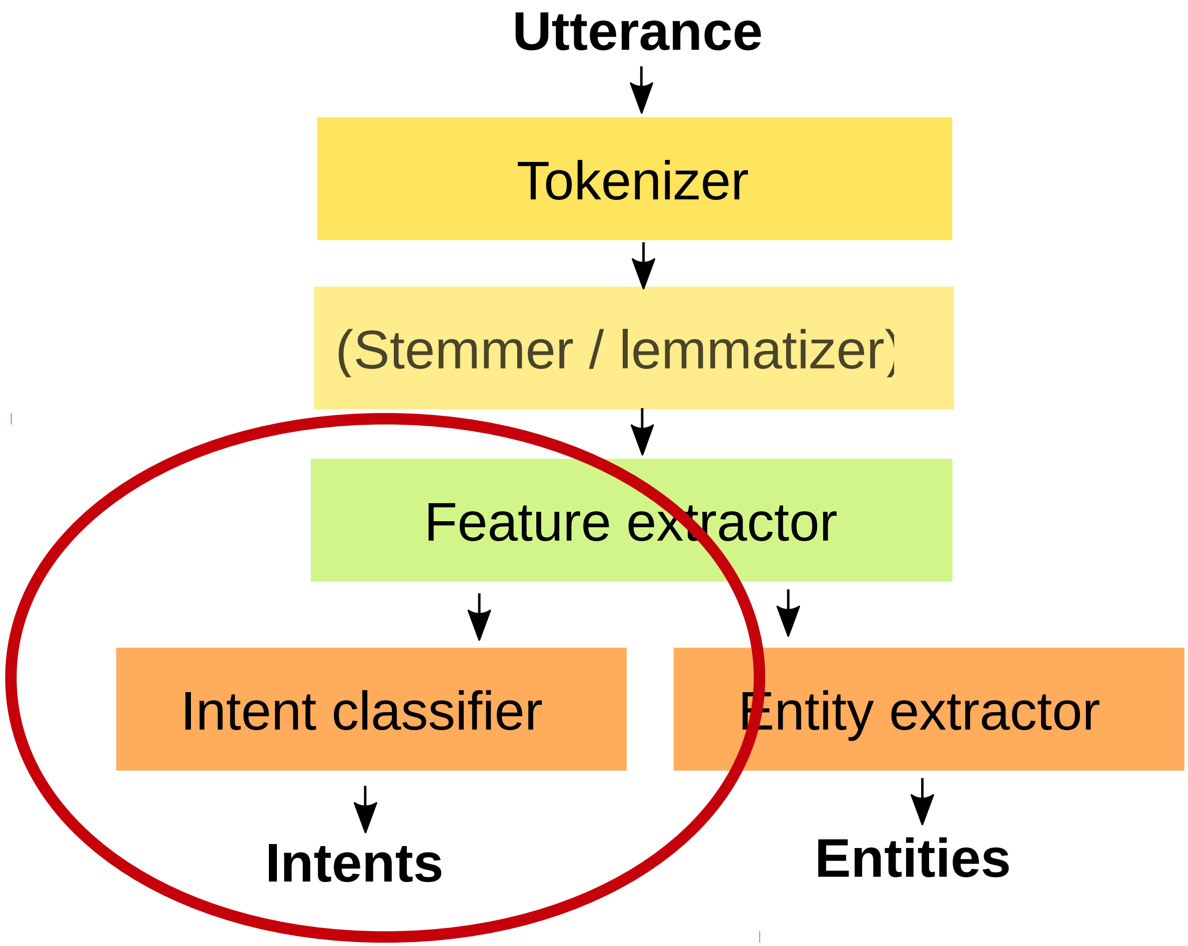
# Rule-based approaches

- Pros:
  - High transparency (easy to understand and troubleshoot)
- Cons:
  - Difficult to deal with noise
  - Cannot generalize to unseen input
  - Binary outcome (success or failure, no confidence/probability)

# Statistical approaches

- Pros:
  - Can deal with noise
  - Can generalize to unseen input
  - Can estimate confidence/probability
- Cons:
  - Low transparency (difficult to troubleshoot)
  - False positives can be difficult to detect
  - May require plenty of training data
  - May require tedious hyperparameter tuning
  - Training may have high footprint (memory, CPU)

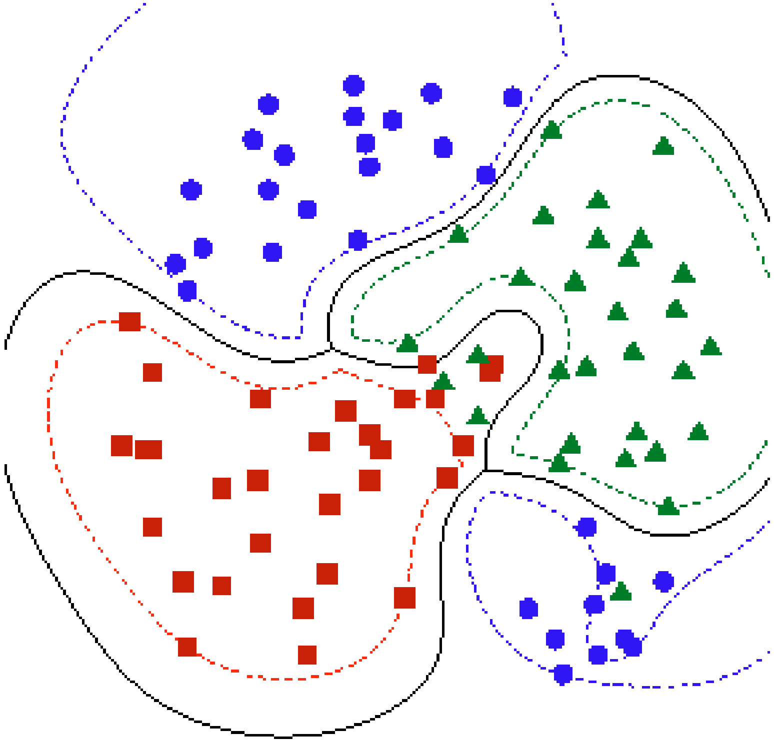
# Statistical approaches



# Statistical intent classification

- Assumption: For any intent, there are linguistic regularities among the phrases that speakers use to express the intent
- Purpose of classifier: to learn such patterns in order to predict the intent from a sequence of words

# Statistical intent classification



# Statistical intent classification

- Feature extraction
  - Bag of words
  - Word vectors
  - Sentence vectors
- Classification
  - Naive Bayes
  - Support vector machines
  - Neural networks

# Bag of words

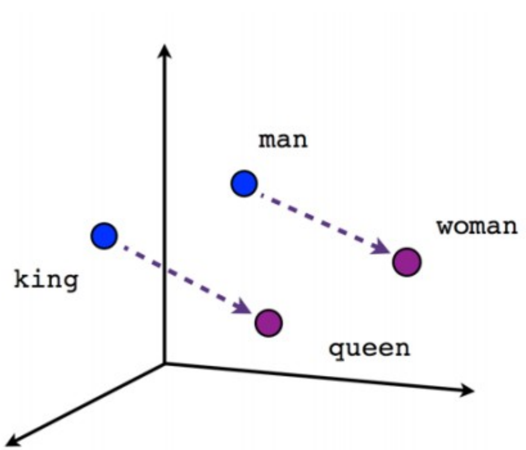
- Utterance featurized as vector of frequency measures
- Example: ”turn on the light” →
  `[ … 0 0 0 .7 0 0 0 .8 0 0 0 .7 0 0 .8  … ]`
- Vector has one component per word in the dictionary
- The dictionary stems from the training data
- Stemming or lemmatization often used (cats → cat)

# Bag of words

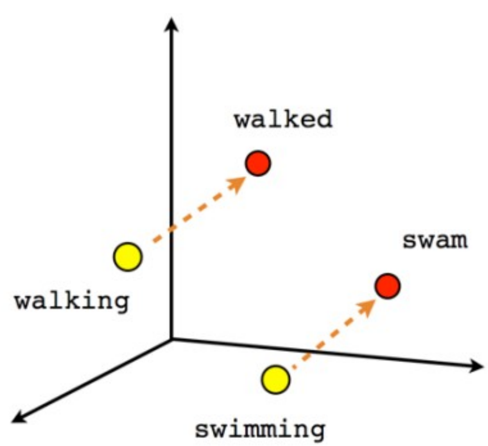
- Pros
  - Simple
- Cons
  - Doesn’t handle polysemy
  - Treats words as independent features
  - Disregards structure, e.g. word order
    - ... but can be addressed with n-grams
  - Can’t handle out-of-vocabulary words
  - Vector size grows with size of training data →
    - Sparsity
    - Complexity

# Word vectors

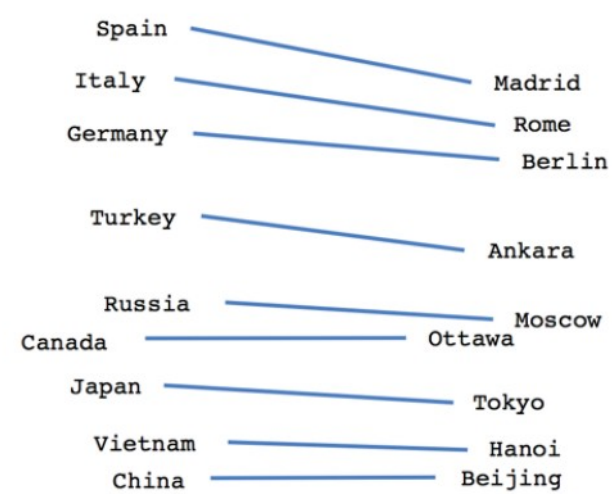
- Word featurized as vector representing point in a word vector space
- Vector space captures semantic relations between words




Male-Female




Verb tense




Country-Capital

offended
strange
alter
monster
sensible
bought
met
tremble
afraid
buy
sick
nt
help
n't
settle
dying
lately
aid
send
pay
thanks
fate
slave
prayers
forgotten
unknown
kill
aside
artful
speak
wrote
spoken
free
fall
sober
gone
walking
instant
flight
pointed
spare
fine
food
coming
set
fill
supper
doing
famous
twice
blood
laid
arise
placed
affords
feel
town
called
arm
corner
legs
joined
gift
cut
finished
state
praise
reply
big
eye
following
bare
employed
firm
mean
permitted
firmly
heavy
full
order
bounds
apt
particularly
weight
things
foundation
action
chiefly
directly
originally
countenance
quite
degrees
painted
filled
strength
situation
passion
tsne V1
tsne V2
14
12
10
8
6
4
2
0
-2
-4
-6
-8
-10
-12
-14
-16
-15
-10
-5
0
5
10
15

# Word vectors

- Theoretical basis: Semantically related words have similar contexts (neighbouring words)
- **Count-based**
  - E.g. Latent Semantic Analysis
  - Reduce dimensionality of co-occurance matrix
- **Predictive**
  - E.g. predict word from context
  - Often called *neural*, since they use neural networks

# Word vectors

- Pros
  - Reflect word ”meaning” (in some sense)
  - Enable classification of words outside training vocabulary
  - Fixed vector size
  - Dense representation
  - Pre-trained models available
- Cons
  - Don’t handle polysemy
  - May reproduce cultural biases
  - Training custom vectors requires plenty of data and time

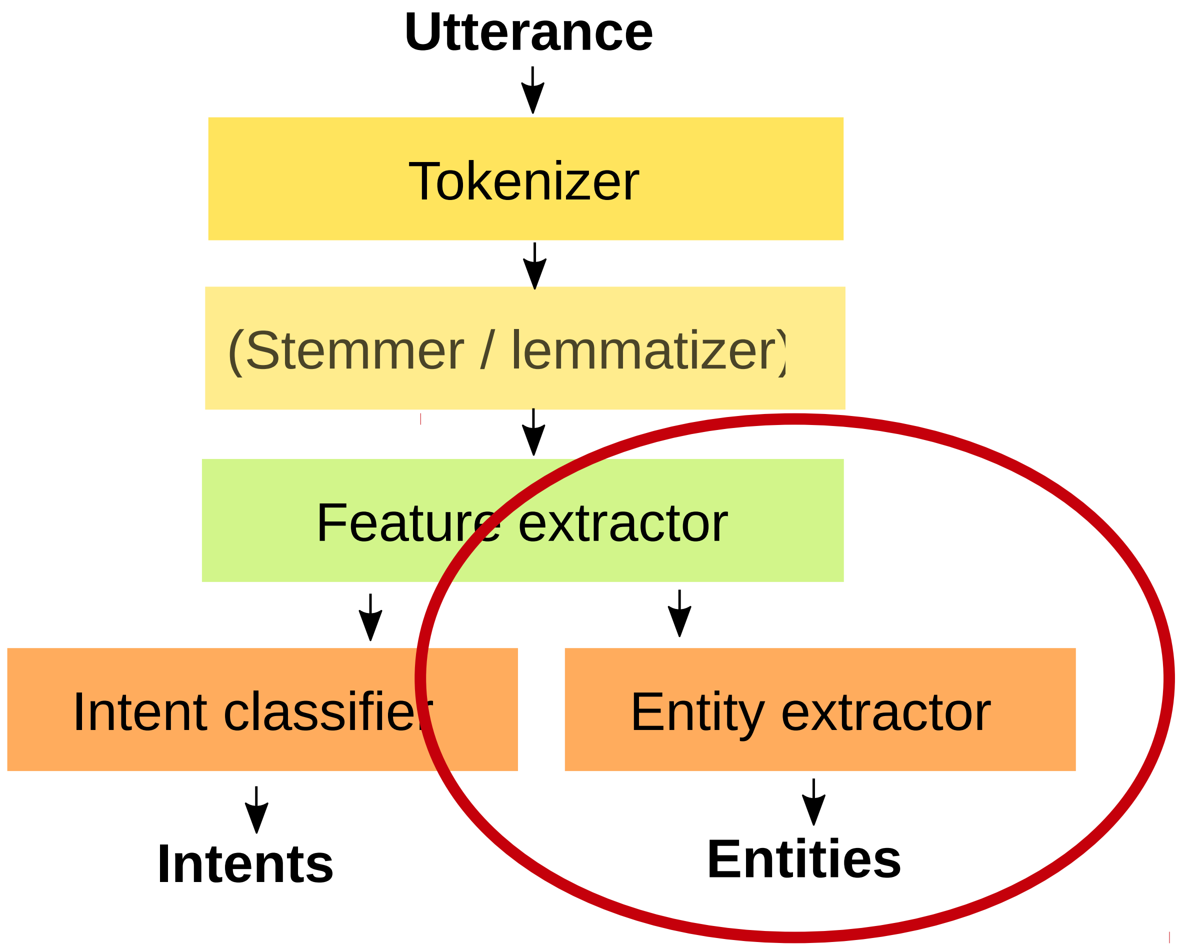
# Sentence vectors

- New approach for text/intent classification
- Similar to word vectors, but embed whole sentences instead
- Examples:
  - Skip-thoughts
  - StarSpace

# Statistical intent classification

- Feature extraction
  - Bag of words
  - Word vectors
  - Sentence vectors
- Classification
  - Naive Bayes
  - Support vector machines
  - Neural networks

# Statistical approaches



# Entity extraction

- Examples of entities:
  - **Named** (person names, cities, organizations etc.)
  - **Date/time**
  - **Duration**
  - **Numbers and ordinals**
  - **Amount of money**
  - **Temperature**
  - **URL**
  - **Phone number**
  - **Domain-specific** (e.g. ”home/mobile number” in phone domain)

# Entity extraction for DS

- Identify known value
  - ”call **John**”
  - ”I need a ticket to **Copenhagen**”
  - ”I want to travel **next Monday morning**”
- Identify unknown value
  - ”I need directions to **Engelbrektsgatan 30A**”
- Detect propositionality
  - ”I want a ticket **from Gothenburg** **to Copenhagen**”

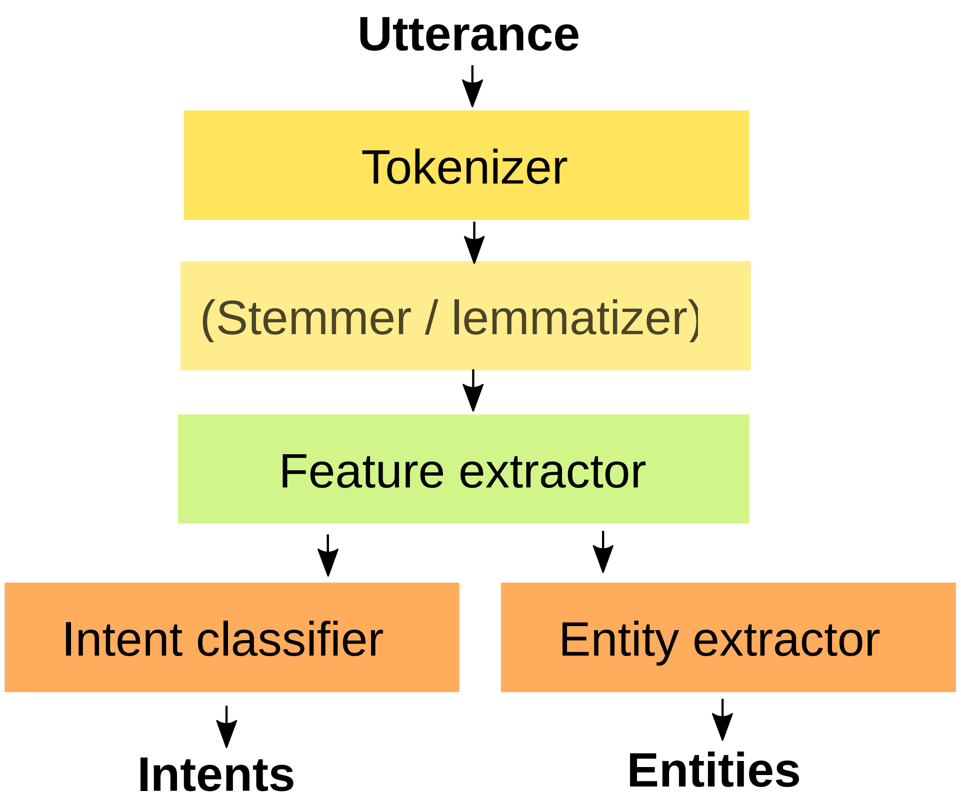
# Entity extraction challenges

- Not only label word correctly, also parse/interpret!
  - ”**October 21st at five PM**” → *datetime(”2018-10-21T05:00:00”)*
- Contextual ambiguity / deixis
  - ”**next Monday**”
- Compositionality / granularity
  - ”**5000 dollars**”: single entity (amount of money) or composition of entities (amount, currency)?
  - ”**5000**”: amount of money?
- Over-generalization
  - ”book a meeting on **Monday at fifty o’clock**”
  - ”give me directions to **eh no forget it**”
  - ”remind me on Saturday, **no I mean on Sunday, to ...**”

# Entity extraction

- Rule-based methods:
  - Keyword spotting
  - Regular expressions
  - CFG
- Statistical methods:
  - **Conditional random field** (CRF)
  - **Probabilistic context-free grammar** (PCFG)

# Putting it all together



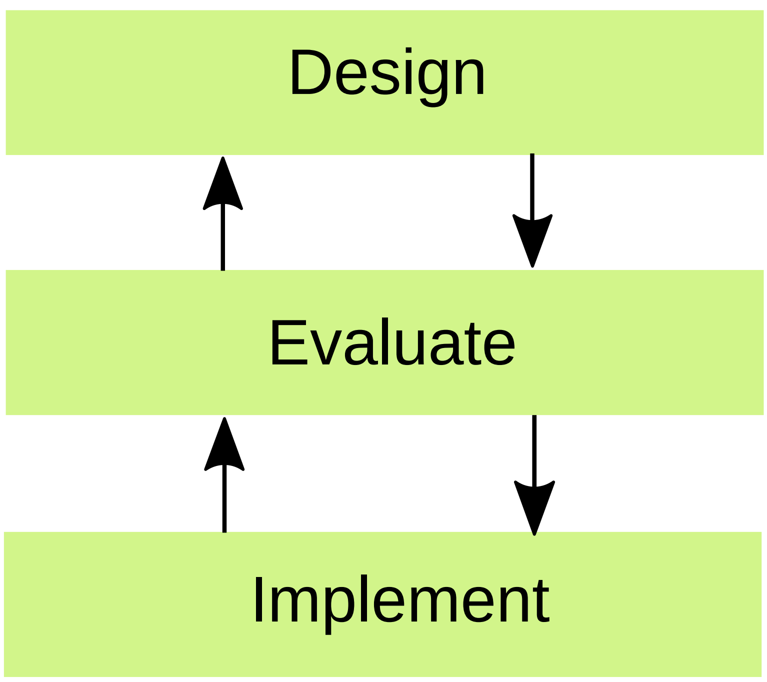
# Putting it all together

- Existing services
  - DialogFlow, Wit.ai, IBM Watson Assistant, Amazon Lex, Microsoft Luis, Recast.ai …
- Software libraries
  - NLTK, Rasa NLU, Spacy, Duckling, scikit-learn ...
- Custom implementation / build from scratch

# Putting it all together

- Example: **Talkamatic Dialogue Manager**
  - Rule-based: Grammatical Framework (Parallel Multiple Context-Free Grammar)
  - Statistical: Rasa NLU

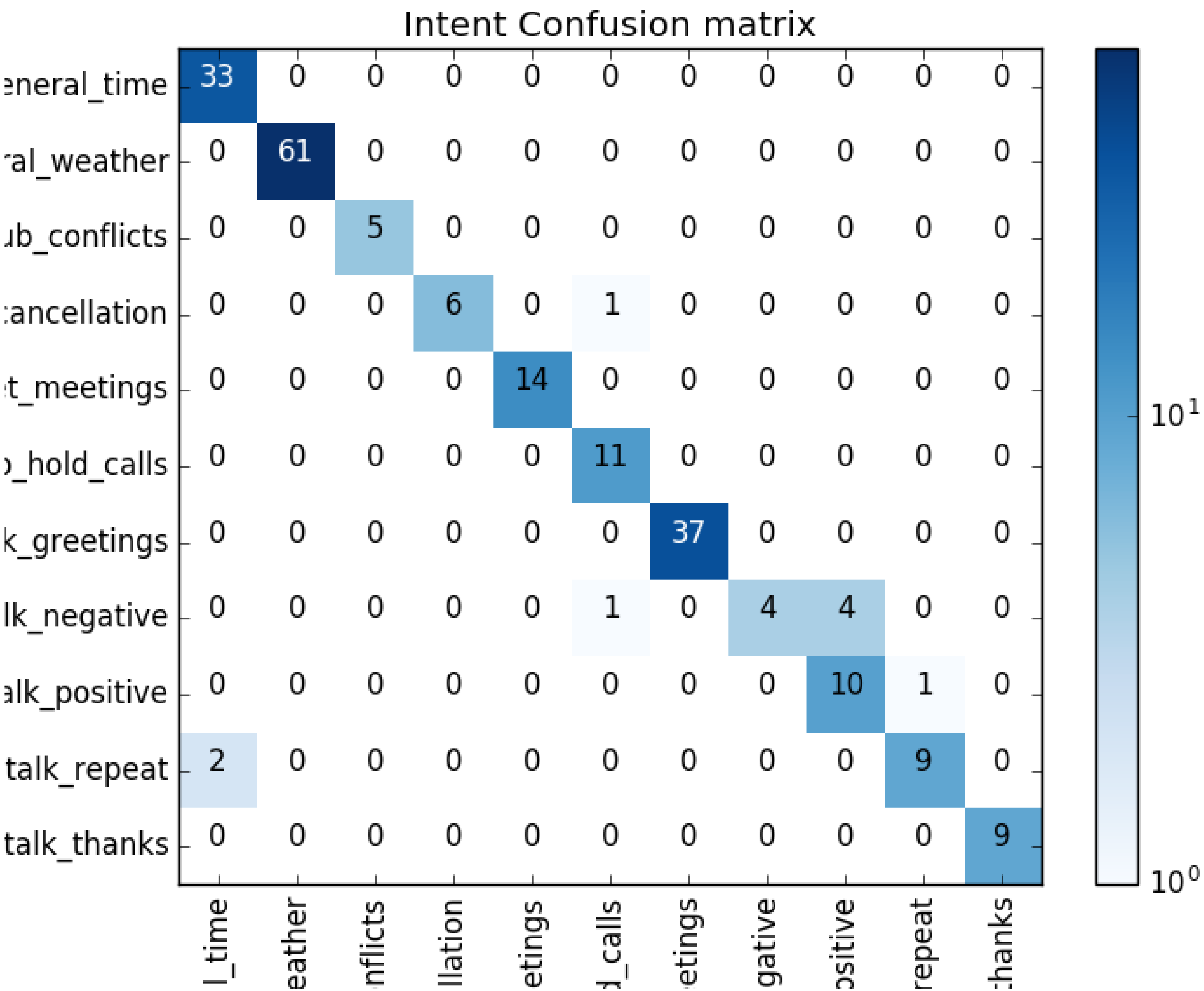
# Evaluating and improving




- Design:
  - Formulate expected interactions
- Evaluate:
  - Measure NLU performance
  - Perform user testing
- Implement
  - Modify/extend/replace
    - Feature extractor
    - Intent classifier
    - Entity extractor

# Measuring NLU performance

- Intents
  - Confusion matrix
  - Cross-validation
  - Precision, recall, accuracy
- Entities
  - Precision, recall, accuracy



Intent Confusion matrix
eneral_time
ral_weather
ub_conflicts
cancellation
t_meetings
_hold_calls
k_greetings
lk_negative
alk_positive
talk_repeat
talk_thanks
33 0 0 0 0 0 0 0 0 0 0
0 61 0 0 0 0 0 0 0 0 0
0 0 5 0 0 0 0 0 0 0 0
0 0 0 6 0 1 0 0 0 0 0
0 0 0 0 14 0 0 0 0 0 0
0 0 0 0 0 11 0 0 0 0 0
0 0 0 0 0 0 37 0 0 0 0
0 0 0 0 0 1 0 4 4 0 0
0 0 0 0 0 0 0 0 10 1 0
2 0 0 0 0 0 0 0 0 9 0
0 0 0 0 0 0 0 0 0 0 9
l_time
eather
nflicts
llation
etings
d_calls
etings
gative
ositive
repeat
thanks
$10^1$
$10^0$

# Research challenges

- Anaphora
- Literal content
- Benchmarking

# Anaphora and common-sense reasoning

User: ”my printer won’t print my document”
System: ”Okay, I will try to help you.”
User: ”is **it** in the wrong paper size?”

document

User: ”my printer won’t print my encrypted PDF”
System: ”Okay, I will try to help you.”
User: ”is **it** too old?”

printer