Spoken dialog systems

Índice

Types of dialog systems

Task-oriented systems: restricted domain, medium size of the vocabulary. Mixed initiative. Short dialogs. Multimodality. developed to help the user solve a specific task as efficiently as possible.

Examples: restaurants, train or flights booking, bus tiemetables…

Conversational agents: open-domain/specific-domain, no specific task. Mixed initiative. Long dialogs.

Examples: information seeking, management of agenda, health personal assistance, e-learning, chats,…

Question Answering: specific/generic domain. User initiative. Originally single turn, although currently there are multiple turn systems.

Initiative

Initiative: Who has control of conversation

User initiative: similar to question answering.

System initiative: dialog directed by the system.

Mixed initiative: initiative can shift between system and user.

User initiative

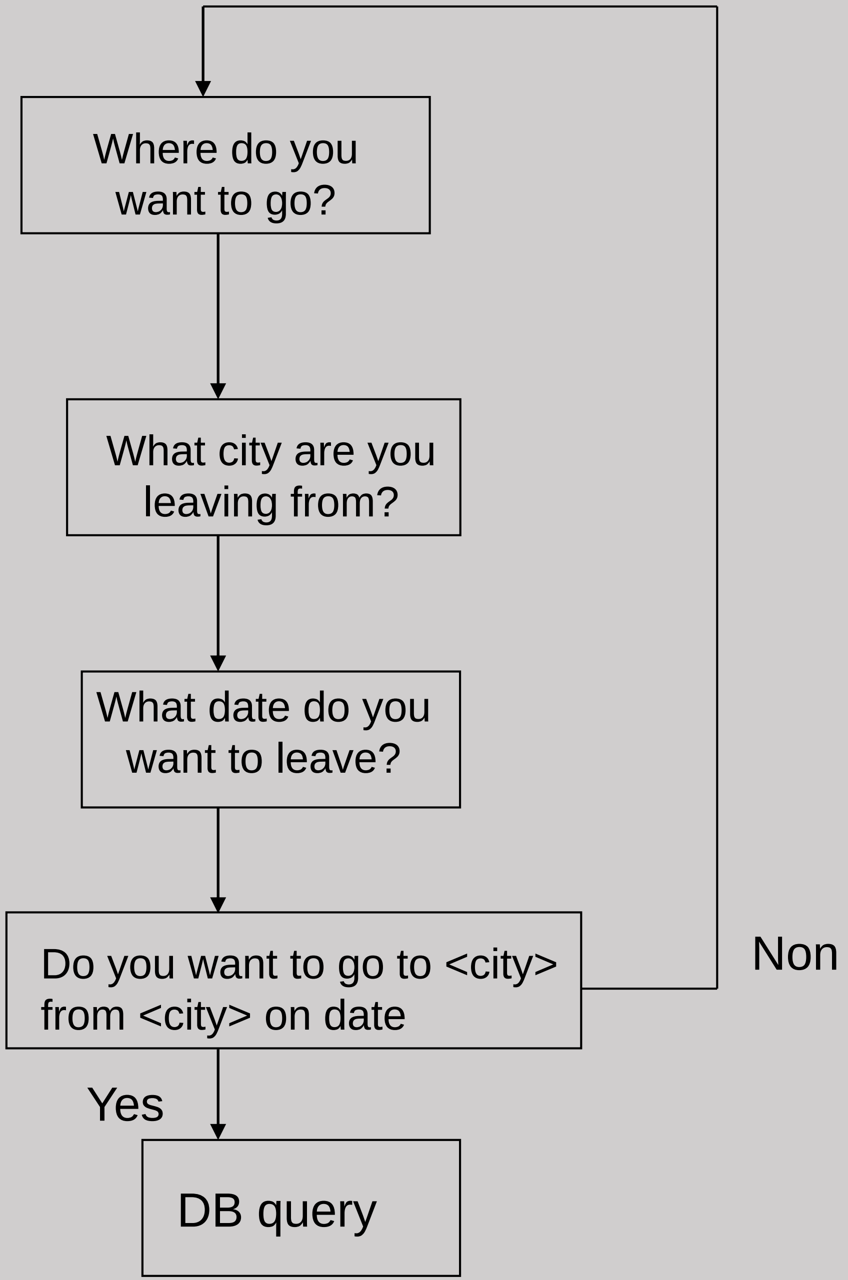
Who is the president of United States?

Where is the Eiffel Tower?

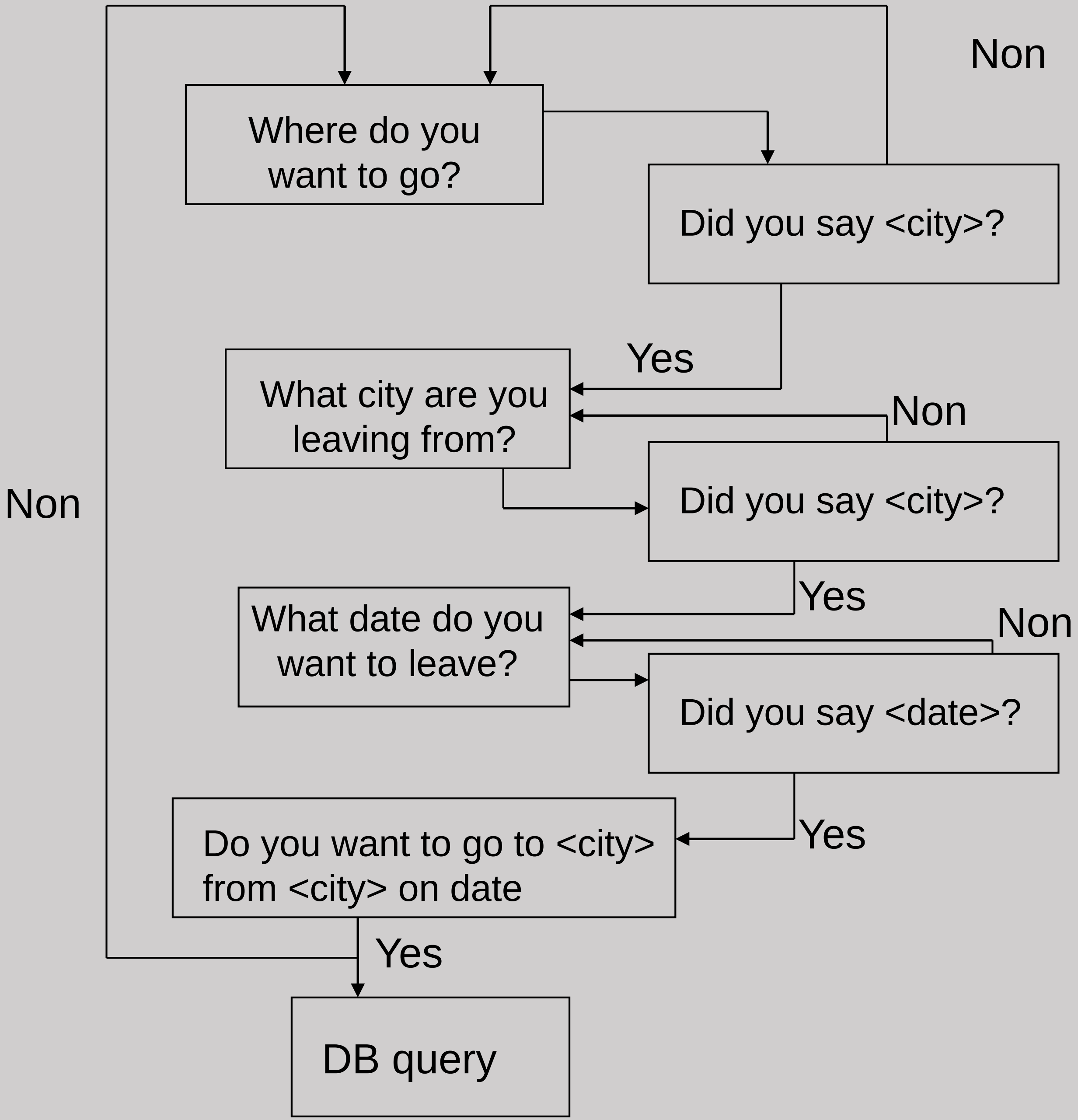
Who won the Europa leage in 2021?

What is the magnitude of earthquake in Pakistan?

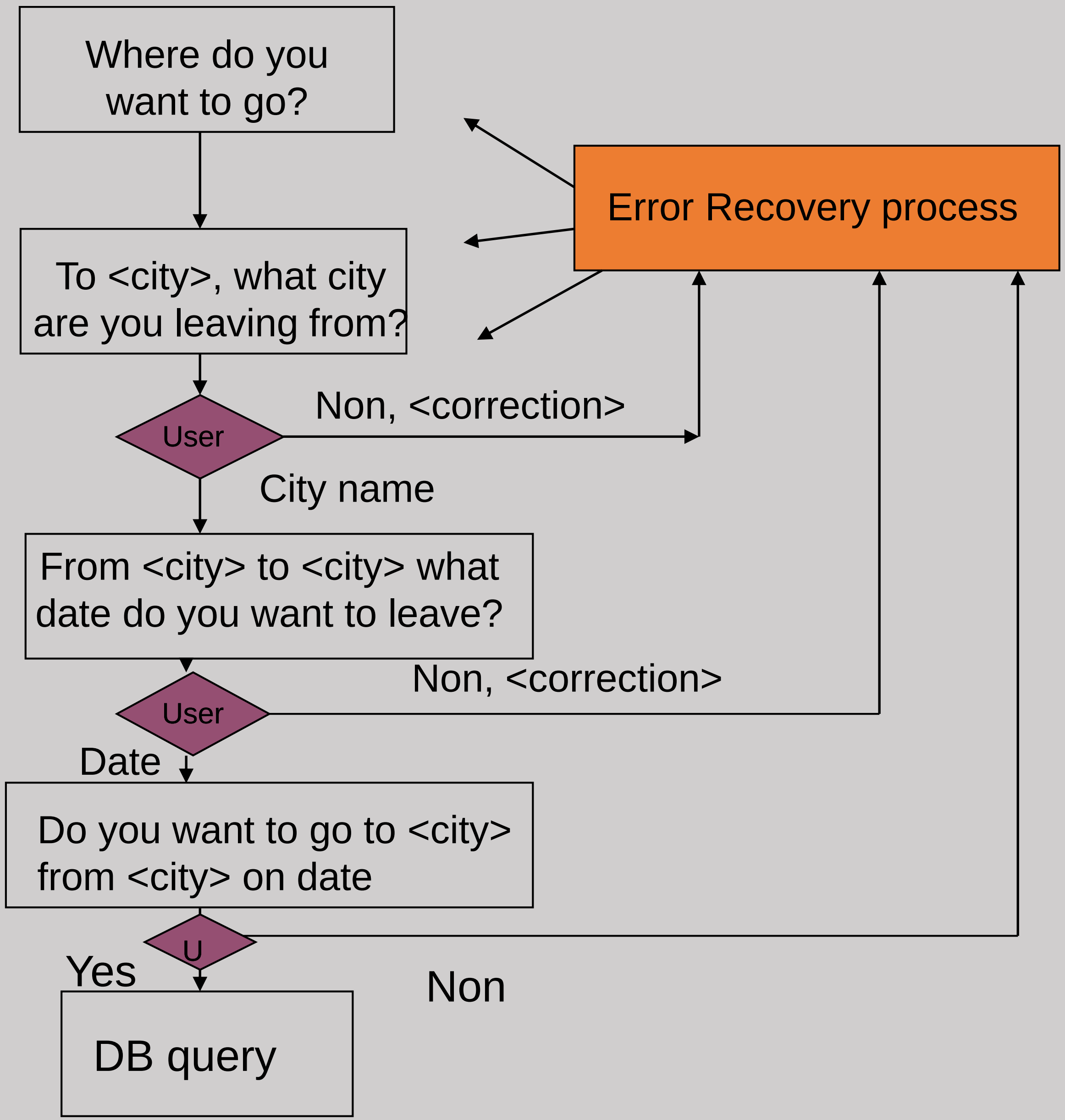
System initiative



System initiative (Explicit Confirmation)



System initiative (Implicit Confirmation)



Mixed initiative

S1:	Hello. You've reached the [Communicator]. Tell me your full name
U2:	*hi i'd like to fly to seattle tuesday morning*
S3:	Traveling to Seattle on Tuesday, August eleventh in the morning. Your full name?
U4:	*[John Doe]*
S5:	Hello, [John], I'm glad to hear from you again. You can now ask me for help.
S6:	OK. Let's see I have a United flight departing Pittsburgh at ten forty five a.m. arriving into Seattle at one fifty seven p.m. Is that OK?
U7:	*Yes*
S8:	Will you return to Pittsburgh from Seattle?
U9:	*no i will go to Los Angeles*
S10:	Traveling to Los Angeles. on what date
U11:	*Sorry can we go back to talk about the previous leg again*
S12:	For the first leg, you are on a United flight 1446 departing Pittsburgh at ten forty five A.M. tomorrow, arriving in Seattle at one fifty seven P.M. Do you want to take that?
U13:	*actually i want to leave on wednesday*
S14:	a flight on wednesday.
S15:	I have a flight departing Pittsburgh at ten forty five a.m., arrives Seattle at one fifty seven p.m. Is that OK?
U16:	*Yes*

Agent:	*So, Pablo, have you ever eaten consistently 2 or 3 pieces of fruit?*
User:	*When my wife was healthier, she used to take care of buying the fruit. Thus it was easier for me to eat it.*
Agent:	*And at any other time of your life?*
User:	*Long time ago, when I lived alone, I used to take care of buying the fruit myself, and I ate it more frequently.*
Agent:	*What does this information suggest to you about your objective?*
User:	*Well... That it's something that basically depends on me.*
Agent:	*So, can you see anything you could do to get closer to your objective?*
User:	*Uhmm... I should start thinking how I am going to organise to buy the fruit.*

System: Welcome to the Pittsburgh bus information line. You can say start over anytime. Where are you going to take your bus from?
User: Hi yeah I want to go from McKeesport
System: Sure. Where do you want to go?
User: Duquesne.
System: Did you mean Duquesne?
User: Yes
System: At what time do you want to take it? Say next bus or a time for example 8.40pm.
User: Around 4.15pm
System: I'm going to search for buses around 4.15pm.
Is that OK?
User: Yes.
System: On which day?
User: Today
System: I have found one bus. It is at 4.15pm. There is a bus number 61C from McKeesport. It will arrive in Duquesne at 4.21pm.
System: Is there anything else I can do for you?
User: No thanks
System: Thanks for calling. Good bye!

Let's Go

History: Applications

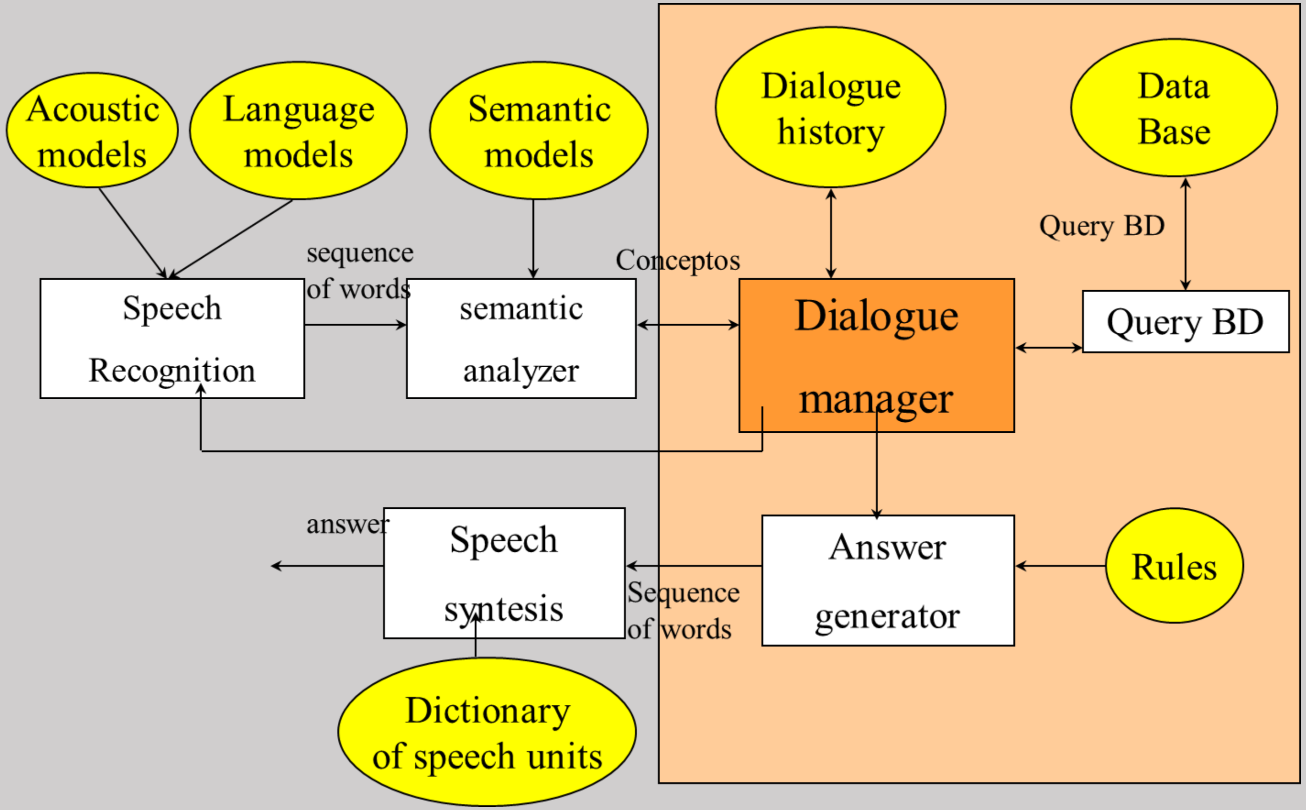
- Call routing
- Acces to an Information Service:
  - flights: Communicator
  - trains: SUNDIAL, ARISE, DIHANA
- Planning: TRAINS, TRIPS
- Restaurants: MIT
- Cambridge Restaurant
- Weather: JUPITER
- Bus service: Let’s Go
- Microsoft Dialogue challenge. A task-oriented dataset collected via Amazon Mechanical Turk
- DSTC series Multi-domain task-oriented dataset
- SNIPS-NLU Task-oriented dialogue dataset colleted in a crowdsourced fashion. It was used to train voice assistant agents
- Twitter: A social media dataset collected from Twitter
- REDDIT: A social media dataset collected from REDDIT

2. Task oriented Dialog Systems

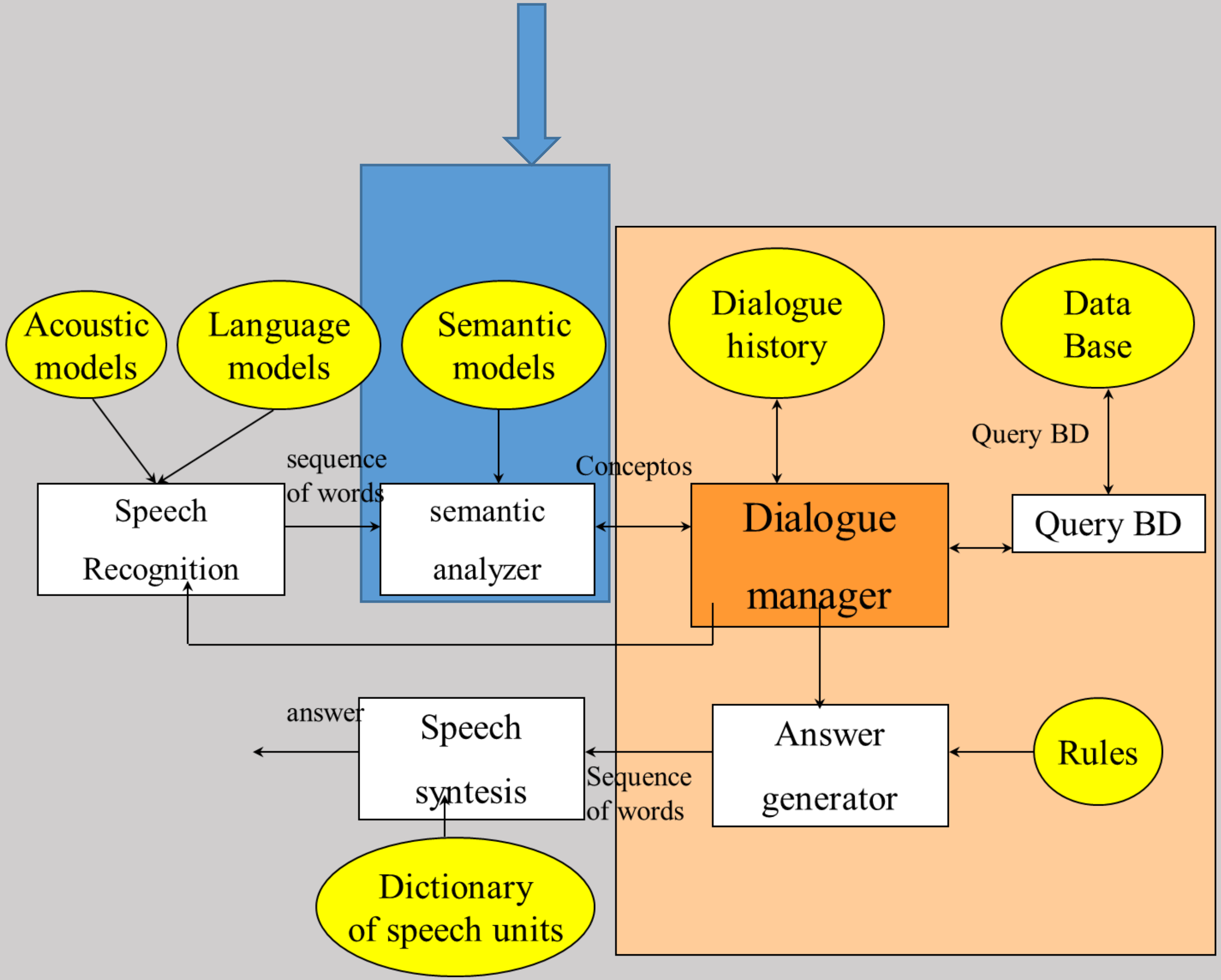
Task oriented Dialog Systems characteristics

- Semantic domain limited
- Size of vocabulary: medium
- Spontaneous speech.
- Mixed initiative.

Task oriented Dialog System architecture



Speech Understanding

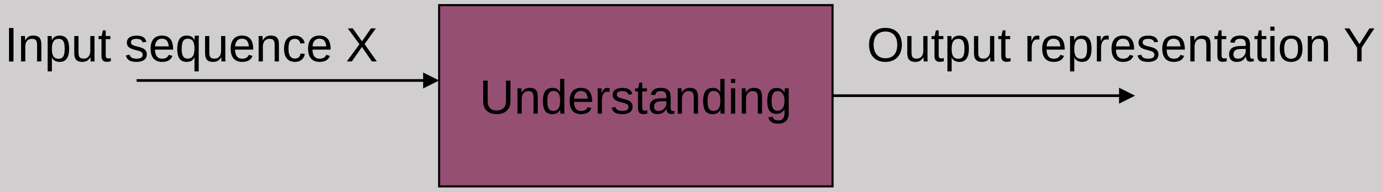


Speech Understanding

Understanding: Obtaining the meaning of a sequence of signs

Natural Language Understanding: Obtaining a conceptual representation of a sentence.

Spoken Language Understanding: Obtaining the meaning of a speech signal.



Spoken Language Understanding

Problems:

-Incorrect syntax of spoken sentences: hesitations, ungrammatical constructions, repetitions,..

-Errors in the Speech Recognition process

- The ASR don’t generate structure information (puntuations, sentence boundaries…)

-Ambiguous semantic sentences. Need of context for desmbiguation.

Types of SLU systems

- **Intent detection**
- **Slot filling**
- **Topic classification**

W	find	recent	comedies	by	james	cameron
	↓	↓	↓	↓	↓	↓
S	O	B-date	B-genre	O	B-dir	I-dir
D	movies					
I	find_movie					

Corpus and representation

ATIS

Over 25,000 utterances were collected (from AT&T, BBN,CMU, MIT, NIST, and SRI)

A: Context-independent queries

D: Context-dependent queries

X: Un-answerable queries

Example: flights from boston to philadelphia
FRAME: FLIGHT
DEPARTURE.CITY = boston
ARRIVAL.CITY = philadelphia

MEDIA

A 1250 French dialogue corpus: 250 speakers have followed each 5 hotel reservation scenarios.

Example of message with concept+value information.

The original French transcription is: "*oui l'hôtel dont le prix est*

inférieur à cinquante cinq euros"

n	W^c	c	**value**
1	yes	answer	yes
2	the	RefLink	singular
3	hotel	BDObject	hotel
4	which	null	
5	price	object	payment-amount
6	is below	comparative-payment	below
7	fifty five	payment-amount-int	55
8	euros	payment-currency	euro

DIHANA

- **DIHANA project**: Development of a dialog system to provide information in natural language about train services, schedules, and fares in Spanish.

Number of users	**225**
Number of dialogs/user	**4**
Number of user turns	**6280**
Average number of user turns/dialog	**7**
Average number of words/user turn	**7.74**
Vocabulary	**823**
Duration of the recording (hours)	**10.8**

DIHANA

Corpus labeling:

USER TURNS: Frame representation of the meaning of the utterance.

- Task-dependent concepts: *Hour, Price, Train-Type, Trip-Time,* and *Services*.
- Task-independent concepts: *Affirmation, Negation, and Not-Understood.*
- Attributes*: Origin, Destination, Departure-Date, Arrival-Date, Departure-Hour, Arrival-Hour, Class, Train-Type, Order-Number,* and *Services.*

Yes, I would like to know the timetables and the prices leaving from Valencia.
(Affirmation) **(Hour)** **Origin: Valencia** **(Price)** **Origin: Valencia**

SNIPS
It was collected by crowdsourcing through the SNIPS voice platform. Intents include requests to a digital assistant to complete various tasks, such as asking the weather, playing a song, book a restaurant, asking for a movie schedule, etc. SNIPS is now often used as a benchmark for NLU evaluations. It contains 15,884 utterances in 7 balanced intent classes. In training, there are 72 slot labels and a vocabulary size of 11,241 words

Seven intents:

•SearchCreativeWork (e.g. Find me the I, Robot television show),
•GetWeather (e.g. Is it windy in Boston, MA right now?),
•BookRestaurant (e.g. I want to book a highly rated restaurant in Paris tomorrow night),
•PlayMusic (e.g. Play the last track from Beyoncé off Spotify),
•AddToPlaylist (e.g. Add Diamonds to my roadtrip playlist),
•RateBook (e.g. Give 6 stars to Of Mice and Men),
•SearchScreeningEvent (e.g. Check the showtimes for Wonder Woman in Paris). The training set contains of 13,084 utterances, the validation set and the test set contain 700 utterances each, with 100 queries per intent.

Intent: setTemperature
Slots:
- - name: room
- entity: room
- - name: roomTemperature
- entity: snips/temperature

Utterances:
- Set the temperature to [roomTemperature](19 degrees) in the [room](bedroom)
- please set the [room](living room)'s temperature to [roomTemperature](twenty two degrees celsius)
- I want [roomTemperature](75 degrees fahrenheit) in the [room](bathroom) please
- Can you increase the temperature to [roomTemperature](22 degrees) ?

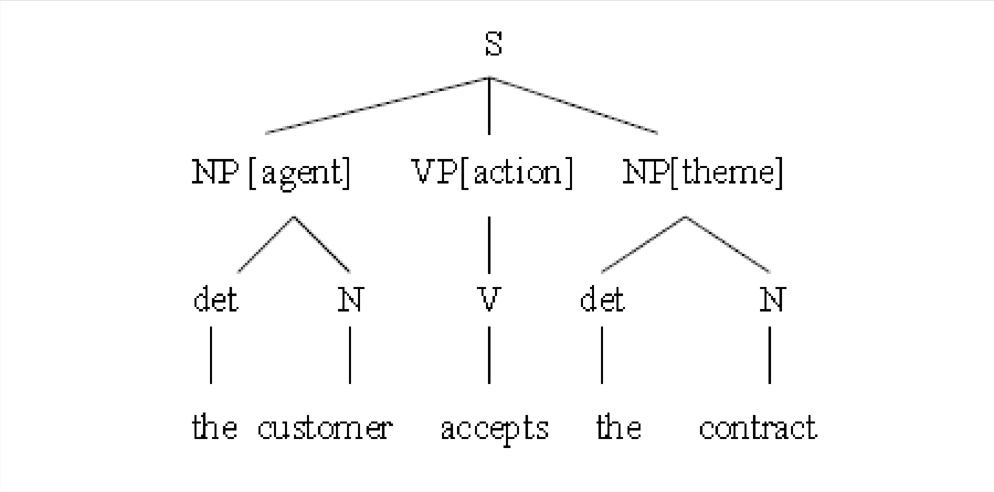
Intent: turnLightOn
Slots:
- name: room
  entity: room

Utterances:
- Turn on the lights in the [room](kitchen)
- give me some light in the [room](bathroom) please
- Can you light up the [room](living room) ?
- switch the [room](bedroom)'s lights on please

Semantic interpretation (Grammatical models)

Syntactic and semantic anaysis performed by a parser which produces a parse tree.

Example: The costumer accepts the contract



Internal nodes have associated semantic labels.

Context Free Grammar in which the Left Side of rules is a semantic category:

Example:

- LIST -> show me | I want | can I see|…
- DEPARTTIME -> (after|around|before) HOUR | morning | afternoon | evening
- HOUR -> one|two|three…|twelve (am|pm)
- FLIGHTS -> (a) flight|flights
- ORIGIN -> from CITY
- DESTINATION -> to CITY
- CITY -> Boston | San Francisco | Denver | Washington

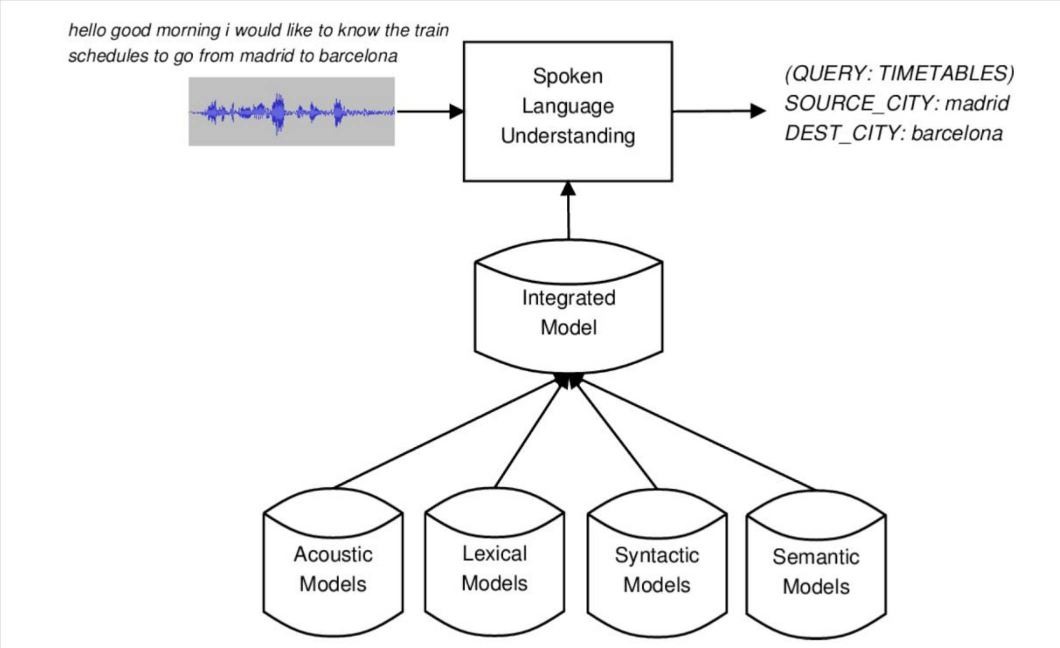
Problems with hand-crafted approaches

Hand-crafted grammars are

- not robust to spoken language input
- require linguistic and engineering expertise to develop if grammar is to have good coverage and optimised performance
- time consuming to develop
- error prone
- difficult to maintain

Semantic interpretation (Statistical models)

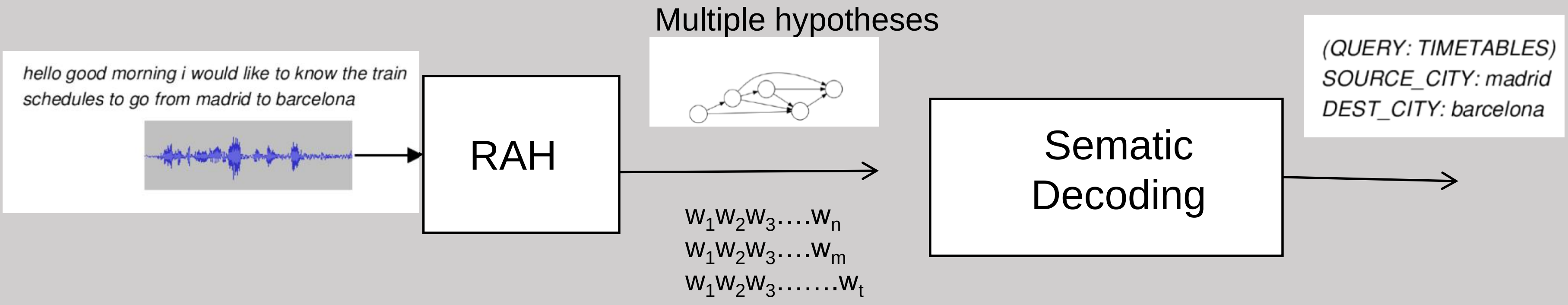
Integrated models



Problems: Excessively large search space

How to adequately weight the different models

Decoupled decoding



Problems: Two error sources connected in cascade
Errors generated in the ASR may imply a loss of information

Solutions: To transmite multiple hypotheses

Statistical modelling for SLU

Given word sequence W, find semantic representation of meaning C that has maximum *a posteriori* probability P(C|W)

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, P(C \mid W) = \underset{C}{\operatorname{argmax}} \frac{P(W \mid C) P(C)}{P(W)}$$

$$= \underset{C}{\operatorname{argmax}}\, P(W \mid C) P(C)$$

P(C): semantic model

P(W|C): Probability of word sequence W given C

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, P(W \mid C)P(C)$$

$$= \underset{C}{\operatorname{argmax}} \prod_{i=2}^{N} P(w_i \mid w_{i-1} \ldots w_1, C) P(w_1 \mid C) \prod_{i=2}^{M} P(c_i \mid c_{i-1} \ldots c_1)$$

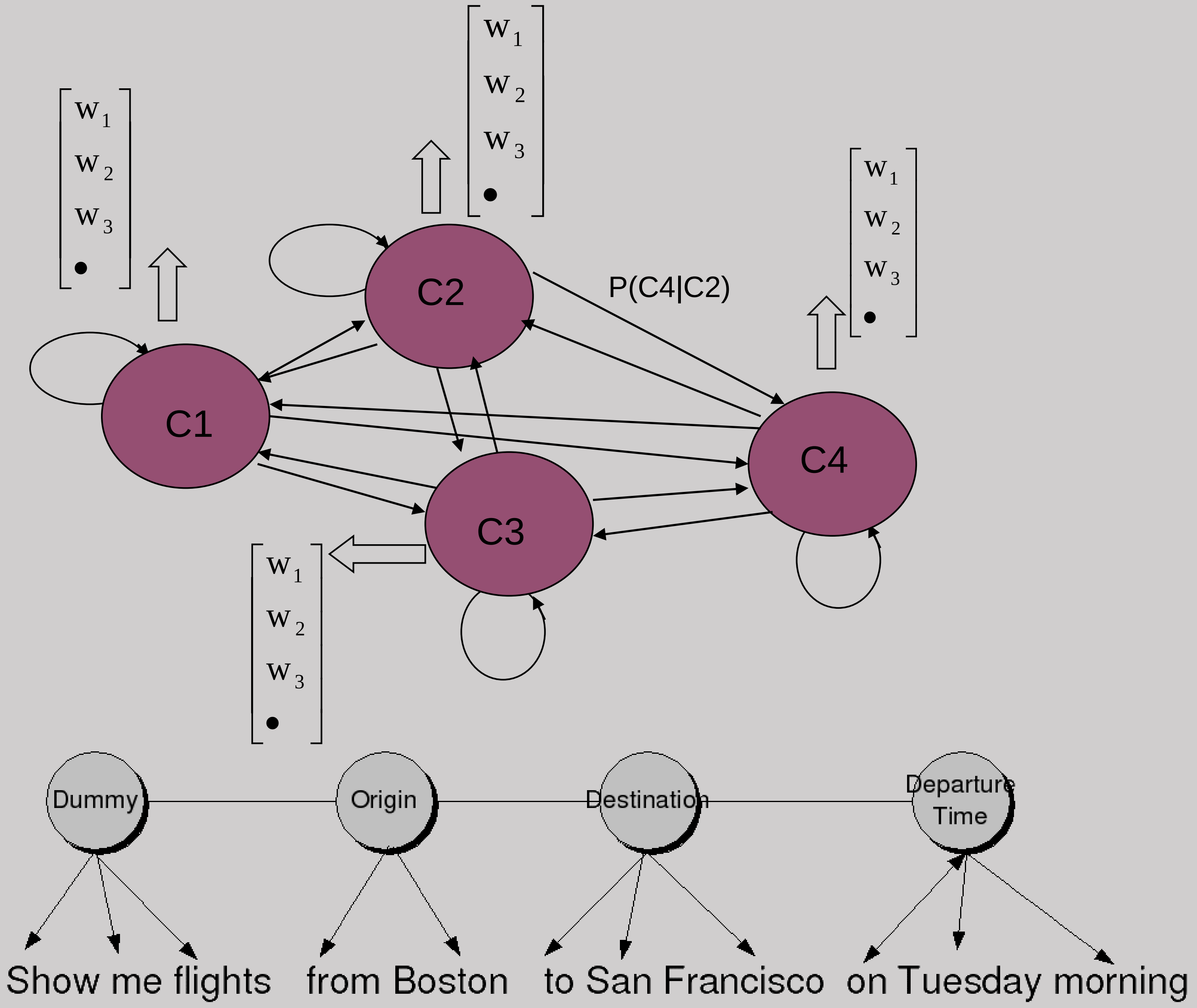
Assuming:

$$P(w_i \mid w_{i-1} \ldots w_1, C) = P(w_i \mid w_{i-1}, \ldots, w_{i-N+1}, c_i)$$

$$P(c_i \mid c_{i-1} \ldots c_1, C) = P(c_i \mid c_{i-1}, \ldots, c_{i-M+1})$$

$$\hat{C} = \underset{C}{\operatorname{argmax}} \prod_{i=2}^{N} P(w_i \mid w_{i-1}, \ldots, w_{i-N+1}, c_i) \prod_{i=2}^{M} P(c_i \mid c_{i-1} \ldots c_{i-M+1})$$

Hidden Markov Models



Learning: Pairs of sequences of words/sequendes of concepts. It is necessary a set of manually labeled sentences

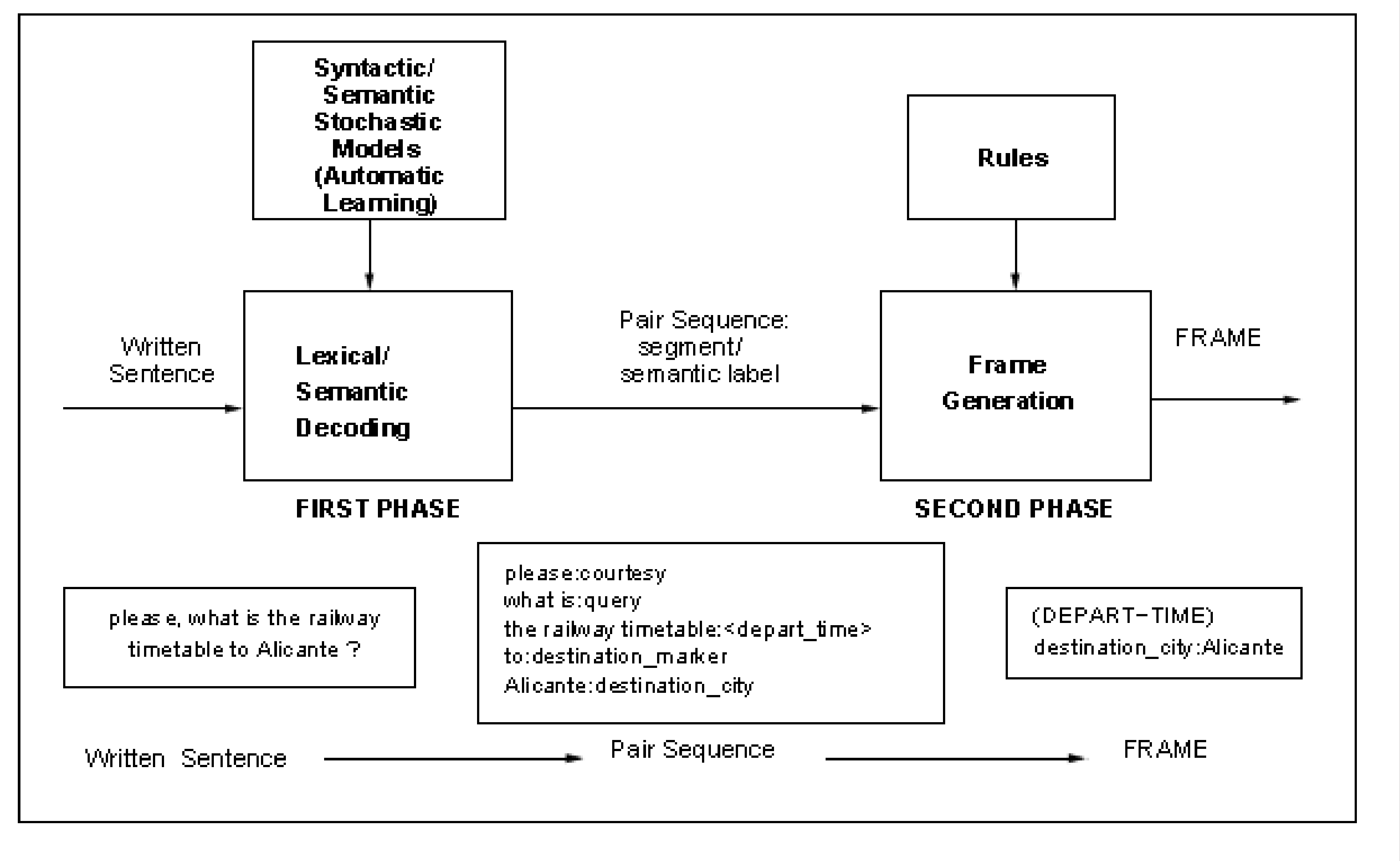
Recognizer: Viterbi algorithm

It can't modelize long term constraints nor other more complex constraints. Good results in a flat semantic representation.

Finite State Automata Models

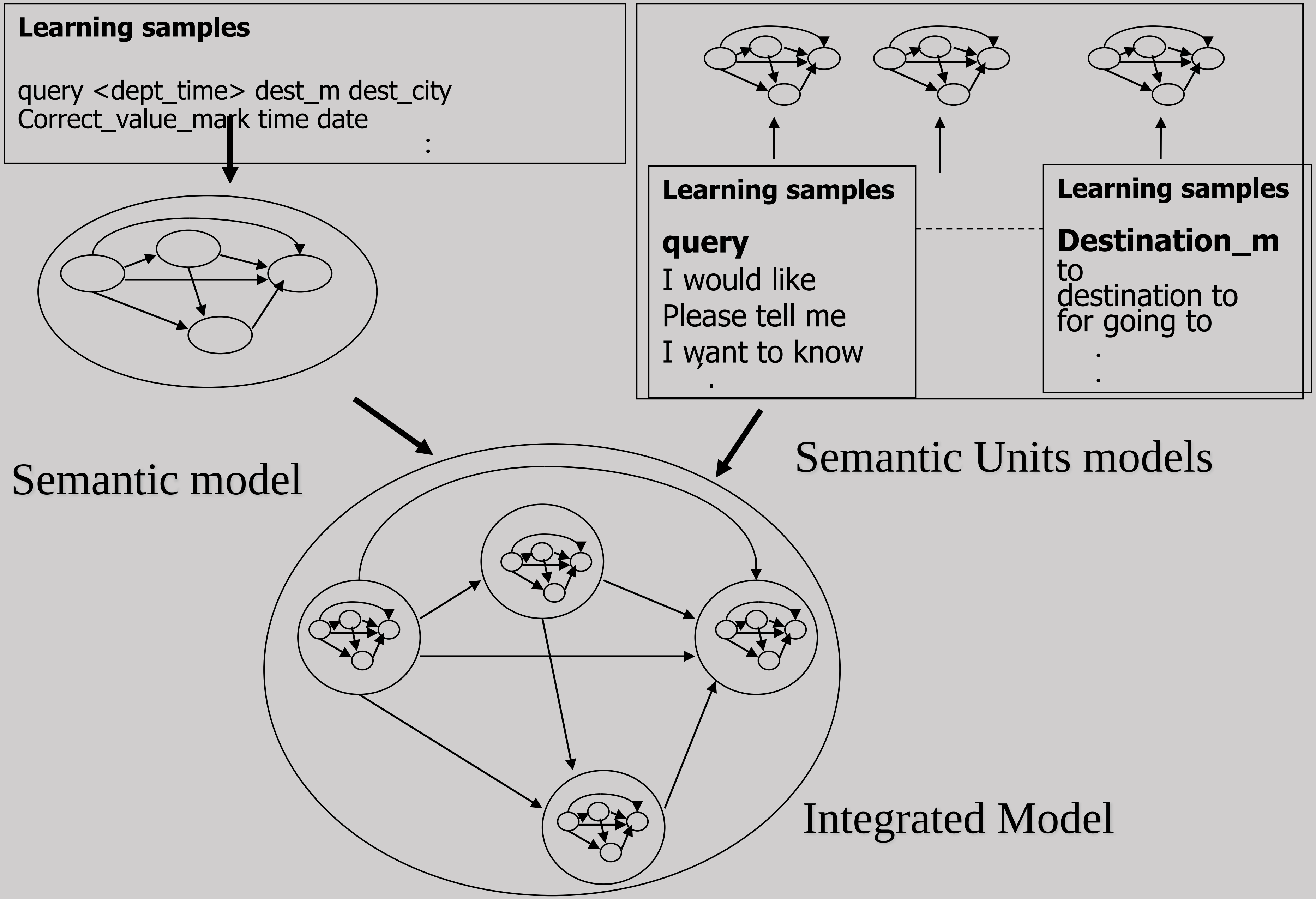
- The semantic representation chosen for the task is based on the concept of **frame**.
- **SU:** takes the sentence supplied by the recognition process as input and generates one or more frames.

Input sentence:
Sí, me gustaría saber el precio y tipo de tren que sale a las once.
Yes, I would like to know the price of the train that leaves at eleven and what type of train it is.
Semantic interpretation:
(ACCEPTANCE)
(FARE)
DEPARTURE-HOUR: 11.00
(TRAIN-TYPE)
DEPARTURE-HOUR: 11.00


Syntactic/ Semantic Stochastic Models (Automatic Learning)
Rules
Written Sentence
Lexical/ Semantic Decoding
Pair Sequence: segment/ semantic label
Frame Generation
FRAME
FIRST PHASE
SECOND PHASE
please, what is the railway timetable to Alicante ?
please:courtesy
what is:query
the railway timetable:<depart_time>
to:destination_marker
Alicante:destination_city
(DEPART-TIME)
destination_city:Alicante
Written Sentence
Pair Sequence
FRAME

- The understanding process is done in two phases:
  - First phase → translates the input sentence into a semantic sequence defined in an intermediate language (ISL).

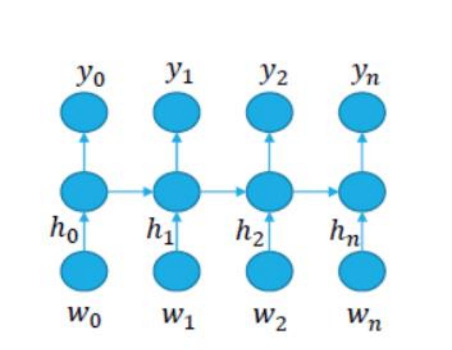
u_1: I would like	v_1: query
u_2: the train timetables	v_2: <departure_hour>
u_3: from	v_3: origin_mark
u_4: Valencia	v_4: origin_city
u_5: to	v_5: destination_mark
u_6: Barcelona	v_6: destination_city
Input pair (u,v) = $(u_1u_2u_3u_4u_5u_6, v_1v_2v_3v_4v_5v_6)$	
Output v = query <departure_hour> origin_mark origin_city destination_mark destination_city	


Learning samples
query <dept_time> dest_m dest_city
Correct_value_mark time date
:
Semantic model
Learning samples
query
I would like
Please tell me
I want to know
Learning samples
Destination_m
to
destination to
for going to
Semantic Units models
Integrated Model

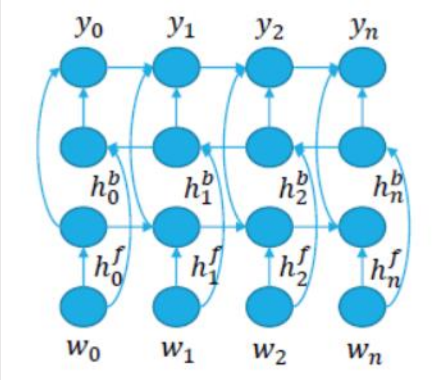
Recurrent Neural Networks (LSTM)

Word tags are not only determined by the associated terms, but also contexts. For example, the city name Boston could be tagged as originating or destination city, according to the lexical context it appears in.

For capturing such dependencies: two extensions to the RNN-LSTM architecture
- look-around LSTM
- bi-directional LSTM (bLSTM-LA)



LSTM



bLSTM-LA

Independent Models for Slot Filling and Intent Classification:

Independent models train each task separately and neural models typically use RNN for SF and IC. At each time step t, the encoder transforms the word representation xt to the hidden state ht. For SF, the output layer predicts the slot label yt condition on ht. For IC, typically the last hidden state hT is used to predict the intent label of the utterance x.

Note that, for independent approaches, the models for SF and IC are trained separately.

Joint models for SF and IC

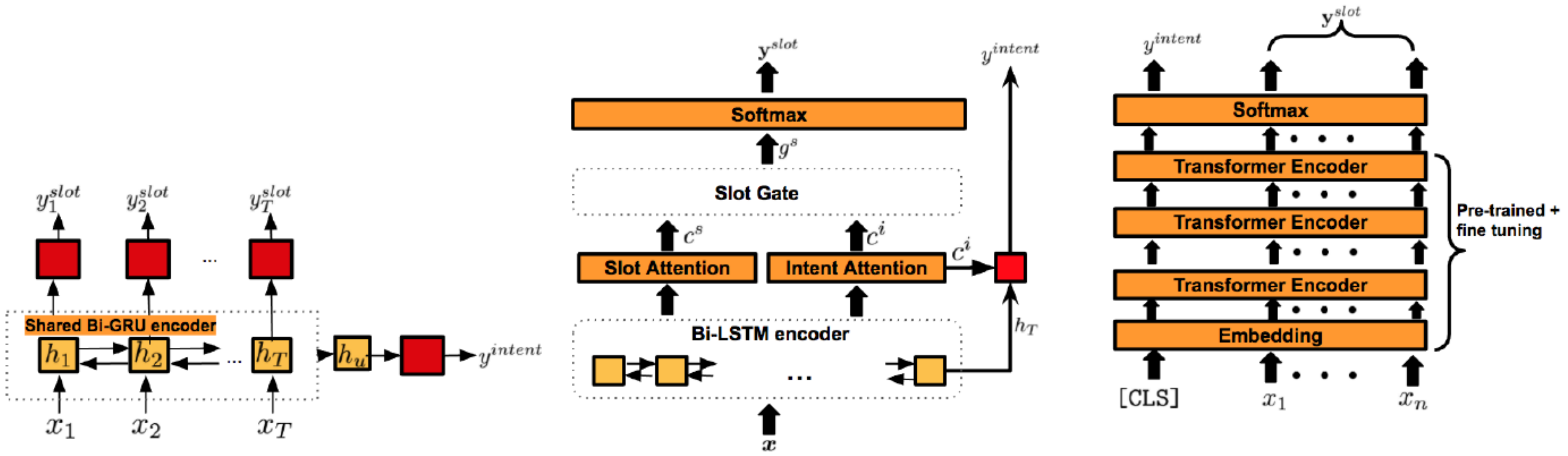


Figure 1: *Left:* Shared Bi-GRU encoder (Zhang and Wang, 2016). *Middle:* Slot-Gate Mechanism (Goo et al., 2018). *Right:* BERT Based (Chen et al., 2019).

Evaluation

- Concept error rate

  - Sequence of concepts.

  - Frame slots (attribute/value pairs)

$$\mathrm{CER} = \frac{\mathrm{Sub} + \mathrm{Ins} + \mathrm{Del}}{|\mathrm{C}_{\mathrm{ref}}|}$$

- Precision

$$P = \frac{Ncorrect}{Nobtained}$$

- Coverage

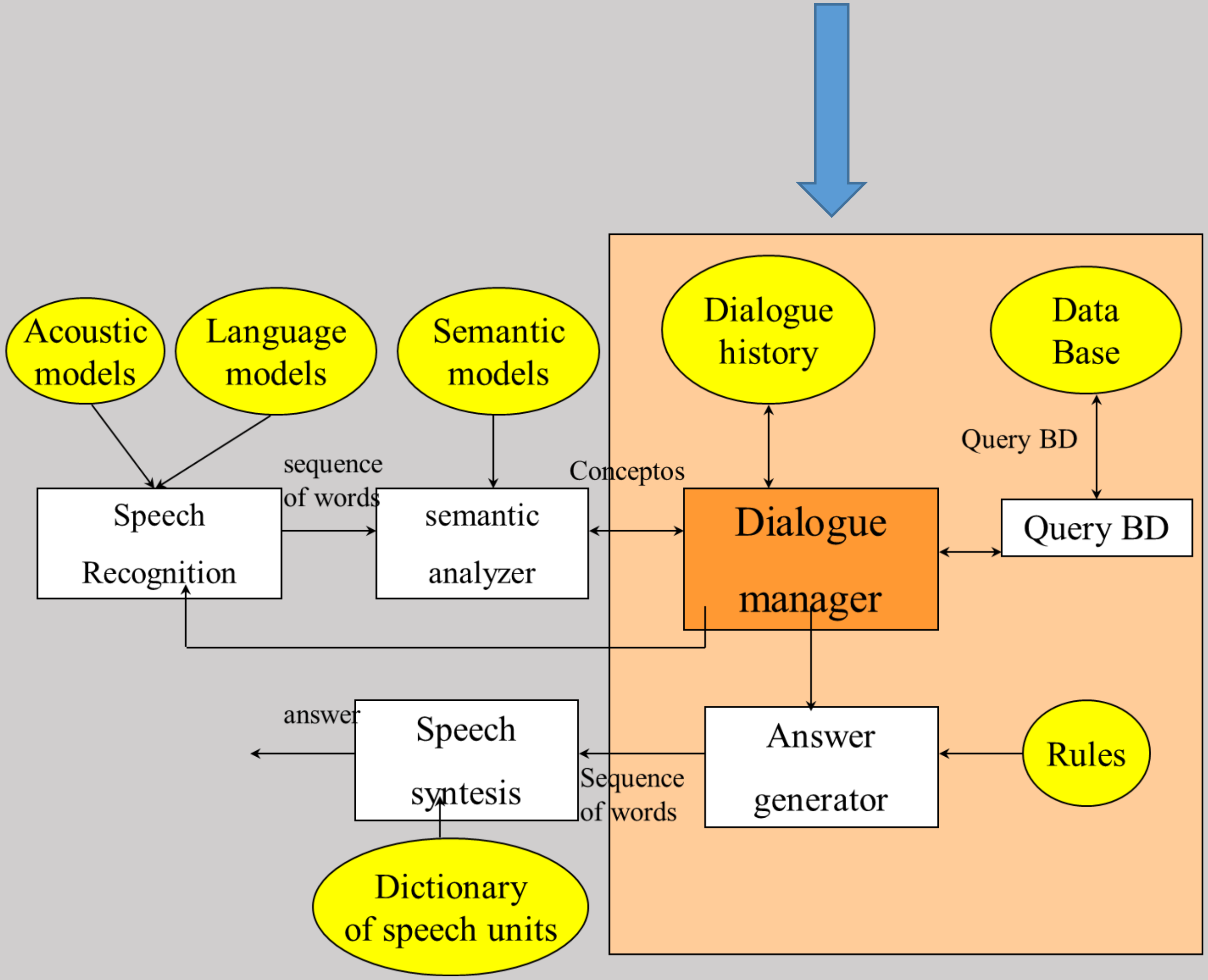
$$C = \frac{Ncorrect}{Nref}$$

Dialog Manager modelization for Task oriented Dialog Systems

Rule- based: Easy to implement. Popularity in earlier industry products. The dialogue flows of these systems are predetermined, which keeps the applications of the dialogue systems within certain scenarios.

Non-neural machine learning based systems: These systems are more flexible compared with rule-based systems because the dialogue flows are not predetermined.

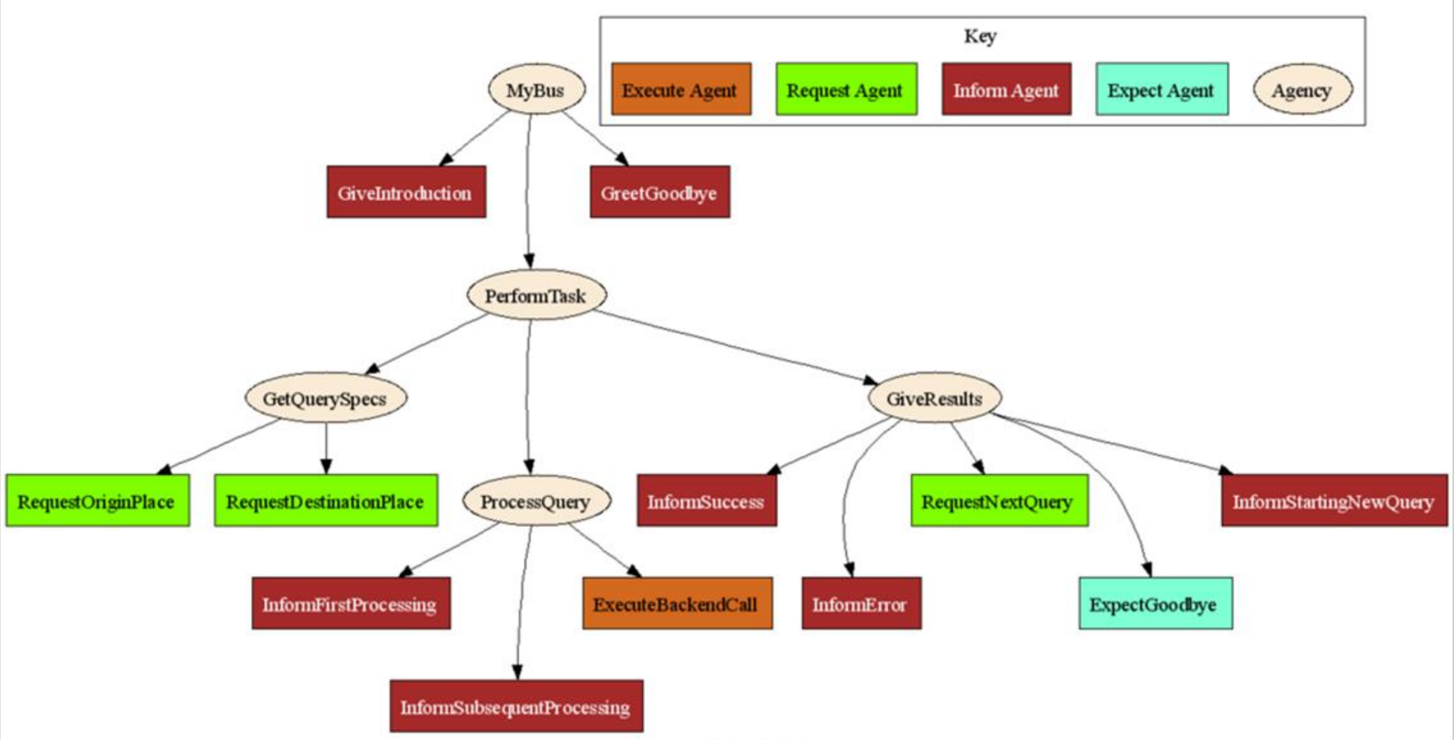
Dialog Manager



RavenClaw

The dialog task specification describes a hierarchical plan for the interaction. More specifically, a dialog task specification consists of a tree of dialog agents, where each agent is responsible for handling a subpart of the interaction

Let'sGo system



MyBus Task Tree

Dialog strategy: Main actions of a Dialog Manager.

- To ask for more information
- To give the answer supplied by the Data Base.
- To start a process of error recovery (or hypothesis confirmation).
- To detect out of domain questions.

Dialog startegy

What can do the system in each turn?

- To verify the consistency of the values (Example: 29th)
- To ask for more information
- To confirm values with low confidence score.

After de DB Query:

- To relax restrictions (if there are many results)
- To ask for more restrictions, or range of values (if there are not results).

How to model the **discurse structure**?

A first level of analysis is based on the identification of **dialogue acts** (DAs). A DA represents the meaning of an utterance at the level of illocutionary forcé.

Types of Dialog Acts:

Constatives: committing the speaker to something's being the case (answering, claiming, confirming, denying, disagreeing, stating)

Directives: attempts by the speaker to get the addressee to do something (advising, asking, forbidding, inviting, ordering, requesting)

Commissives: committing the speaker to some future course of action (promising, planning, vowing, betting, opposing)

Acknowledgments: express the speaker's attitude regarding the hearer with respect to some social action (apologizing, greeting, thanking, accepting an acknowledgment)

Tag	Example
THANK	*Thanks*
GREET	*Hello Dan*
INTRODUCE	*It's me again*
BYE	*Alright bye*
REQUEST-COMMENT	*How does that look?*
SUGGEST	*from thirteenth through seventeenth June*
REJECT	*No Friday I'm booked all day*
ACCEPT	*Saturday sounds fine*
REQUEST-SUGGEST	*What is a good day of the week for you?*
INIT	*I wanted to make an appointment with you*
GIVE_REASON	*Because I have meetings all afternoon*
FEEDBACK	*Okay*
DELIBERATE	*Let me check my calendar here*
CONFIRM	*Okay, that would be wonderful*
CLARIFY	*Okay, do you mean Tuesday the 23rd?*
DIGRESS	*[we could meet for lunch] and eat lots of ice cream*
MOTIVATE	*We should go to visit our subsidiary in Munich*
GARBAGE	*Oops, I-*

	Utterance	Dialogue act
U:	Hi, I am looking for somewhere to eat.	`hello(task = find,type=restaurant)`
S:	You are looking for a restaurant. What type of food do you like?	`confreq(type = restaurant, food)`
U:	I'd like an Italian somewhere near the museum.	`inform(food = Italian, near=museum)`
S:	Roma is a nice Italian restaurant near the museum.	`inform(name = "Roma", type = restaurant, food = Italian, near = museum)`
U:	Is it reasonably priced?	`confirm(pricerange = moderate)`
S:	Yes, Roma is in the moderate price range.	`affirm(name = "Roma", pricerange = moderate)`
U:	What is the phone number?	`request(phone)`
S:	The number of Roma is 385456.	`inform(name = "Roma", phone = "385456")`
U:	Ok, thank you goodbye.	bye()

Switchboard

A person-person corpus annotated.

- In order to tag the 1155 SWBD conversations. SWBD-DAMSL tagset consists of approximately 60 which the labelers combined to produce 220 unique tags for the 205,000 SWBD utterances. The SWBD conversations had already been handsegmented into utterances by the Linguistic Data Consortium.

- An utterance roughly corresponds to a sentence. Each utterance thus received exactly one of these 220 tags.

- They then clustered these 220 tags into 42 final tags.

Most common tags in Switchboard

Tag	Example	Count	%
Statement	*Me, I'm in the legal department.*	72,824	36%
Backchannel	*Uh-huh.*	37,096	19%
Opinion	*I think it's great*	25,197	13%
Agree/Accept	*That's exactly it.*	10,820	5%
Abandoned/Turn-Exit	*So, -/*	10,569	5%
Appreciation	*I can imagine.*	4,633	2%
Yes-No-Question	*Do you have to have any special training*	4,624	2%
Non-verbal	*<Laughter>,<Throat_clearing>*	3,548	2%
Yes answers	*Yes.*	2,934	1%
Conventional-closing	*Well, it's been nice talking to you.*	2,486	1%
Uninterpretable	*But, uh, yeah*	2,158	1%
Wh-Question	*Well, how old are you?*	1,911	1%
No answers	*No.*	1,340	1%
Response Ack	*Oh, okay.*	1,277	1%
Hedge	*I don't know if I'm making any sense or not.*	1,182	1%
Declarative Question	*So you can afford to get a house?*	1,174	1%
Other	*Well give me a break, you know.*	1,074	1%
Backchannel-Question	*Is that right?*	1,019	1%

Example of Dialog acts in a Task-oriented Dialog System

Do you want to know the timetables?

(Confirmation:Departure-Hour:Nil)

Tell me the departure date
(Question:Departure-Date:Nil)

Yes, I would like to know the timetables and the prices leaving from Valencia.

(Affirmation)
(Question:Departure-hour, Origin: Valencia)
(Question:Price, Origin: Valencia)

Dialog History management and Dialog Strategy

Dialogue state tracking The Dialogue State Tracker infers the current ***belief state*** of the conversation, given the dialogue history up to the current point *t.* The current belief state encodes the user's goal and the relevant dialogue history, i.e. it is an internal representation of the state of the conversation. It is important to take the previous belief states into account in order to handle misunderstandings. The main challenge for the DST module is to handle the uncertainty, which stems from the errors made by the ASR module and the NLU unit. Typically, the output of the DST unit is represented as a probability distribution over multiple possible dialogue states $b(s)$, which provides a representation of the uncertainty.

Strategy The strategy is learned by the dialogue manager. The input is the current belief state computed by the DST module. The DM generates the next action of the system, which is represented as a dialogue act. In other words, based on the current turn values and on the value history the system performs an action (e.g. retrieve data from a database, ask for a missing information, etc.).

Modelization of Dialog Manager

Initially, dialogue managers were implemented using **rule-based** approaches.

When data had become available in sufficient amount, **data-driven methods** were proposed for learning dialogue strategies from data.

The dialogue manager can be modeled as:

- Finite state automata

- Markov decision problem

- Partially observable MDPs

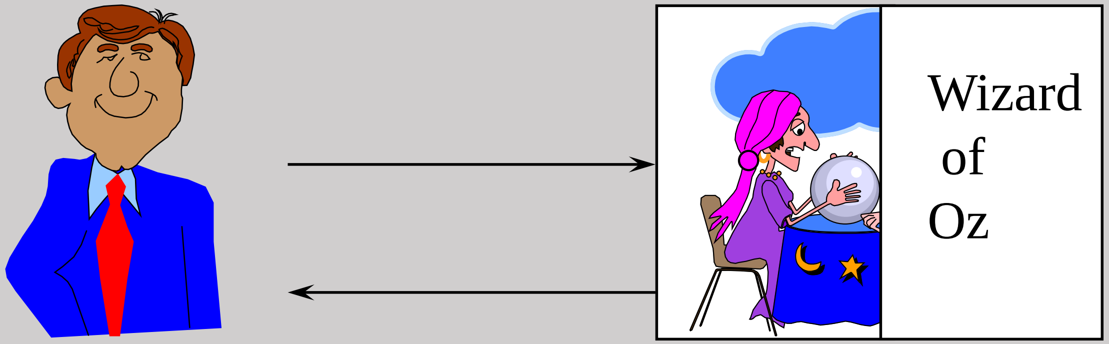
- Neural Network based approaches

How to obtain learning data: **Wizar of Oz**

Wizard of Oz

Learning: Corpus acquisition

Wizard of Oz.



Different behavior of the User when he is talking to a machine or a human.

Stochastic dialog manager

Dialogue Policy:

The goal of the dialog policy is to decide what action the system should take next, that is, what dialog act to generate.

A simple policy based on local context:
The goal of the dialog policy at turn i in the conversation is to predict which action Ai to take, based on the entire dialog state. The state could mean the entire sequence of dialog acts from the system (A) and from the user (U), in which case the task would be to compute:

$$\hat{A}_i = \underset{A_i \in A}{\operatorname{argmax}} P(A_i | (A_1, U_1, \ldots, A_{i-1}, U_{i-1})$$

We can simplify this by maintaining as the dialog state the set of slot-fillers that the user has expressed, collapsing across the many different conversational paths that could lead to the same set of filled slots. Such a policy might then just condition on the current state of the frame Frame (which slots are filled and with what) and the last turn by the system and user:

$$\hat{A}_i = \underset{A_i \in A}{\operatorname{argmax}} P(A_i | \text{Frame}_{i-1}, A_{i-1}, U_{i-1})$$

These probabilities can be estimated by a classifier.

User turns labeling

- The representation of user and system turns is done in terms of dialog acts:
  - **USER TURNS**: Frame representation of the meaning of the utterance.
    - Task-dependent concepts: *Hour, Price, Train-Type, Trip-Time,* and *Services*.
    - Task-independent concepts: *Affirmation, Negation, and Not-Understood.*
    - Attributes*: Origin, Destination, Departure-Date, Arrival-Date, Departure-Hour, Arrival-Hour, Class, Train-Type, Order-Number,* and *Services.*

Yes, I would like to know the timetables and the prices leaving from Valencia.

(Affirmation)
(Hour)
Origin: Valencia
(Price)
Origin: Valencia

System turns labeling

- **SYSTEM TURNS**: Three levels of labeling were defined:
  - First level: General acts of any dialog.
  - Second level: Concepts involved in the turn.
  - Third level: Values of the attributes given in the turn.
  - **1st level:** *Opening, Closing, Undefined, Not-Understood, Waiting, New-Query, Acceptance, Rejection, Question, Confirmation,* and *Answer.*
  - **2nd and 3rd level:** *Departure-Hour, Arrival-Hour, Price, Train-Type, Origin, Destination, Date, Order-Number, Number-Trains, Services, Class, Trip-Type, Trip-Time,* and *Nil.*

Do you want timetables to Barcelona, from Valencia?

(Confirmation:Departure-Hour:Destination)(Confirmation:Origin:Origin)

The stochastic Dialog Manager

- A Dialog Manager (DM) can be based on the stochastic modelization of the sequences of dialog acts.
- It can generate system turns based only on the information supplied by the user turns and the information contained in the model.
- A labeled corpus of dialogs can be used to estimate the stochastic DM.
- Formal description:
  - A_i: the system answer at time *i*.
  - U_i: semantic representation of the user turn at time *i*.

Dialog → $\underbrace{(A_1, U_1), \cdots, (A_i, U_i)}_{S_i}, \cdots, (A_n, U_n)$

S_i: State of the dialog sequence at time *i*

The stochastic Dialog Manager

- Formal description:
  - At time *i*, the objective of the dialog manager is to find the best system answer A_i
  - The selection is made by maximizing:

$$\hat{A}_i = \underset{\underbrace{A_i \in \mathcal{A}}_{\text{All the possible system answers.}}}{\operatorname{argmax}} P(A_i | S_1, \cdots, S_{i-1})$$

  - We establish a partition in the space of sequences of states:
    - DR_i: Dialog register at time *i* (concepts and attributes).

The stochastic Dialog Manager

- <u>Formal description</u>:
  - For a sequence of states of a dialog, there is a corresponding sequence of *DR*:

$$\begin{array}{ccccccccc} & S_1, & & \cdots, & & S_i, & \cdots, & & S_n \\ \uparrow & & \uparrow & & \uparrow & & & \uparrow & \\ \underbrace{DR_1} & & DR_2 & & DR_i & & & DR_n & \end{array}$$

Default information of the dialog manager: *Origin* and *Class*.

  - **<u>Two different sequences of states are considered to be equivalent if they lead to the same DR_i</u>**
    - Great reduction in the number of different histories in the dialogs.
    - A loss in the chronological information.

The stochastic Dialog Manager

- Formal description:
  - The selection of the best A_i is given by:

$$\hat{A}_i = \underset{A_i \in \mathcal{A}}{\operatorname{argmax}}\, P(A_i | DR_{i-1}, S_{i-1})$$

  - Each user turn:
    - supplies the system with information about the task;
    - provides other kinds of information, such as task-independent information (*Affirmation*, *Negation*, and *Not-Understood* dialog acts).
  - The probabilities of the proposed model are obtained from a labeled training corpus through a maximum likelihood estimation.

The stochastic Dialog Manager

- Dialog Register representation:
  - The DR is a sequence of 15 fields:
    - Five concepts: *Hour*, *Price*, *Train-Type*, *Trip-Time*, and *Services*.
    - Ten attributes: *Origin, Destination, Departure-Date, Arrival-Date, Departure-Hour, Arrival-Hour, Class, Train-Type, Order-Number,* and *Services.*
  - We have assumed that the exact values of the attributes are not significant to determine the next system answer:
    - **0**: The concept is not activated, or the value of the attribute is not given.
    - **1**: The concept or attribute is activated with a confidence score that is higher than a given threshold.
    - **2**:The concept or attribute is activated with a confidence score that is lower than the given threshold.
  - ***DR*** = 15 length string of elements from {0,1,2}.

The stochastic Dialog Manager

- Example of a dialog:

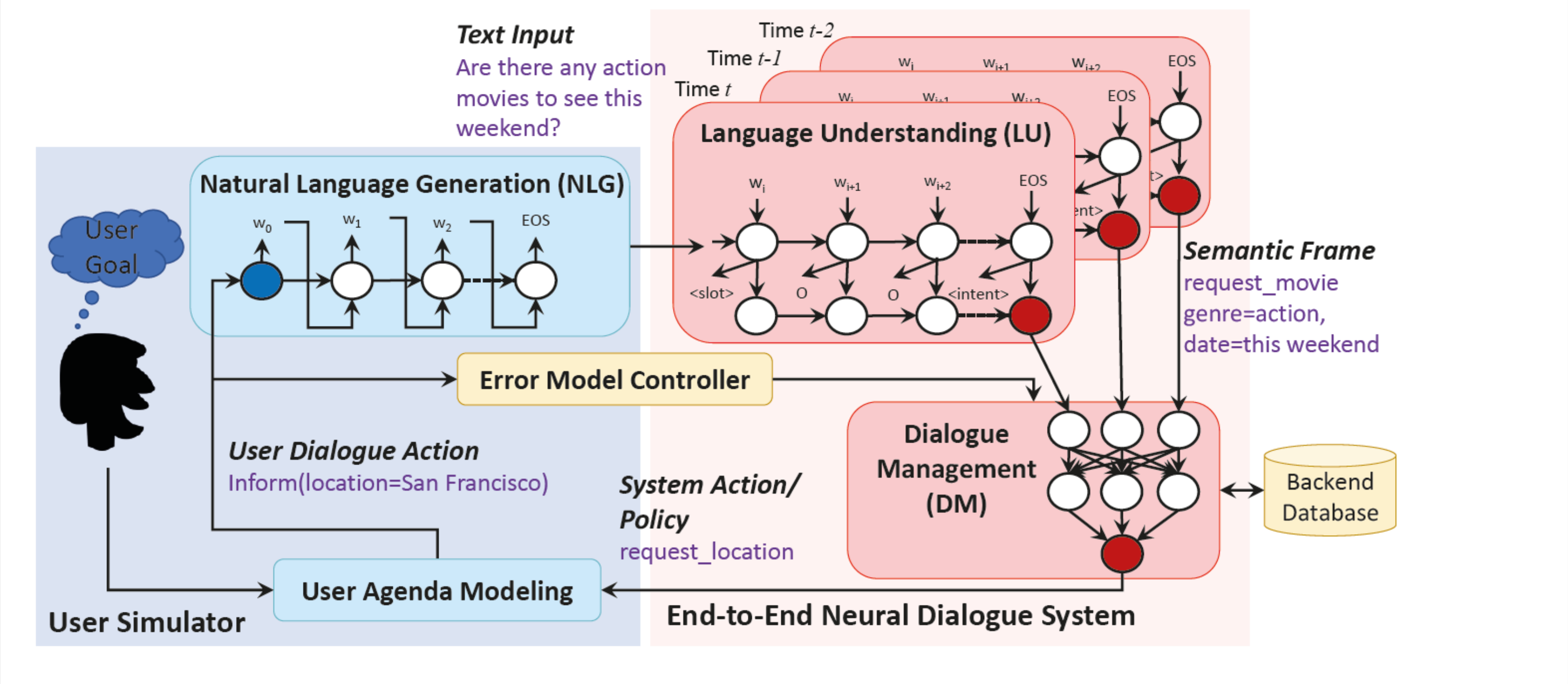
System$_1$: Welcome to the railway information system. How can I help you? **A_1: (Opening:Nil:Nil) DR: 00000-1000001000**
User$_1$: I want to go to Barcelona. **U_1=()**
DR: 00000-1100001000 + Opening:Nil + U_1→ A_2 = (Confirmation:Departure-Hour:Nil) **System$_2$: Do you want to know the timetables?**
User$_2$: Yes, for the Euromed train. U_2=(Affirmation)
DR: 10000-1100001100 + Confirmation:Departure-Hour + U_2 → **A_3: (Question:Departure-Date:Nil)** **System$_3$: Tell me the departure date.**

The stochastic Dialog Manager

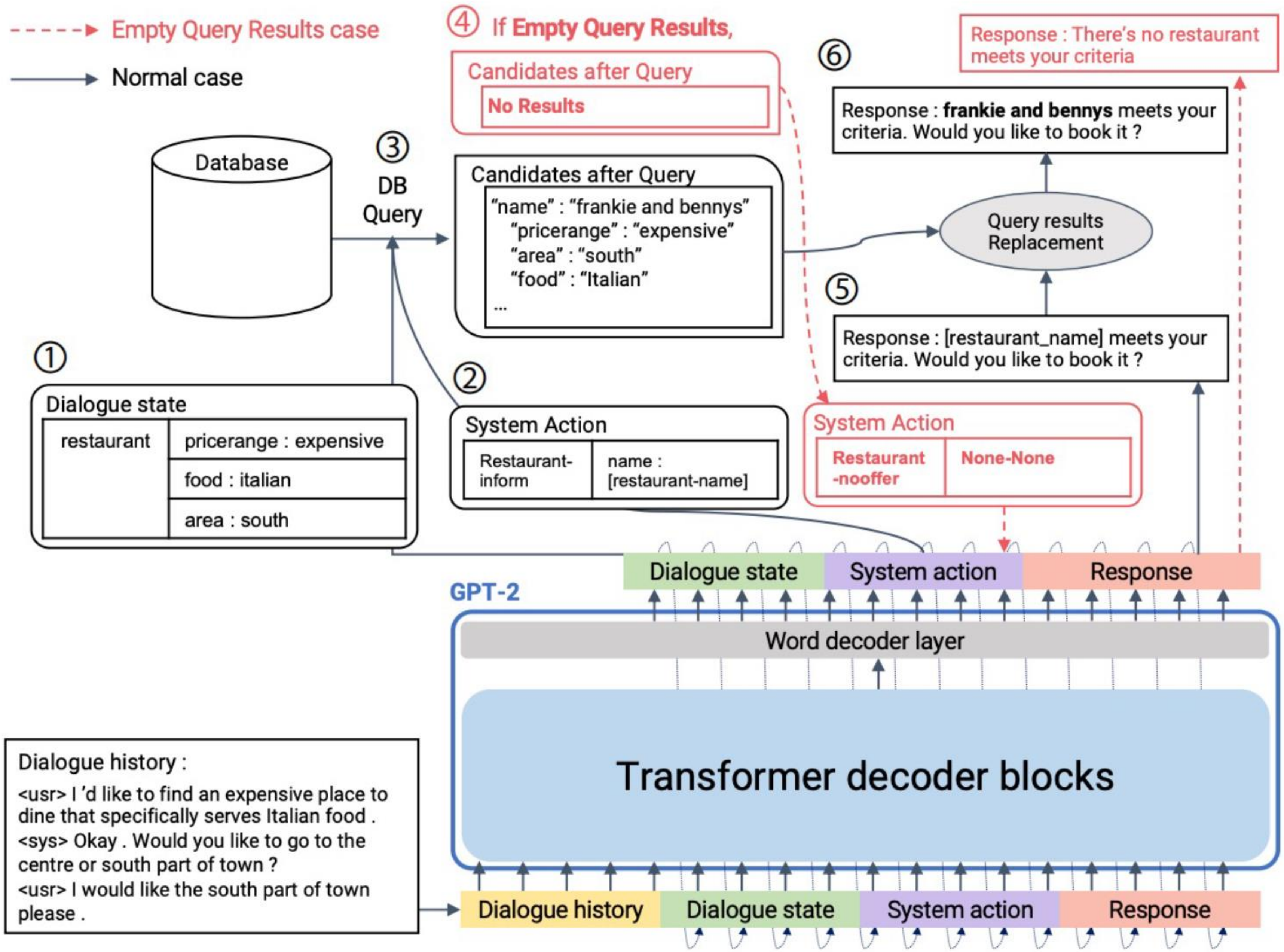
- Example of a dialog:

User$_3$: Tomorrow U_3=()
DR: 10000-1120001100 + Question:Departure-Date + U_3 → **A_4 = (Confirmation:Departure-Date:Departure-Date)** **System$_4$: Do you want to leave tomorrow?**
User$_4$: Yes U_4=(Affirmation)
DR: 10000-1110001100 + Confirmation:Departure-Date + U_4 → **A_5 = (Answer:Departure-Hour:Number-Trains,Train-Type,Departure-Hour)** **(New-Query:Nil:Nil)** **System$_5$: There are several Euromed trains. The first one leaves at 08:54 and the last one at 23:45. Anything else?**
User$_5$: No, thank you **U_5=(Negation)**
DR: 10000-1110001100 + Answer-New-Query:Departure-Nil + U_5 → **A_6 = (Closing:Nil:Nil)** **System6: Thanks for using this service. Have a good trip.**

Neural-network based dialog systems



Li et al. 2017



Empty Query Results case
Normal case
④ If Empty Query Results,
Candidates after Query
No Results
⑥
Response : There's no restaurant meets your criteria
Response : frankie and bennys meets your criteria. Would you like to book it ?
Database
③
DB Query
Candidates after Query
"name" : "frankie and bennys"
"pricerange" : "expensive"
"area" : "south"
"food" : "Italian"
...
Query results Replacement
⑤
Response : [restaurant_name] meets your criteria. Would you like to book it ?
①
Dialogue state
restaurant
pricerange : expensive
food : italian
area : south
②
System Action
Restaurant-inform
name : [restaurant-name]
System Action
Restaurant -nooffer
None-None
Dialogue state
System action
Response
GPT-2
Word decoder layer
Transformer decoder blocks
Dialogue history :
<usr> I 'd like to find an expensive place to dine that specifically serves Italian food .
<sys> Okay . Would you like to go to the centre or south part of town ?
<usr> I would like the south part of town please .
Dialogue history
Dialogue state
System action
Response

3. Open-domain dialogue systems

Three categories: Rule-based, Retrieval-based, Generative systems

Rule-based systems

The Rule-based model selects the response from a set of rules without generating new text responses. The Dialog Management component passes the placeholder values that may be needed to fill the template for the response to the Response Generation Module. Rule-based models use a Knowledge Base organized with conversational patterns.

.

Corpus-based systems

Retrieval-based systems: the dialogue is modelled as an information retrieval task. The retrieval-based model is more flexible as it selects the most suitable response with the check and analysis of available resources. A set of candidate utterances is ranked by relevance. The idea is to retrieve the most relevant answer to a given utterance, thus learning to map multiple semantically equivalent user-utterances to an appropriate answer.

As retrieval systems do not have a generative component and their outputs originate from human conversations, they are generally fluent and understandable. They are also relatively safe, as many types of harmful responses can be filtered. However, retrieval systems are limited in their ability to converse about topics not covered in the provided responses.

Corpus-based systems

Generative systems: sequence-to-sequence models to map the user message and dialogue history into a response sequence that may not appear in the training corpus.
The Generative model uses Natural Language Generation (NLG) to respond in a humanlike natural language based on the last and previous inputs. However, developing and training such a model is challenging because it needs an extensive set of data for training to establish a fruitful conversation. When the training corpus is small, grammatical errors are made, particularly in long sentences.
The dialogue systems are based on deep neural networks. Usually, the dialogue structure is learned from a large corpus of dialogues. Thus, the corpus defines the dialogue behaviour of the conversational agent.
The architectures are inspired by the machine translation literature, especially neural machine translation. Neural machine translation models are based on the Sequence to Sequence (seq2seq) architecture which is composed of an encoder and a decoder. They are usually based on a Recurrent Neural Network (RNN). The encoder maps the input into a latent representation on which the decoder is conditioned. Their systems are LSTM-based seq2seq models Transformer-based models include generative variants of the transformer memory network.

Several algorithms to train generative systems have been proposed.
Given a training set $D = \{(R_1, C_1, B_1), \ldots, (R_N, C_N, B_N)\}$ with N examples consisting of context C_i, background information B_i, and response R_i, models are most commonly trained using maximum likelihood estimation

Generative systems are not limited to a predefined set of utterances. Their responses are not guaranteed to be well-formed. However, in contrast to retrieval systems, they are not restricted to talking about topics within a predefined set of responses

4. Evaluation of dialog systems

Objetive: dialog completion, duration, distance to ground-truth,…

Subjetive: human evaluation

First attempt: PARADISE

Automatic Evaluation metrics

- Task completion: adequate for task oriented dialog systems
- Efficiency: number of turns, number of error recovery turns.
- State tracker: comparison with ground-truth.
    - Current state description (belief)
    - Answer generation:
        Comparison with ground-truth (Precision, Recall, F-1)
            1) Word-overlap metrics: BLEU, METEOR, ROUGE
            2) Word embedding-based metrics:

Human evaluation

Lab-experiments: Users were invited to participate in the lab where they interacted with the dialogue system and subsequently filled a questionnaire (very controlled, not comparable to real world)

In-field experiments: collecting feedback from real users of the dialogue systems.

Crowdsourcing: using crowdsourcing platforms such as Amazon Mechanical Turk (high variability of user behavior). The task need to be prepared and the users need to be properly instructed, Benchmarks: DSTC 1-8 Dialog System Technology Challenges basically tracking state.

- Twitter Corpus, 850k Twitter dialogues
- Movie Dialog Dataset, 1 million Reddit dialogues

4. Question Answering systems

- Single-turn QA

- Conversational QA

Single-turn QA can be approached from two main perspectives:

• Open QA, where systems collect evidences and answers across several sources such as Web pages and knowledge bases

• Reading Comprehension (RC), where the
RC systems can be oriented to:
• Extractive RC, where systems extract spans of text with the answer. Datasets contains thousands of examples, which permits to train Deep Learning systems and obtain good results.
• Generative QA, where systems create a text that answers the question. The exact text is not necessarily contained in any document, which makes this a challenging task

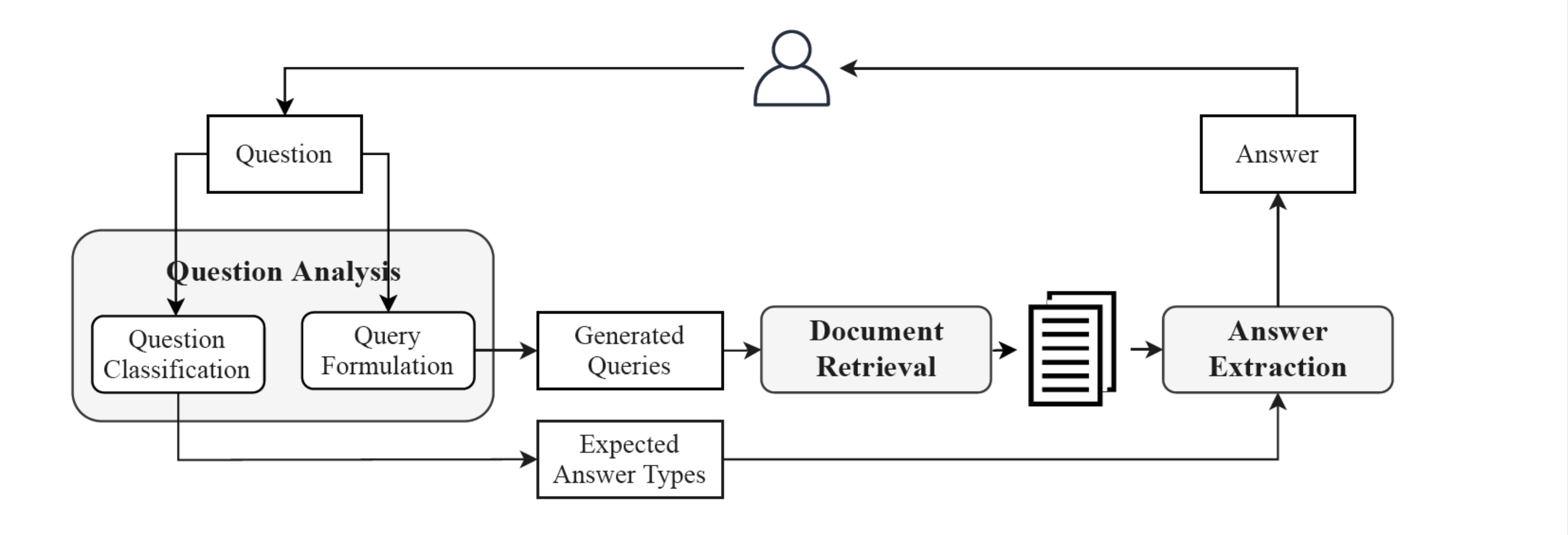
Traditional Architecture of OpenQA

Three stages**: Question Analysis**, **Document Retrieval**, and Answer **Extraction**.

Question Analysis: Given a natural language question it aims to understand the question first so as to facilitate document retrieval and answer extraction in the following stages (for example reformulating the question or classifying the type of question (who, where, when…)).

Document Retrieval stage searches for question-relevant documents based on a IR system, using the search queries generated by Question Analysis.

Answer Extraction is responsible for extracting final answers to user questions from the relevant documents. It can be used some matching methods to detect answers, such as surface text pattern matching, word or phrase matching, and syntactic structure matching.



Neural Network approaches

For **Question Analysis**, some works develop neural classifiers to determine the question types.

For **Document Retrieval**, dense representation based methods have been proposed to address “term-mismatch”. Unlike the traditional methods such as TF-IDF, deep retrieval methods learn to encode questions and documents into a latent vector space where text semantics beyond term match can be measured.

For **Answer Extraction**, as a decisive stage for OpenQA systems to arrive at the final answer, neural models can also be applied. Extracting answers from some relevant documents to a given question essentially makes the task of Machine Reading comprehension (MRC). Current QA technologies for single-turn QA are based on pre-trained transformer models. These models have been pre-trained from unlabeled text. Afterwards, each model can be fine-tuned in specific tasks. Fine-tuning for QA systems is done by modelling the span detection as prediction of the start and end token in the paragraph. The input to the system is a pair of question and paragraph.

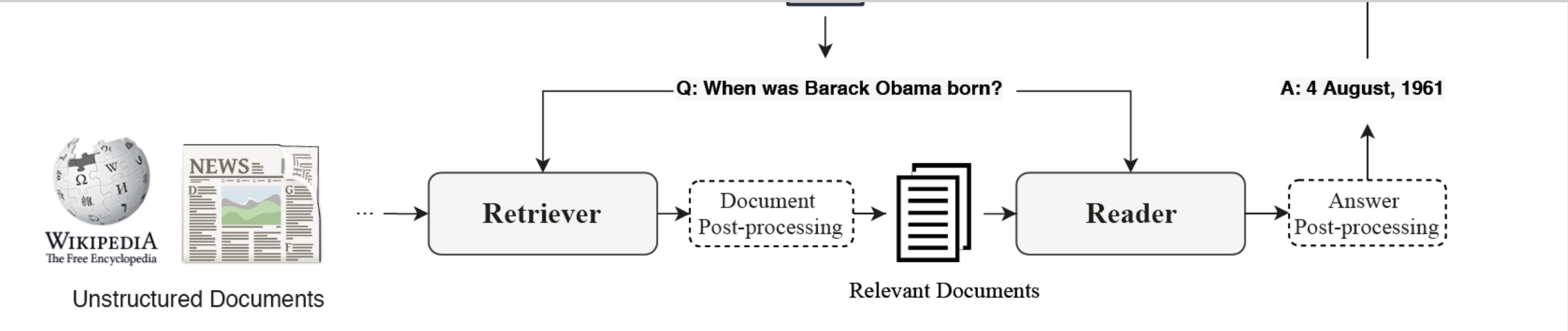
Architecture Retriever-Reader

End-to-end Methods: In recent years, various OpenQA systems have been developed, in which Retriever and Reader can be trained in an end-to-end manner.

Broadly, existing Readers can be categorised into two types: **Extractive Reader** that predicts an answer span from the retrieved documents, and **Generative Reader** that generates answers in natural language using sequence-to-sequence models.

Extractive Reader is based on the assumption that the correct answer to a given question definitely exists in the context, and usually focuses on learning to predict the start and end position of an answer span from the retrieved documents

Generative Reader : It can be used pretrained models and generate answers by taking the input question as well as the documents



Retriever-free

Recent advancement in pre-training Seq2Seq language models brings a surge of improvements for downstream NLG tasks, most of which are built using Transformer-based architectures.

It is able to correctly generate the answer given only a natural language question without fine-tuning.

Conversational OpenQA

Non-conversational OpenQA is challenged by several problems that are almost impossible to resolve, such as the lengthy words for a complex question (e.g. Who is the second son of the first Prime Minister of Singapore?), ambiguity resulting in incorrect response (e.g. When was Michael Jordan born?). These problems would be well addressed under the conversational setting.

Conversational systems are equipped with a dialogue-like interface that enables interaction between human users and the system for information exchange. For
the complex question example given above, it can be decomposed into two simple questions sequentially: “Who is the first Prime Minister of Singapore?” followed by “Who is the second son of him?”. When ambiguity is detected in the question, the conversational OpenQA system is expected to raise a follow-up question for clarification, such as “Do you mean the basketball player?”.