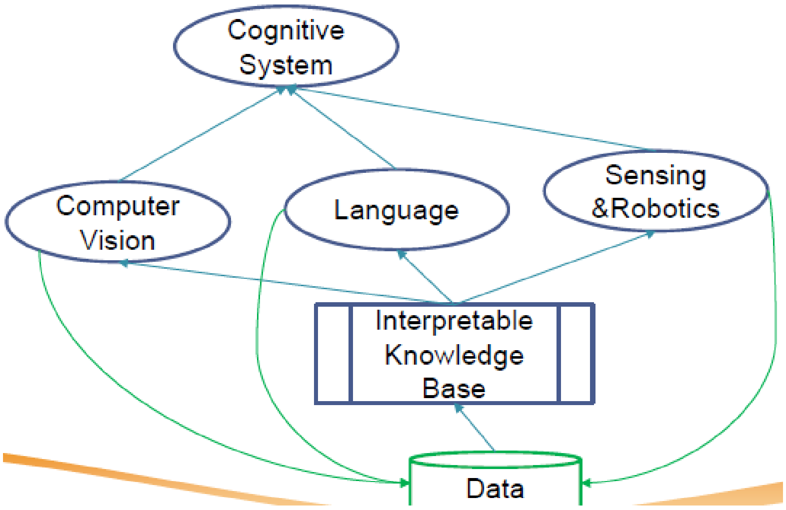
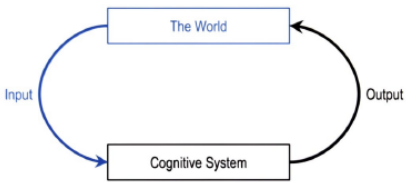
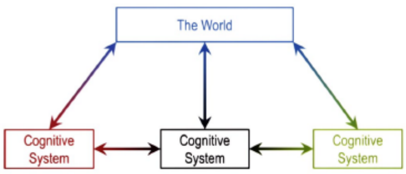
Cognitive Systems

- Cognitive Systems
    - Systems that exhibit human-like intelligence through processes like learning, reasoning, and memory

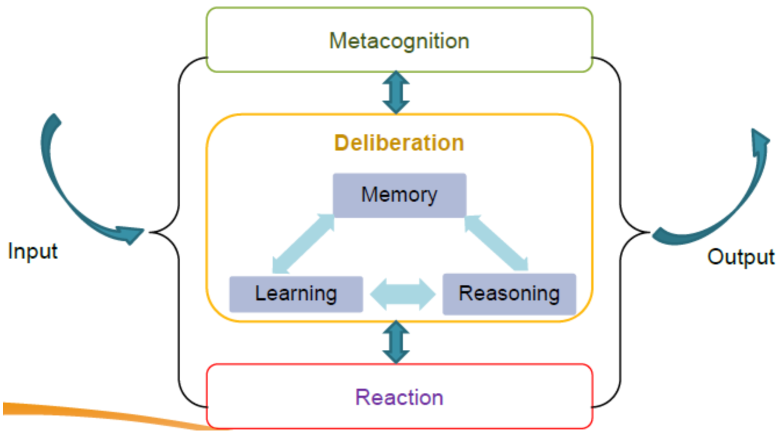
      

    - Architecture

        - CS Architecture
            - Intelligence is about selecting the right kind of action given a particular state of the world.

            - SOAR

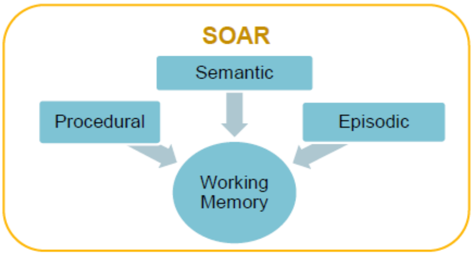
By Newell, Laird, 1983 -> **present**

Representation:

- Graphs of objects and relations

Production System

- Working memory
- Long-Term Memory
  - Procedural
  - Episodic
  - Semantic



- Architecture

Two mugs without graduation

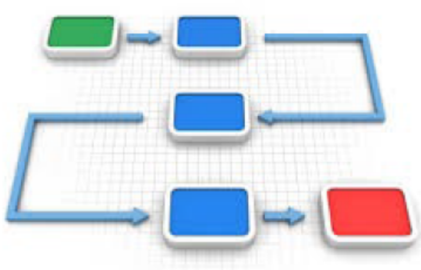
- To get 1 litre water in 3L mug

- Formalize/Symbolize the problem

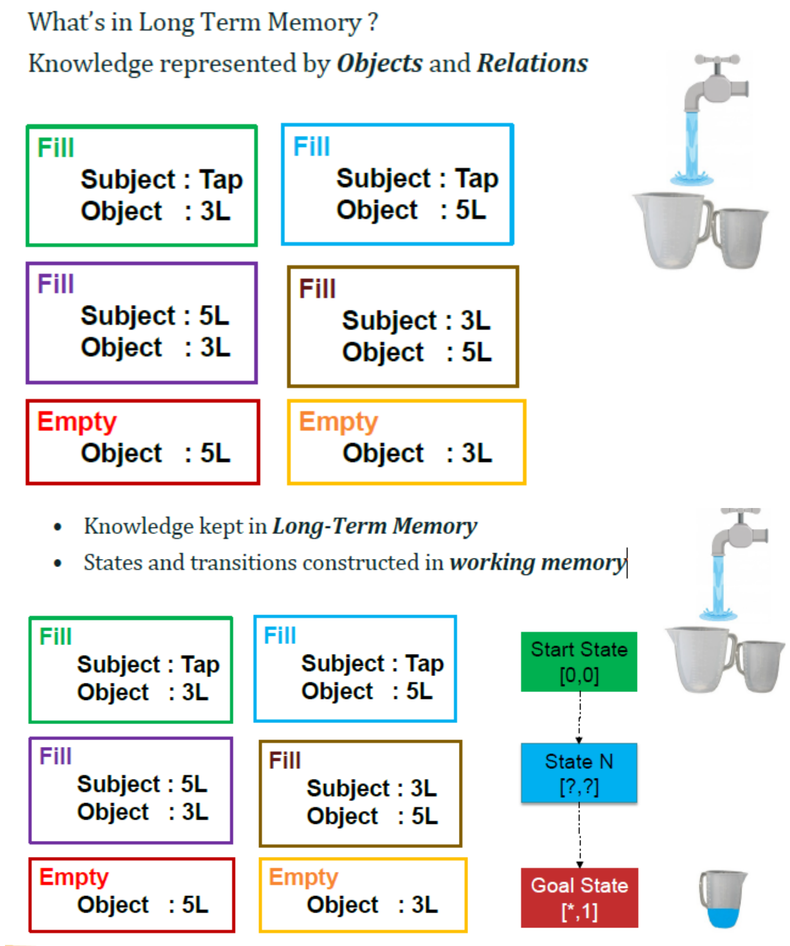
Formalize/Symbolize the problem

- What’s in Working Memory?

What’s in Working Memory?

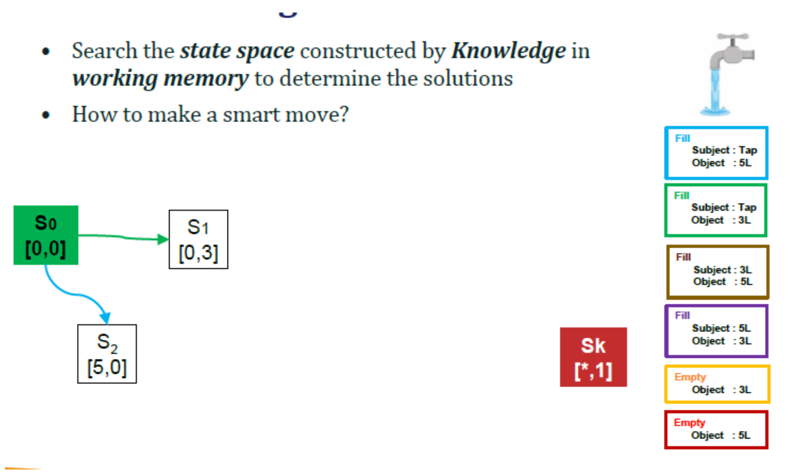


- What’s in Long Term Memory ?



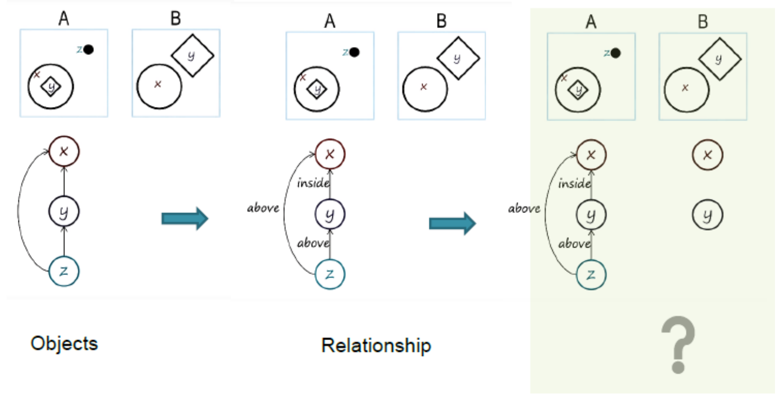
    - Knowledge represented by Objects and Relations
    - Knowledge kept in Long-Term Memory

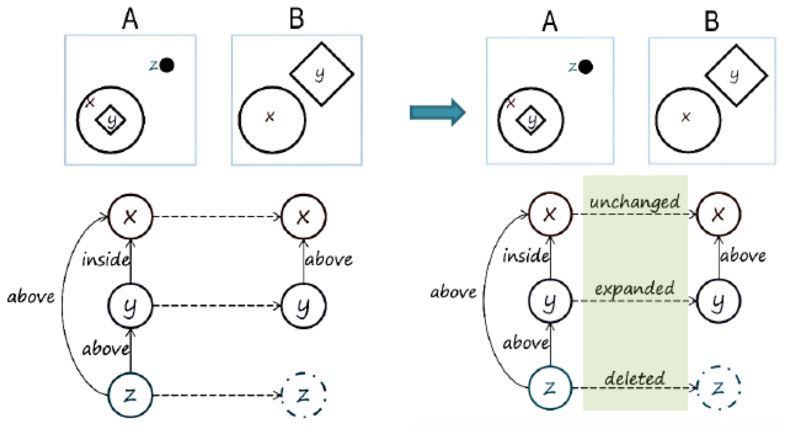
- States and transitions constructed in working memory How to make a smart move?



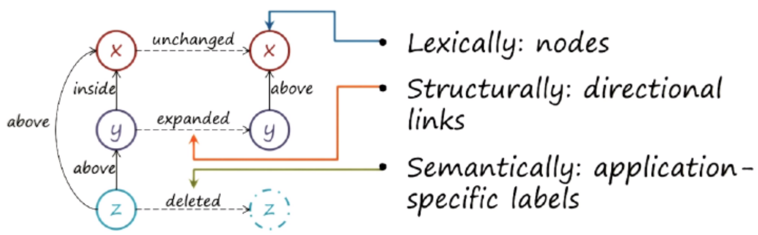
- Summary
    - Knowledge represented by graphs of objects and relations
    - Knowledge kept in Long-Term Memory
    - States and transitions constructed in working memory
    - Search the state space in working memory to determine the solutions
- Knowledge Representation

- Motivation
    - To represent information about the world in a form that a computer system can utilize to solve complex tasks
- Semantic Networks (WordNet)



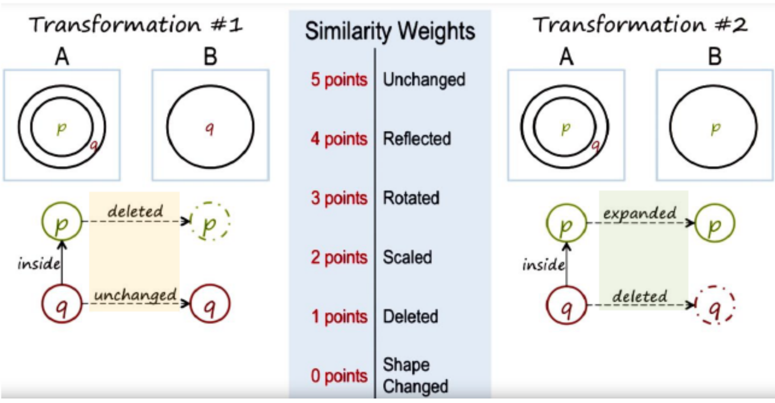


- represents semantic relations between concepts in a network
- Structure

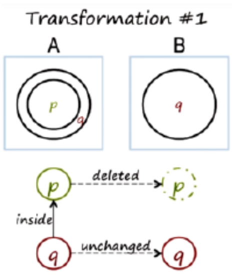


- **Objects** : Vocabulary
- **Links**: directions capturing relationships
- **Labels**: for reasoning

- Choosing matches by weight



- Logical Forms



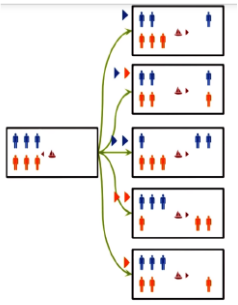
- **Logical Forms:**

Transf_p(Transtep ((Object 'p'),(Action 'deleted'),(Relas (inside 'q'))))
Transf_q(Transtep ((Object 'q'),(Action 'unchanged'),(Relas (outside 'p'))))

- **Computable in JSON-Like Format**

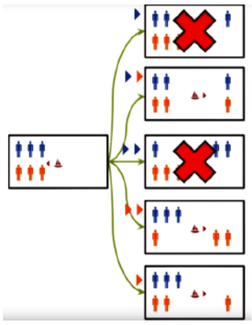
```
{
  "Transf1": [
    {"Object":"p","Action":"deleted","Relas"
      :[{"inside":"q"}]},
    {"Object":"q","action":"unchanged","Relas"
      :[{"outside":"p"}]}
  ]
}
```

- What makes a good relationship
  - Explicit
  - Expose natural constraints
  - Bring objects and relations together
  - Exclude extraneous details
  - Transparent, concise, complete, fast, computable
- Ontology
  - representation of a set of concepts within a domain(typically common sense domain) and the relationships between those concepts.
- Knowledge representation goes hand in hand with automated reasoning
- Generate and Test
  - Generator



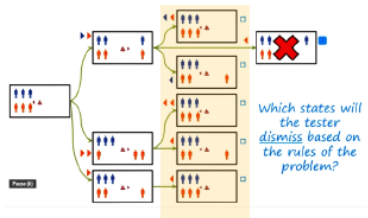
- Generates all the possible solutions
- A dumb generator make things complicated
- More steps (computational power) needed

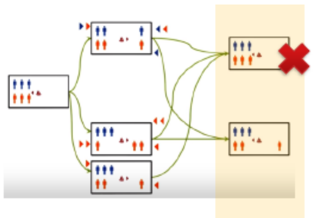
- Tester



- Validate the outputs of generator
- Remove the invalid status

- Generator and Tester can be smarter by:





- Merging duplicated status
- Removing status identical to previous status

- Responsibility can be balanced between Generator and Tester
  - Depending on the number of status

- Frames
  - Properties of Frames

Ashok ate a frog.

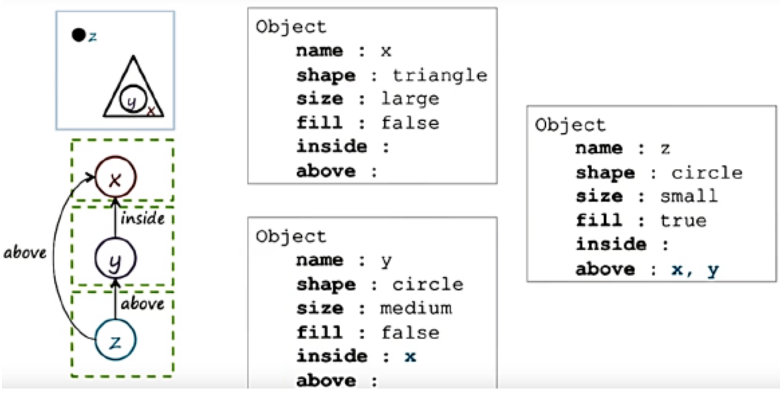
```
Ate
    subject : Ashok
    object : a frog
    location :
    time :
    utensils :
    object-alive : false
    object-is : in-subject
    subject-mood : happy
```

David ate a pizza at home.

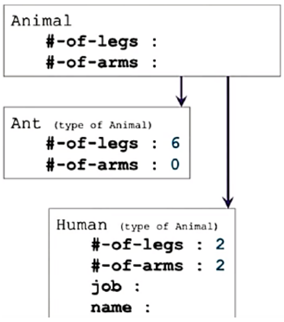
```
Ate
    subject : David
    object : a pizza
    location : at home
    time :
    utensils :
    object-alive : false
    object-is : in-subject
    subject-mood : happy
```

- Slots and Fillers
- Provide default values

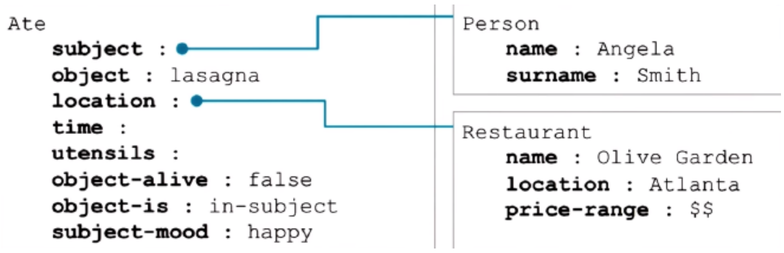
    - Frames represent stereotypes



    - Exhibit inheritance

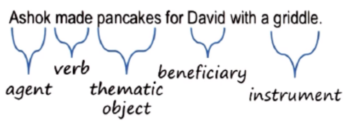


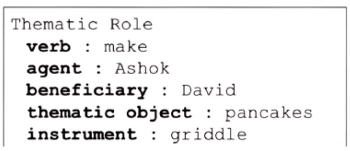
- Complex Frames Systems



    - Structured knowledge representation
    - Carry more information in organized manner

- Thematic Role Systems

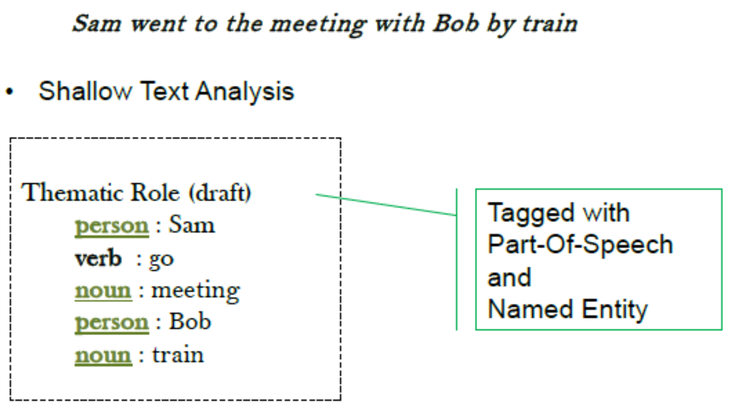


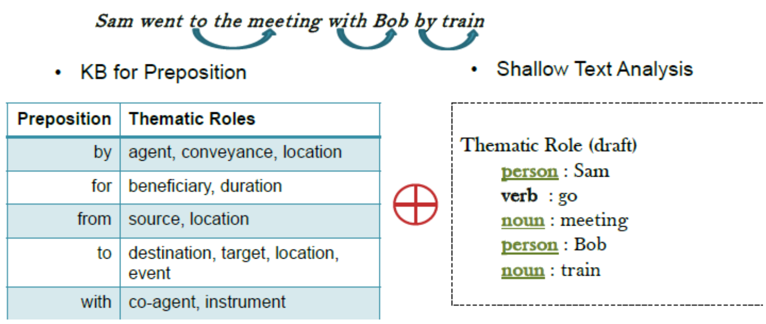


    - A type of Frame system
    - Focusing on verbs
    - Semantic slots/roles
    - Resolving ambiguity
    - Major theta roles include (but not limited to):
        - Agent – The entity that intentionally carries out the action of the verb.

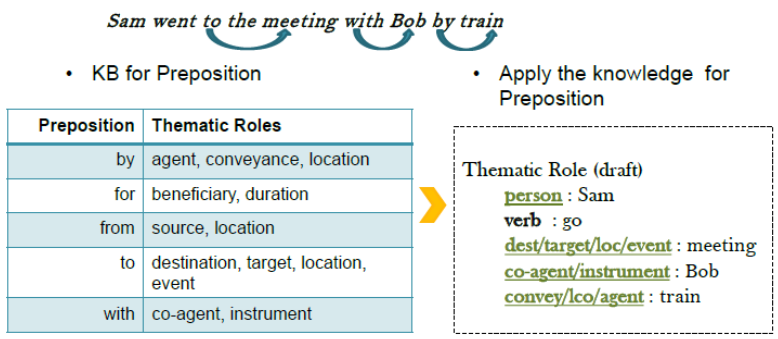
- Experiencer – The entity that undergoes an emotion, a state of being, or a perception expressed by the verb.
- Theme – The entity that directly receives the action of the verb.
- Instrument – The entity by which the action of the verb is carried out.
- Goal – The direction towards which the action of the verb moves.
- Source – The direction from which the action originates.
- Location – The location where the action of the verb takes place.
- Benefactive – The entity that receives a concrete or abstract element as a result of the action of the verb

- Common Sense Reasoning in AI
  - Sense making
    - Encoding text into Frames with the help of Knowledge Base



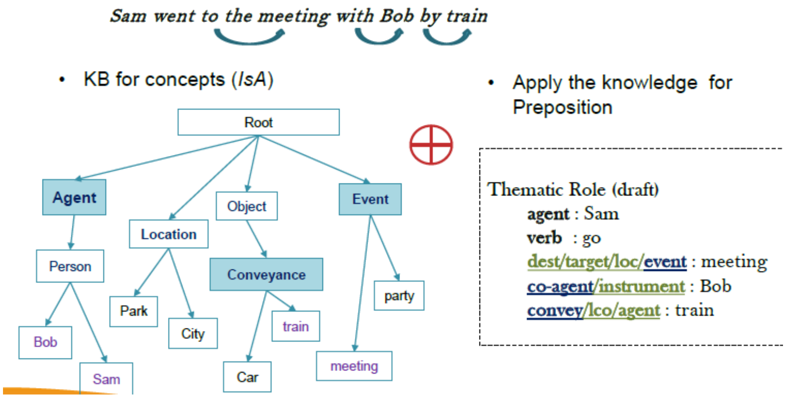


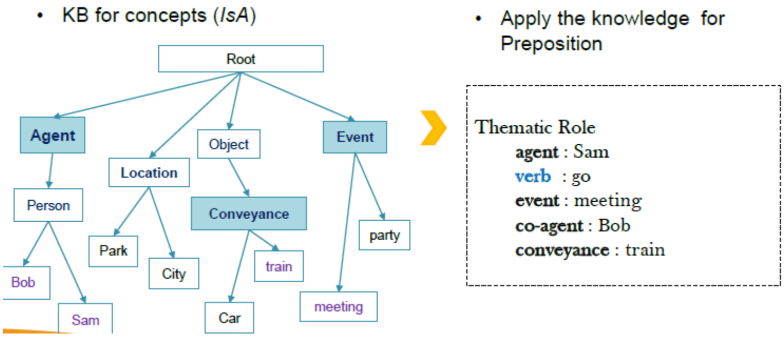
Preposition	Thematic Roles
by	agent, conveyance, location
for	beneficiary, duration
from	source, location
to	destination, target, location, event
with	co-agent, instrument



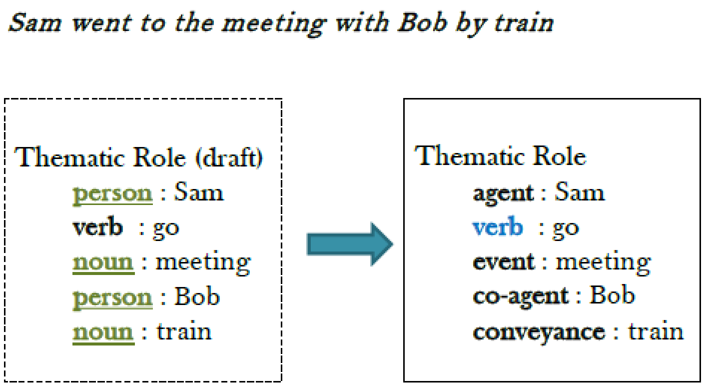
Preposition	Thematic Roles
by	agent, conveyance, location
for	beneficiary, duration
from	source, location
to	destination, target, location, event
with	co-agent, instrument

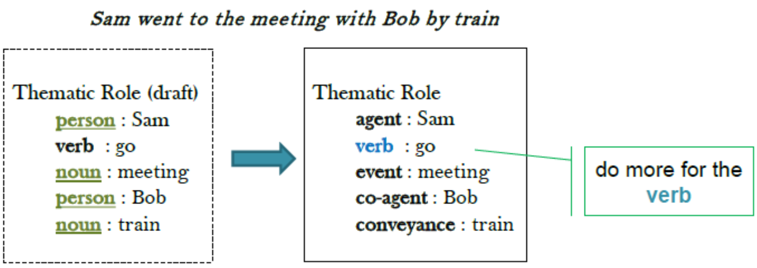
    - Build the knowledge base



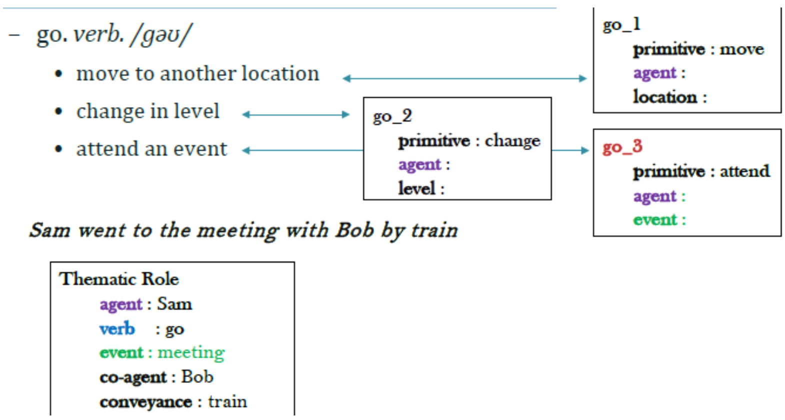


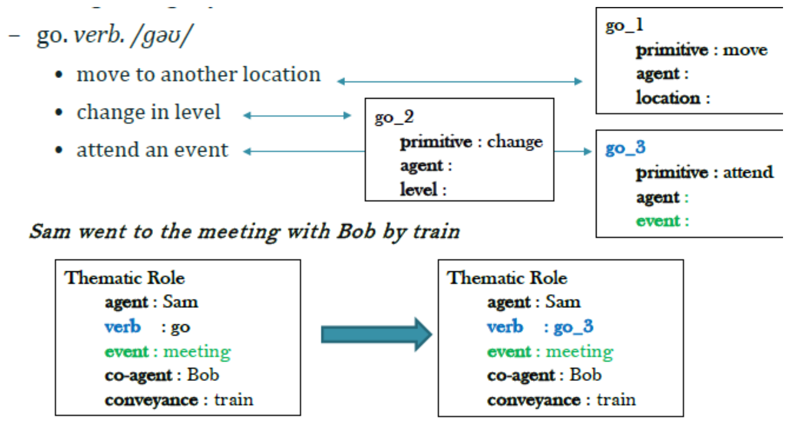
- Apply the knowledge



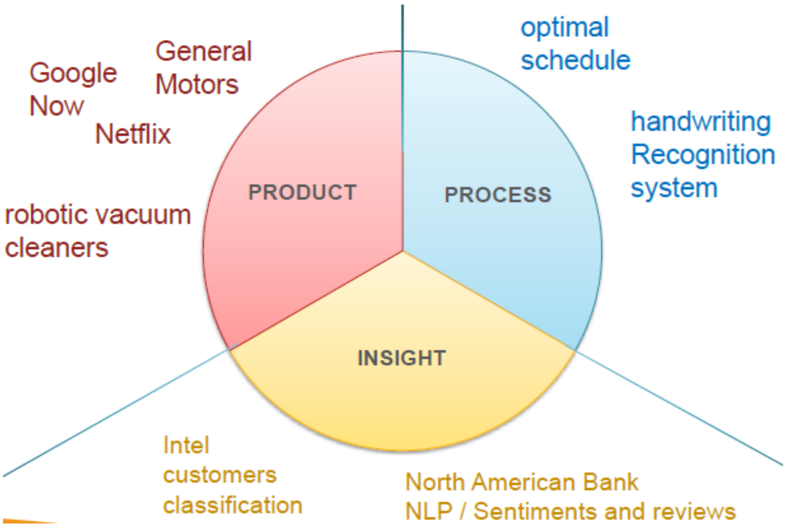


- Resolving Ambiguity in Verbs with Primitive Actions





- How to apply AI



  - Product
    - applications embed cognitive technologies in a product or service providing in customer benefits like ease of use, simplicity, or automation.
  - Process
    - applications embed the technology in an organization's workflow automating some tasks to get things done faster, better, cheaper, or some combination.
  - Insight
    - applications use advanced analytic capabilities and machine learning to uncover insights to make better operational and strategic decisions based on large mounts of data

- Whether and Where to apply AI

VIABLE	VALUABLE	VITAL
Vison Speech Handwriting	specialists in rare cancers drilling engineers in oil	spam filtering fraud detection
Forecasting Document review	medical decision-making financial decision-making	processing large volumes of handwritten or printed forms
Data-driven decisions Scheduling	deliver features or experiences that your customers care about.	analyzing large amounts of social media text

- NLP Tasks
  - intent detection ->classification

- Brief

	amazing	service	lost	glamour	disappoint	brilliant	super	expensive	noisy	...
Doc1	1	1	0	0	0	1	0	0	0	
Doc2	0	0	1	1	1	0	0	1	0	
Doc3	0	0	0	1	0	0	1	0	0	
Doc4	0	0	0	0	2	0	0	1	1	
...										

  - Using labelled data to build machine learning models that can classify input (sentence/doc) into intent classes (supervised learning)
  - Popular techniques: SVM/NB/LR/DT/KNN
  - Pre-process the input text into features (vector model)
- Pre-processing
  - Tokenization

    What is the weather in Seattle today?

    ['What', 'is', 'the', 'weather', 'in', 'Seattle', 'today', '?']

    - To break a stream of characters into tokens
    - This is done by identifying token delimiters
      - Whitespace characters such as space, tab, newline
      - Punctuation characters like ( ) < > ! ? “ ”
      - Other characters . , : -‘ ’ etc.
  - Case normalisation

    Case normalization: convert all tokens to lower case to remove the variation of words due to case differences.

    ['what', 'is', 'the', 'weather', 'in', ‘seattle', 'today', '?']

  - Lemmatization/stemming

    Lemmatization/stemming

    – To reduce the words to its root form

    – E.g. classes -> class, ran -> run , production -> produce

    ['what', ‘be', 'the', 'weather', 'in', ‘seattle', 'today', '?']

  - Punctuation removal

    Punctuation removal

    ['what', ‘be', 'the', 'weather', 'in', ‘seattle', 'today']

  - Stopword removal

Stopword removal

- To remove extremely common words (with little meaning) like functional words (the, a, of...)

['weather', 'seattle', 'today']

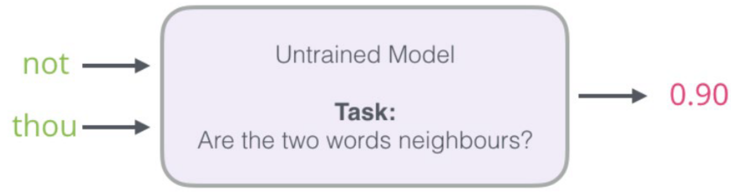
- To remove extremely common words (with little meaning) like functional words (the, a, of…)

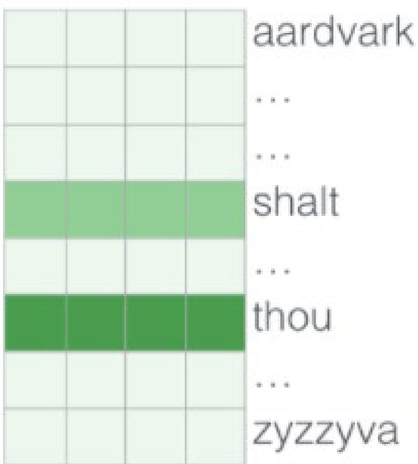
- Indexing: Creating Vector Representations

$$
\begin{array}{ccccc}
 & T_1 & T_2 & \cdots & T_t \\
D_1 & w_{11} & w_{21} & \cdots & w_{t1} \\
D_2 & w_{12} & w_{22} & \cdots & w_{t2} \\
\vdots & \vdots & \vdots & & \vdots \\
\vdots & \vdots & \vdots & & \vdots \\
D_n & w_{1n} & w_{2n} & \cdots & w_{tn}
\end{array}
$$

T: term
D: document
w: weight of the term

  - Creating vector representation of documents (term-document matrix) using “bag-of-words” approach
  - Usually only content words (adjectives, adverbs, nouns, and verbs) are used as vector features
  - Word Embeddings
    - Words represented as vectors of real numbers in a continuous vector space with a much lower dimension
    - Learned by deep neural networks during a prediction task e.g. Word2Vec





  - Term Weighting

- Binary
    - 0 or 1, simply indicating whether a word has occurred in the document (but that's not very helpful).
- Frequency-based

	amazing	service	lost	glamour	disappoint	brilliant	super	expensive	noisy	...
Doc1	1	1	0	0	0	1	0	0	0	
Doc2	0	0	1	1	1	0	0	1	0	
Doc3	0	0	0	1	0	0	1	0	0	
Doc4	0	0	0	0	2	0	0	1	1	
...										

    - term frequency, the frequency of words in the document, which provides additional information that can be used to contrast with other documents.
- tf-idf Indexing

$$tf\text{-}idf_{t,d} = tf_{t,d} * idf_t \qquad idf_t = \log \frac{N}{df_t}$$

– $tf_{t,d}$: term frequency – number of occurrences of term t in document d

– idf_t : inverted document frequency of term t

N : the total number of documents in the corpus

df_t : the document frequency of term t, i.e., the number of documents that contain the term.

    - To modify the frequency of a word in a document by the perceived importance of the word(the inverse document frequency), widely used in information retrieval
        - When a word appears in many documents, it's considered unimportant.
        - When the word is relatively unique and appears in few documents, it's important.
    - example

TERM VECTOR MODEL BASED ON $w_i = tf_i * IDF_i$

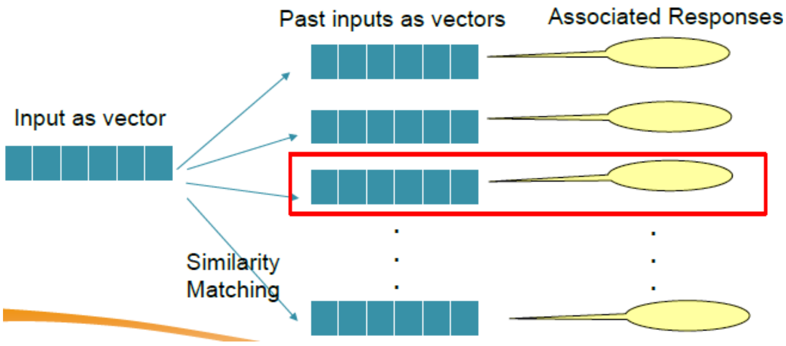
Query, Q: "gold silver truck"
D_1: "Shipment of gold damaged in a fire"
D_2: "Delivery of silver arrived in a silver truck"
D_3: "Shipment of gold arrived in a truck"
D = 3; IDF = $\log(D/df_i)$

	Counts, tf_i							Weights, $w_i = tf_i * IDF_i$			
Terms	Q	D_1	D_2	D_3	df_i	D/df_i	IDF_i	Q	D_1	D_2	D_3
a	0	1	1	1	3	3/3 = 1	0	0	0	0	0
arrived	0	0	1	1	2	3/2 = 1.5	0.1761	0	0	0.1761	0.1761
damaged	0	1	0	0	1	3/1 = 3	0.4771	0	0.4771	0	0
delivery	0	0	1	0	1	3/1 = 3	0.4771	0	0	0.4771	0
fire	0	1	0	0	1	3/1 = 3	0.4771	0	0.4771	0	0
gold	1	1	0	1	2	3/2 = 1.5	0.1761	0.1761	0.1761	0	0.1761
in	0	1	1	1	3	3/3 = 1	0	0	0	0	0
of	0	1	1	1	3	3/3 = 1	0	0	0	0	0
silver	1	0	2	0	1	3/1 = 3	0.4771	0.4771	0	0.9542	0
shipment	0	1	0	1	2	3/2 = 1.5	0.1761	0	0.1761	0	0.1761
truck	1	0	1	1	2	3/2 = 1.5	0.1761	0.1761	0	0.1761	0.1761

- Model Building
    - Pipeline
        - Divide the labelled data (inputs and their intent classes) into training set, validation set, testing set

- Select a classification algorithm (e.g. SVM) and train a classifier
- Tune the parameters of the model using validation set
- Test the final model performance using test set

- Similarity Based Intent Identification



- Given vector representations of inputs (tf-idf, embeddings, etc), a quite common approach is to detect the intent of an input based on its similarity to the existing inputs with known intents or even responses.
- Cosine Similarity

A similarity measure between two vectors (input and candidate response)

measuring the cosine of the angle between them

$$Sim(D_i, D_j) = \frac{D_i \bullet D_j}{|D_i| * |D_j|} = \frac{\sum_k w_{ki} w_{kj}}{\sqrt{\sum_k w_{ki}^2 \sum_k w_{kj}^2}}$$

Example: Given 3 vectors shown here

$$|D_1| = \sqrt{0.1761^2 + 0.4771^2 + 0.1761^2} = \sqrt{0.2896} = 0.5382$$

$$|D_2| = \sqrt{0.4771^2 + 0.4771^2 + 0.1761^2 + 0.1761^2} = \sqrt{0.5173} = 0.7192$$

$$|D_3| = \sqrt{0.1761^2 + 0.4771^2 + 0.9542^2 + 0.1761^2} = \sqrt{1.2001} = 1.0955$$

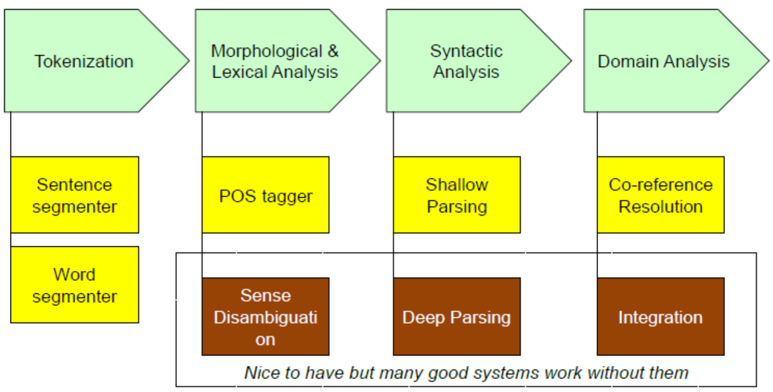
$Sim(D_1, D_2)$ = (0.1761*0.1761)/(0.5382*0.7192)=0.0801

$Sim(D_1, D_3)$ = (0.4771*0.9542+0.1761*0.1761)/(0.5382*1.0955)=0.8246

D_1	D_2	D_3
0	0	0
0	0	0.1761
0	0.4771	0
0	0	0.4771
0	0.4771	0
0.1761	0.1761	0
0	0	0
0	0	0
0.4771	0	0.9542
0	0.1761	0
0.1761	0	0.1761

- slot detection
  - Information Extraction -the automatic extraction of (possibly pre-specified) information from natural language documents
    - Facts about types of entities, events, relationships
    - definition
      - Name Entity = lowest level of recognition by an IE system
        - Normally recognized by dictionaries or rules
      - Concept= rule or heuristic to create an abstraction
        - Sometimes called a “natural class” = different people at different times and in different places would refer to the same referent with that concept
        - “president of the United States” vs. “president of the United Kingdom
      - Information= words, named entities, concepts which fulfill a need

      - So if you have a question, and a phrase answers that question, then that phrase is an example of information
      - Information is often regular, i.e., with a pattern
  - Approaches
    - Rule-based Systems
      - Hand-coded rules
        - Coded by linguists, with domain input
        - Iterative method based on document inspection
        - Slow but very good results
      - Induced (machine learning) rules
        - Fully machine learning
          - Given an annotated corpus, derive a basis set of rules that cover a pre-determined % of the annotated examples (and only the annotated examples)
          - Heuristic approach: one rule at a time!
        - Hybrid systems –machine learning to fine-tune the rules
    - Statistics-based Systems
      - Start with a well-annotated corpus
      - Depending on the method (e.g., Hidden Markov Models), derive statistical rules to create a model that generates the examples
      - Advantages compared to Rule based systems
        - Language independent (within representational limits)
        - No linguistic or domain knowledge needed in the team
        - Relatively small effort in creating the models
      - Issues
        - The complexity moves to the corpus –must be well annotated and must cover the full space of possibilities
        - Requires very large number of training examples to get good results
- Component



- Named Entity Recognition
  - Recognition of particular types of proper noun phrases, specifically persons, organizations, locations, and sometimes money, dates, times, and percentages.
  - Very useful in text mining applications, by turning verbose text data into a more compact structural form
  - example

    [LOC Houston] , Monday, July 21 -- Men have landed and walked on the moon. Two [MISC Americans] , astronauts of [ORG Apollo] 11, steered their fragile four-legged lunar module safely and smoothly to the historic landing yesterday at 4:17:40 P.M., Eastern daylight time. [PER Neil A. Armstrong] , the 38-year-old civilian commander, radioed to earth and the mission control room here: "[LOC Houston] , [ORG Tranquility Base] here; the Eagle has landed."

    Generated by UIUC NER system
  - Rule-based NER
    - Rule-based systems can and do work well
      - Corpus is relatively static (in terms of vocabulary, language structure, etc.)
      - Can be fast especially in well-defined limited domains(compared to annotating training examples)
    - A typical rule-based system comprises
      - Set of rules
      - Policies to control when and how (multiple) rules are applied, e.g., order, looping.
    - example
      - Lexical pattern matching
      - Form:
        - Match(pattern) then Do(action)

      ```
      Rule: Company1                                          from gate.ac.uk
        ( ( {Token.orthography == upperInitial} )+
            {Lookup.kind == companyDesignator}
        ):match
       -->
          :match.NamedEntity = { kind=company, rule="Company1" }
      ```
  - statistics based systems
    - Many top performing systems are statistics based
      - Machine learning (ML) on very large corpora is state-of-the-art
    - Annotation based corpora for training
      - You have a well annotated corpora with many features
      - Various ML techniques from simple to sophisticated

- Relatively homogeneous real data (not training data) in any given domain. Note that models don’t transfer well across domains
- You don’t have domain or language resources in that area

- Popular models

  - Hidden Markov Models (HMM)
    - Simple, joint probability
  - Conditional Random Fields (CRF)
    - Conditional probability
    - Considers features of current token, and of preceding *n* tokens (window=*n*)
  - Similarity algorithms
    - Measure distance of group of words to a dictionary list
    - Works especially well for jargon and other terminology
  - Support Vector Machines (SVM)
    - Training method for standard perceptron
    - Optimize the points to determine the hyperplane dividing the positive training samples from the negative ones

```
Alex I-PER
is O
going O
to O
Los I-LOC
Angeles I-LOC
in O
California I-LOC
```

- POS Tagging(Entity Labelling)
  - To determine POS or grammatical category of a term
    - Nouns, verbs, adjectives, adverbs, pronouns, determiners, prepositions, conjunctions, etc.
    - LDC Penn Tree Bank has 36 categories with detailed information

CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential *there*
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative

UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singula present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun

  - Dictionary with word-POS correspondence is needed
  - Challenge –POS disambiguation (words with >1 POS)
    - E.g. “book” can be a noun (“my book”) or a verb (“to book a room”)
    - Example:
      - About six and a half hours later, Mr. Armstrong opened the landing craft's hatch, stepped slowly down the ladder and declared as he planted the first human footprint on the lunar crust: "That's one small step for man, one giant leap for mankind.”

IN/ About CD/ six CC/ and DT/ a JJ/ half NNS/ hours RB/ later ,/ , NNP/ Mr. NNP/ Armstrong VBD/ opened DT/ the NN/ landing NN/ craft POS/ 's NN/ hatch ,/ , VBD/ stepped RB/ slowly IN/ down DT/ the NN/ ladder CC/ and VBD/ declared IN/ as PRP/ he VBD/ planted DT/ the JJ/ first NN/ human NN/ footprint IN/ on DT/ the NN/ lunar NN/ crust :/ : ``/ " DT/ That VBZ/ 's CD/ one JJ/ small NN/ step IN/ for NN/ man ,/ , CD/ one JJ/ giant NN/ leap IN/ for NN/ mankind ./ . "/ "

Generated by UIUC POS Tagger

  - Taggers
    - Rule-based -e.g. Brill’s tagger by Eric Brill
      - Error-driven transformation-based tagger
      - Initially assign the most frequent tag to each word, based on dictionary and morphological rules
      - Contextual rules are then applied repeatedly to correct any errors
    - Stochastic taggers –e.g. CLAWS, Viterbi, Baum-Welch, etc.
      - based on Hidden Markov Models (HMMs) and n-gram probabilities
      - Manually tagged corpus is needed to estimate probabilities
    - Many machine learning methods have also been applied
    - Stanford’s Statistical NLP website lists many free taggers
- Shallow Parsing / Chunking(Entity Linking)
  - To identify phrases in a text (noun phrases, verb phrases, and prepositional phrases, etc.)
  - Example:

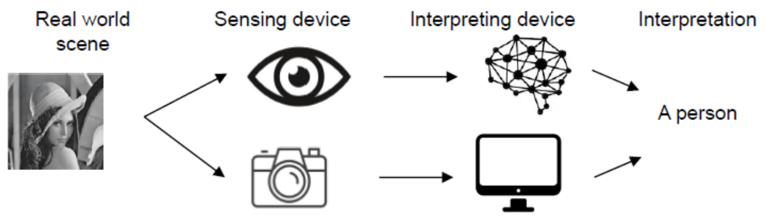
    

    - About six and a half hours later, Mr. Armstrong opened the landing craft's hatch, stepped slowly down the ladder and declared as he planted the first human footprint on the lunar crust: "That's one small step for man, one giant leap for mankind.”
  - After morphological analysis and disambiguation, using information of lemmata, morphological information, and word order configuration
  - Largely stochastic techniques based on probabilities derived from an annotated corpus
  - Avoiding the complexity of full parsing, faster, more robust
  - Useful in Information Extraction, Summary Generation, and Question Answering
- Co-reference Resolution
  - Determine relationship between entities which are related
    - Identity relation (morning star vs. evening star)
    - Whole-part relation
  - Simple version
    - Determine entities which have the same referent
      - Anaphora (Pronouns)
      - Proper names, proper nouns, noun phrases,…
    - Definite descriptions (may be time dependent)
      - UsainBolt & “the fastest man in the world”

        - examples

          > *Chatbot: Hello! Nice to see you here! How can I help you?*
          >
          > *You: Hi! I would like to know when Barack Obama was born.*
          >
          > *Chatbot: He was born on August 4, 1961.*
          >
          > *You: Well, when was his wife born?*
          >
          > *Chatbot: I don't understand what you said, please make it clear for me.*
          >
          > *You:*

    - Common Strategy for Slot Detection
      - If values of slots are standard entities like location names and human names, an off-the-shelf NER module will be able to detect them.
      - Otherwise, training data with such entities labelled is required to train a recognizer for them.
      - Often domain specific rules turn out to be very useful in capturing slot values that are not NEs.
      - After identifying intent and collecting necessary information (slot values), determine how to proceed
      - Options:
        - acting upon the new information directly and produce a reply
        - remembering an incomplete interpretation and waiting to see what happens next
        - seeking out information to fill in the blanks
        - asking the speaker for clarification
- SPEECH/VISION
  - Human & Computer

Computers	Brains
Fixed architecture	Evolving architecture
Modular, (primarily) serial	Massively parallel
Separate hardware, software	No distinction between hardware and software
Separate computation, memory	No distinction between computation and memory

  - Vision cognitive system pipeline
    - Image acquisition

- Image enhancement

- Image processing and analysis

- Feature representation & description

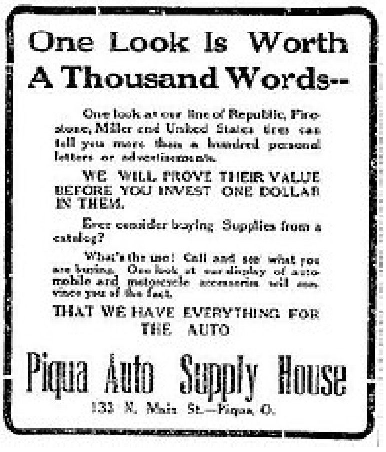
Online demo: http://demo.ipol.im/demo/my_affine_sift/

- Object detection and recognition

Three fundamental tasks

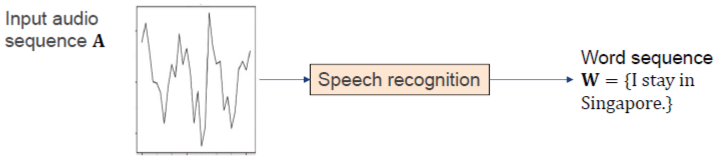
- Classification
- Detection
- Segmentation

- Scene understanding



- Automatic speech recognition
  - Challenges of speech recognition
    - Style: Read speech or spontaneous (conversational) speech?
    - Continuous natural speech or command & control?
    - Speaker characteristics: Rate of speech, accent, prosody (stress, intonation), speaker age, pronunciation variability even when the same speaker speaks the same word
    - Channel characteristics: Background noise, room acoustics,microphone properties, interfering speakers
    - Task specifics: Vocabulary size (the number of words to be recognized), language-specific complexity, computational resource limitations
  - pipeline
    - Recognize the word sequence given the input audio sequence.

Objective: Recognize the word sequence given the input audio sequence.



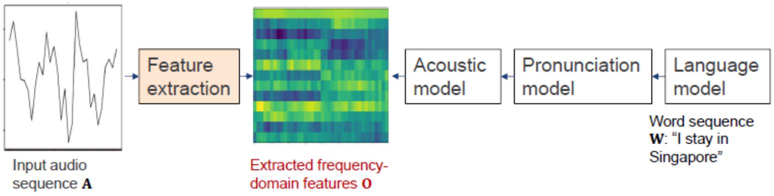
Let $\mathbf{A}$ represent an audio sequence and $\mathbf{W}$ denote a word sequence, then the speech recognizer decodes $\mathbf{W}^*$ as

$$\mathbf{w}^* = \underset{\mathbf{W}}{\operatorname{argmax}} P(\mathbf{W}|\mathbf{A}) = \underset{\mathbf{W}}{\operatorname{argmax}} \frac{P(\mathbf{A}|\mathbf{W})P(\mathbf{W})}{P(\mathbf{A})} \propto \underset{\mathbf{W}}{\operatorname{argmax}} P(\mathbf{A}|\mathbf{W})\mathrm{P}(\mathbf{W})$$

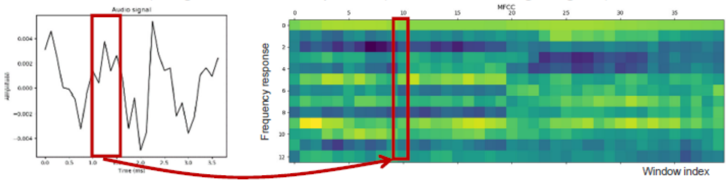
Further introduce acoustic features $\mathbf{O}$, phoneme $\mathbf{L}$, the optimization problem statement can be rewritten to be

$$\mathbf{w}^* = \underset{\mathbf{W}}{\operatorname{argmax}} P(\mathbf{A}|\mathbf{W})\mathrm{P}(\mathbf{W}) = \underset{\mathbf{W}}{\operatorname{argmax}} P(\mathbf{A}|\mathbf{O})P(\mathbf{O}|L)P(\mathbf{L}|\mathbf{W})P(\mathbf{W})$$

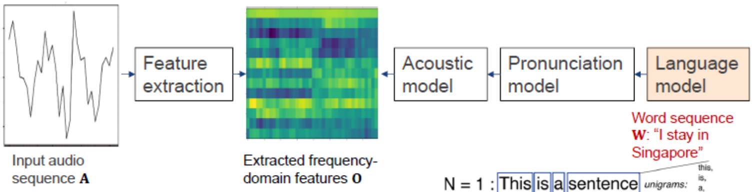
    - Feature extraction

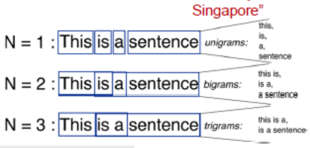


Speech signal represented in time-domain (left figure) and frequency-domain (right figure), e.g., *Mel frequency cepstral coefficient* (MFCC), where a sliding window is applied to select a short interval signal then apply a frequency transformation (e.g., Fourier transform) to generate the response (one column in right figure).



- language model





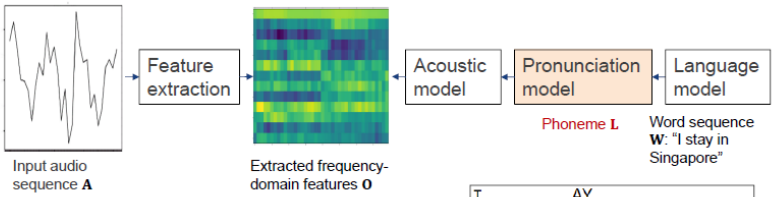
N-gram models: Build the language model by calculating probabilities from text training corpus: How likely is one word to follow another.

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

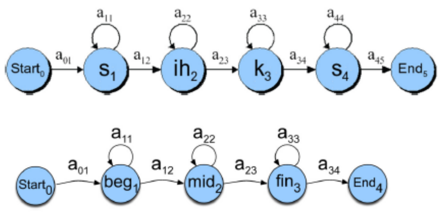
Example: Bi-gram in the Berkeley Restaurant Project corpus of 9332 sentences.

Reference: https://deepai.org/machine-learning-glossary-and-terms/n-gram

- pronunciation model



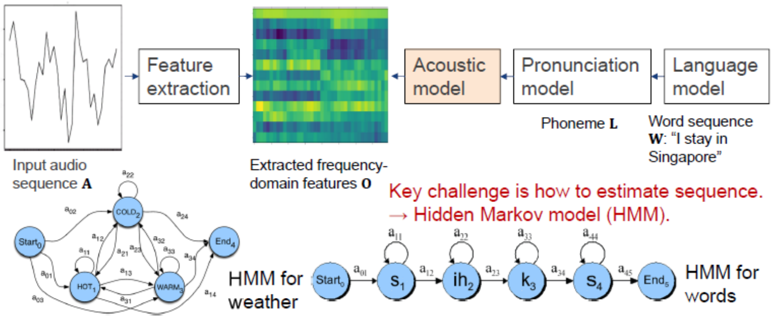

```
I         AY
STAY      S T EY
IN        IH N
SINGAPORE S IH NG AH P AO R
```



Reference: http://www.speech.cs.cmu.edu/cgi-bin/cmudict

The Carnegie Mellon University Pronouncing Dictionary is an open-source machine-readable pronunciation dictionary for North American English that contains over 134,000 words and their pronunciations.

- Acoustic model



$Q = \{q_1, q_2, \cdots, q_T\}$	A set of N states for observations	Each observation has one state
$A = \{a_{11}, a_{12}, \cdots, a_{nn}\}$	A state transition probability matrix A, each a_{ij} representing the probability of moving from the state i to the state j, $s.t. \sum_{j=1}^{n} a_{ij} = 1$	Learned from speech training dataset
$O = \{o_1, o_2, \cdots, o_T\}$	A sequence of T observations	Observed speech data
$B = b_i(o_t)$	An observation likelihood, called emission probability, representing the probability of an observation o_t being generated from a state i	Learned from speech training dataset
S	A set of states (e.g., HOT_1, $COLD_2$, $WARM_3$, S_1, ih_2, k_3, etc), a special start state $Start_0$ and an end state End_4 that are not associated with observations, together with their transition probabilities out of the start state and into the end state.	

- HMM
  - example

State pransition probability				Observation likelihood		
Today weather	Tomorrow weather			Weather	Probability of	
	Sunny (S)	Raining (R)	Cloudy (C)		Umbrella (U)	No umbrella (N)
Sunny (S)	0.8	0.05	0.15	Sunny (S)	0.1	0.9
Raining (R)	0.2	0.6	0.2	Raining (R)	0.8	0.2
Cloudy (C)	0.2	0.3	0.5	Cloudy (C)	0.3	0.7

Q: Given that today weather is S, what is the probability that tomorrow is S and the day after is R?

Markov assumption

$$P(q_2 = S, q_3 = R | q_1 = S) = P(q_3 = R | q_2 = S, q_1 = S) P(q_2 = S | q_1 = S)$$
$$= P(q_3 = R | q_2 = S) P(q_2 = S | q_1 = S) = 0.05 \times 0.8 = 0.04$$

Q: Given that you don't use umbrella (N) for three days, calculate the probability for the weather on these three days to be $\{q_1 = S, q_2 = C, q_3 = S\}$. Note that the prior probability for the start state as sunny (S) on day one is assumed to be 1/3 (three weather has the same probability).

$$P(q_1 = S, q_2 = C, q_3 = S | o_1 = N, o_2 = N, o_3 = N)$$
$$= P(o_1 = N | q_1 = S) P(o_2 = N | q_2 = C) P(o_3 = N | q_3 = S) P(q_1 = S) P(q_2 = C | q_1 = S) P(q_3 = S | q_2 = C)$$
$$= 0.9 \times 0.7 \times 0.9 \times 1/3 \times 0.15 \times 0.2 = 0.0057$$

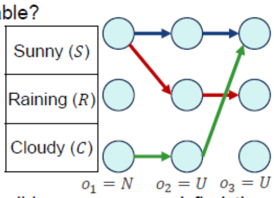
  - Sequence estimation

Q: Given that three days your umbrella observations are: {no umbrella (N), umbrella (U), umbrella (U)}, find the most probable weather-sequence.

Idea 1: If we ignore the weather as a 'sequence' and treat each day weather separately, the most probable weather are Sunny (S), Raining (R), Raining (R).

Idea 2: Exhaustively evaluate probability of each sequence. For example, consider following three possible sequences, which is most probable?

- Blue sequence: Sunny (S), Sunny (S), Sunny (S)
- Red sequence: Sunny (S), Raining (R), Raining (R)
- Green sequence: Cloudy (C), Cloudy (C), Sunny (S)



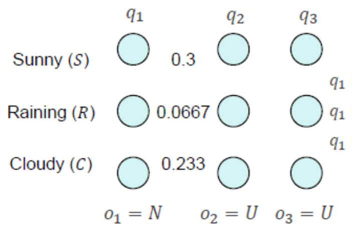
Idea 3: Design an efficient method to evaluate all possible sequence and find the most probable one.
→ We will study Viterbi algorithm in next few slides.

Viterbi: A single-line $.predict(O)$ function in hmmlearn library

  - Viterbi algorithm

Key idea: "Optimal policy is composed of optimal sub-policies".

1. Initialization: Calculate probability of the first day state based on first day observation and (assumed to be equal) prior probability starting from all possible states.
2. Recursion: For all following days, calculate probability of each state based on current observation and the largest (previous state probability × transition probability) from the previous day. Record the 'best path' ending at current state from the previous day.
3. Termination and back tracing: For the last day, choose the state with the highest probability. Trace back according to the recorded most probable path.



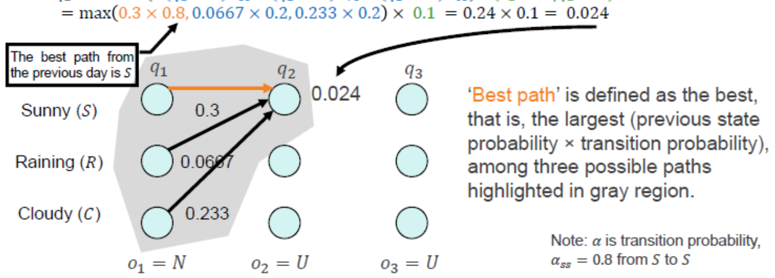
q_1 is S: $P(q_1 = S) \times P(o_1 = N|q_1 = S) = \frac{1}{3} \times 0.9 = 0.3$

q_1 is R: $P(q_1 = R) \times P(o_1 = N|q_1 = R) = \frac{1}{3} \times 0.2 = 0.0667$

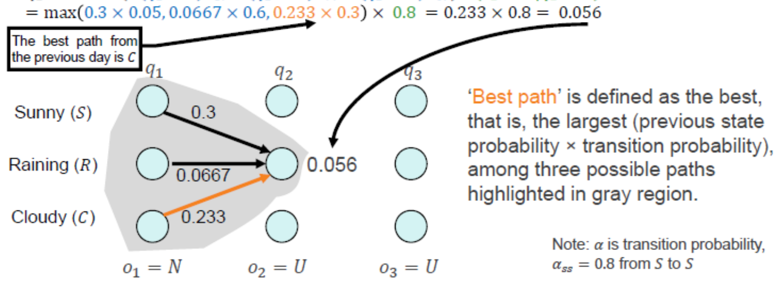
q_1 is C: $P(q_1 = C) \times P(o_1 = N|q_1 = C) = \frac{1}{3} \times 0.7 = 0.233$

-

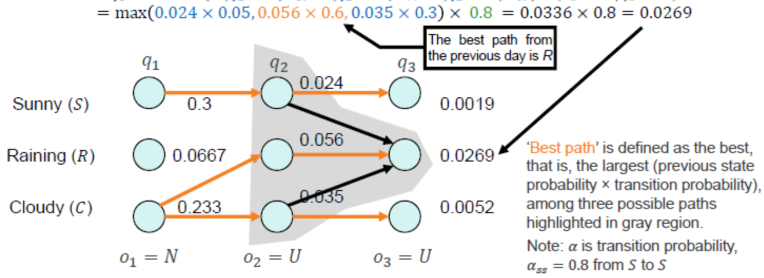
q_2 is S: $\max(P(q_1 = S)\alpha_{ss}, P(q_1 = R)\alpha_{rs}, P(q_1 = C)\alpha_{cs})\, P(o_2 = U|q_2 = S)$
$= \max(0.3 \times 0.8, 0.0667 \times 0.2, 0.233 \times 0.2) \times\ 0.1\ = 0.24 \times 0.1 =\ 0.024$



q_2 is R: $\max(P(q_1 = S)\alpha_{sr}, P(q_1 = R)\alpha_{rr}, P(q_1 = C)\alpha_{cr})\, P(o_2 = U|q_2 = R)$
$= \max(0.3 \times 0.05, 0.0667 \times 0.6, 0.233 \times 0.3) \times\ 0.8\ = 0.233 \times 0.8 =\ 0.056$



q_3 is R: $\max(P(q_2 = S)\alpha_{sr}, P(q_2 = R)\alpha_{rr}, P(q_2 = C)\alpha_{cr})\, P(o_3 = U|q_3 = R)$
$= \max(0.024 \times 0.05, 0.056 \times 0.6, 0.035 \times 0.3) \times\ 0.8\ = 0.0336 \times 0.8 = 0.0269$



The optimal sequence: Cloudy (C), Raining (R), Raining (R).
Recall that the result (in previous Idea 1) is Sunny (S), Raining (R), Raining (R).