Chatbots

SHRIMAI PRABHUMOYE

ALAN W BLACK

SPEECH PROCESSING 11-[468]92

Overview

- Chatbots
- Task Oriented
- Non-Task Oriented Dialog Systems
- Building Dialog Systems
    - Retrieval Based
    - Similarity Metric
    - Generative models

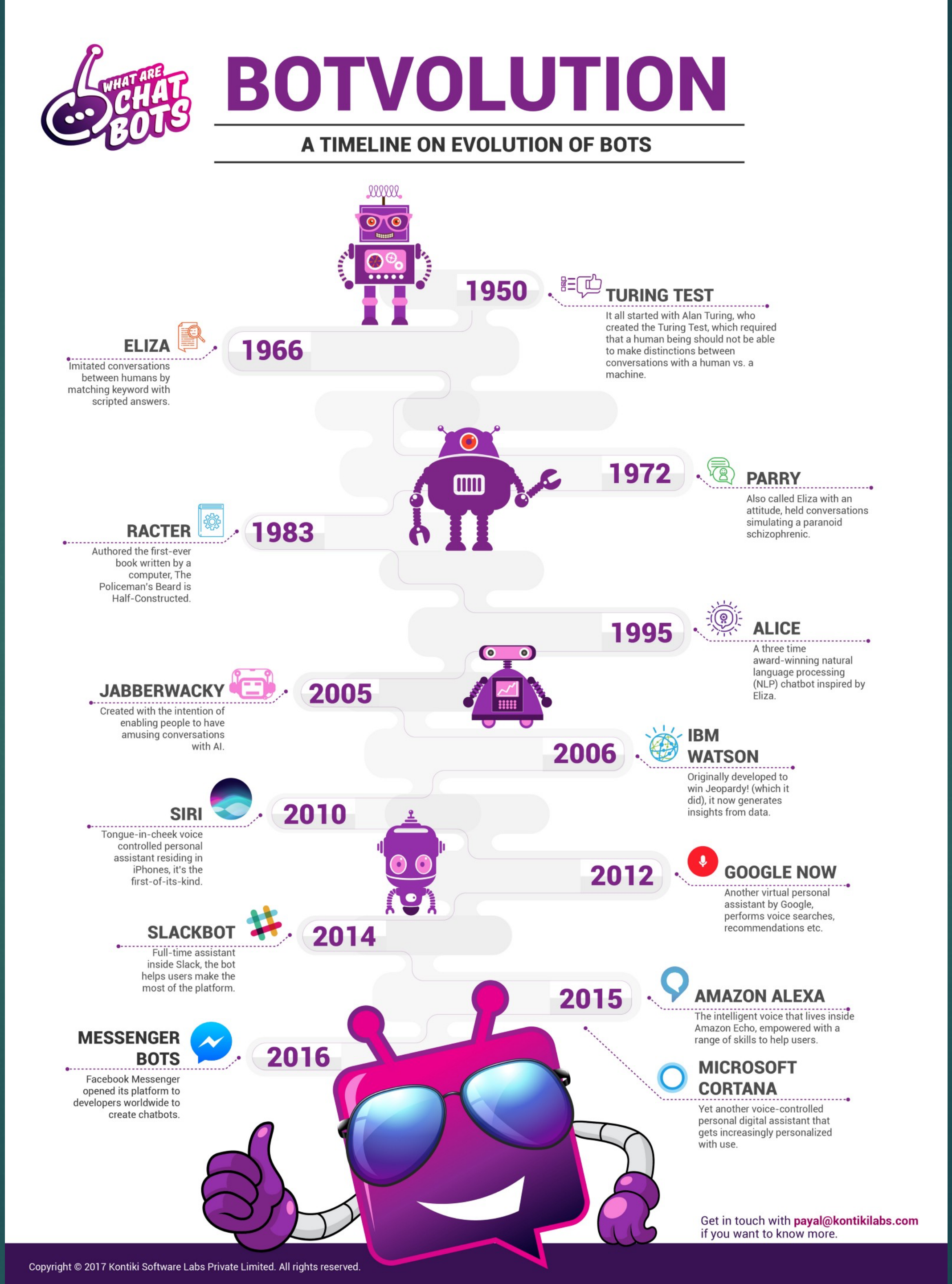
Chatbots

- Designed to simulate how a human would behave as a conversational partner, thereby passing the Turing test.
- Chatbots are used for various practical purposes like customer service, personal assistants or information acquisition.

Chatbots

- Personal Dialog Assistants
  - Siri, Alexa
- Helpline Chatbots
  - hotel booking, airline reservation
- Conversational bots
  - Zo, Tay, Xiaoice, Facebook M

Eliza was based on keyword matching



Parry was Eliza with an attitude

Aspects to think about

Aspects to think about

- Persona
  - voice, age, gender, background
- Domain
- Scenarios it can handle
- Response for other scenarios
- Variation in input and output
- …

Chatbots - Classes

Task Oriented

- Clear and explicit intentions
- The system should have the capability to process the intents.

Non-Task Oriented

- No particular agenda
- “What’s up”, “How are you doing?”

Task Oriented

- Intents: actions that the user wants the system to perform
- Slots: arguments you need to fulfill the intent
- Actions: system performs the task
  - Eg: “Adds a meeting to your calendar”
- Responses: the utterance said by the system
  - Eg: “Meeting has been added”

Task Oriented - Example

- User: I want to add one reminder to switch off the light.
- System: Intent → Add Reminder
- System: Slot → date, time and purpose
- System: When would you like to set the reminder?
- User: In an hour
- System: Slot → 16 Sept at 4.30pm
- System: Action → Set a reminder at 4.30pm on 16 Sept to switch off the light
- System: I have set a reminder to switch off the light in an hour.

Task Oriented - Example

- User: I am getting late to class, so I need the next bus to CMU. I want to reach as fast as possible.
- System: Intent → to go
- System: Slot → date, time, departure_stop, arrival_stop (today, in the next 10 minutes, -, CMU)
- System: Where would you like to leave from?
- User: I live on Whitman and bartlett
- System: departure_stop → whitman and forbes (Wh-Fo)
- System: Action → Find the next bus from Wh-Fo to CMU
- System: The next bus is 61A in 12 minutes from Wh-Fo

Task Oriented

- Evaluation Metric:
  - Task completion success
  - Eg: The system was able to give the correct bus information, the system was able to set an alarm for the right time.
  - Length of dialog
  - Eg: The system should not ask too many questions, too many repetitions etc

Task Oriented - Example

- User: I am getting late to class, so I need the next bus to CMU. I want to reach as fast as possible.
- System: Intent → to go
- System: Slot → date, time, departure_stop, arrival_stop (-,-, -, CMU)
- System: What date would you like me to check the bus for?
- User: Today Sept 12!
- System: When would you like to leave?
- User: In the next 10 min!!

Non Task Oriented

- User1: Hey, what's up?
- User2: Nothing much. What about you?
- User1: Nothing, just the usual hectic life of CMU.
- User2: Yes, its been so long since we caught up. We should may be grab some coffee sometime…
- User1: Ya, that's true. We should definitely meet up sometime. Ok, I need to run for a class, ping me!
- User2: Sure, bye!

Non Task Oriented

- User1: Hey, have you seen the new Kingsman movie?
- User2: No, what's it about?
- User1: It's a science fiction thriller movie. Do you want to go watch it sometime?
- User2: Ya sure, I like sci-fi movies.
- User1: Let's watch it over the weekend
- User2: Ok ☺

Non Task Oriented

- Intents and slots are hard to design
- Can have multiple responses
- Evaluation:
  - Engagement
  - User satisfaction
  - Length of dialog
  - …

Chatbots Architectures

- Rule-based
  - Used very often to build some aspects of personal assistants.
  - Eg: “Add ‘Meet Alan’ in my Calendar”
- Corpus-based
  - Retrieval Techniques
    - Used very often to build helpline chatbots.
    - Examples: “How do I install Ubuntu on my machine?”, “I cannot connect to network. How can I connect to wifi?”
  - Generative Models
    - Used very often to build conversational chatbots.
    - Example: “How are you doing?”, “Can you tell me a secret?”

Eliza: Weizenbaum (1966)

- Men are all alike.
- IN WHAT WAY
- They're always bugging us about something or other.
- CAN YOU THINK OF A SPECIFIC EXAMPLE
- Well, my boyfriend made me come here.
- YOUR BOYFRIEND MADE YOU COME HERE
- He says I'm depressed much of the time.
- I AM SORRY TO HEAR YOU ARE DEPRESSED

(Jurafsky, 2019)

Eliza pattern/transform rules

- (* YOU * ME) [pattern]
- (WHAT MAKES YOU THINK I PAT YOU) [transform]
- Example
  - You hate me
  - WHAT MAKES YOU THINK I HATE YOU

(Jurafsky, 2019)

Eliza Rules

Keyword Pattern Ranked list of transforms

$$\left(K \quad \begin{array}{l} \left((D_1) \ \left(R_{1,1}\right) \ \left(R_{1,2}\right) \ \cdots \ \left(R_{1,m1}\right) \right) \\ \left((D_2) \ \left(R_{2,1}\right) \ \left(R_{2,2}\right) \ \cdots \ \left(R_{2,m2}\right) \right) \\ \qquad\qquad\qquad \vdots \\ \left((D_n) \ \left(R_{n,1}\right) \ \left(R_{n,2}\right) \ \cdots \ \left(R_{n,mn}\right) \right) \end{array} \right)$$

(Jurafsky, 2019)

Eliza Architectures

- Examine each word w in user sentence
  - Return the w with highest keyword rank
- If w exists:
  - Check each rule for w in ranked order
    - Choose first one that matches sentence
    - Apply transform
- If no keyword applies, either
  - Apply the transform for the "NONE" key, or
  - Grab an action off the memory queue

(Jurafsky, 2019)

Eliza Example

- I know everybody laughed at me
  - "I" is a very general keyword:
  - (I   ((I *)   (You say you 2) …)
  - YOU SAY YOU KNOW EVERYBODY LAUGHED AT YOU
  - "Everbody" is much more interesting (someone using universals like everybody/always is probably "referring to some quite specific event or person")
  - WHO IN PARTICULAR ARE YOU THINKING OF?
- Implementation: keywords stored with their rank
  - Everybody 5 (transformation rules)
  - I 0 (transformation rules)

(Jurafsky, 2019)

None

- PLEASE GO ON
- THAT'S VERY INTERESTING
- I SEE

(Jurafsky, 2019)

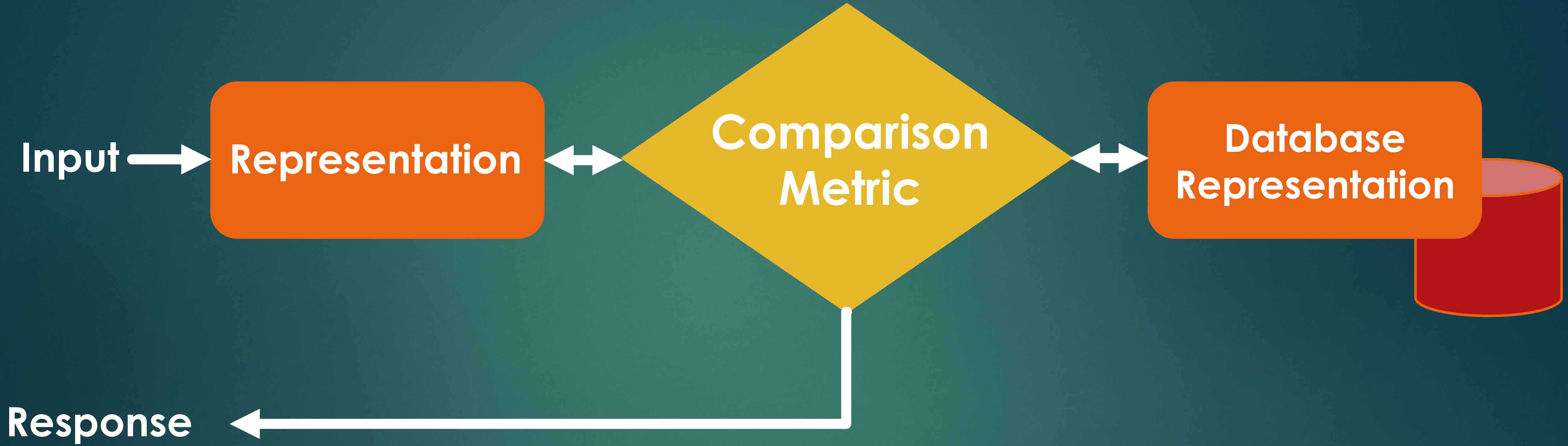
Retrieval Techniques

- Chatbots built using this technique:
  - Cleverbot
  - Xiaoice
- Database of conversations:
  - Human-human chats
  - Human-machine chats
- Find a turn in the database that matches the user’s input, then give the response of that turn from the database.

Retrieval Techniques

- Fixed set of query-response pairs in the database.
- Representation of the query and the database.
- Metric to compare and evaluate the best fitting response.

Retrieval Pipeline



Representation

- Words themselves!
- Term Frequency – Inverse Document Frequency (Tf-Idf)
- N-grams
- Word Vectors

Representation

- Words themselves!
- Term Frequency – Inverse Document Frequency (Tf-Idf)
- N-grams
- Word Vectors

TF-IDF

- Term Frequency (TF):
  - measures how frequently a term occurs in a document.
  - term frequency $tf(t,d)$ of term t in document d is defined as the number of times that t occurs in d.
  - The term frequency is often divided by the document length.

$$tf(t,d) = f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$$

Term Frequency

- Raw term frequency is not what we want:
  - A document with 10 occurrences of the term is more relevant than a document with 1 occurrence of the term.
  - But not 10 times more relevant.
- Relevance does not increase proportionally with term frequency.

Inverse Document Frequency

- Are all words equally informative?
- Rare terms are more informative
  - Example: stop words like *the, a, and, that* etc
- Suppose the input contains a rare term like phagocytosis. (The term is rare in the database)
- A document containing the term phagocytosis is very likely to be relevant to the input
- We want a high weight for rare terms like phagocytosis.

Inverse Document Frequency

- measure of how much information the word provides, that is, whether the term is common or rare across all documents.
- Total number of documents ($\boldsymbol{N}$) divided by the count of the number of documents that contain term $\boldsymbol{t}$

$$idf(t, D) = \log \frac{N}{1 + |\{d \in D : t \in d\}|}$$

TF-IDF Example

- Document (d) → 100 words, term "dog" appears 5 times in d.

$$tf(\text{“}dog\text{”}, d) = \frac{5}{100}$$

- Suppose, D =10 million and "dog" appears in 999 of them

$$idf(\text{“}dog\text{”}, D) = \log \frac{10000000}{1 + 999} = 4$$

- TF-IDF score: 0.05 * 4 = 0.12

TF-IDF Representation

Vocabulary Table

Vocab	Tf-Idf
“the”	0.8
“dog”	0.3
“and”	0.5
“play”	0.6
“UNK”	0.1

Representation of the input

the	dog	and	the	cat	play
0.8	0.3	0.5	0.8	0.1	0.6

TF-IDF Limitations

- Cannot work for synonyms
  - *I find it very common* and *I find it very prosaic* could have very different representations depending on the TF-IDF of common and prosaic
- Does not take context into account
- Doesn’t consider the ordering of words in the query or the document
  - *Bob loves Mary* and *Mary loves Bob* have the same representations!

Representation

- Words themselves!
- Term Frequency – Inverse Document Frequency (Tf-Idf)
- N-grams
- Word Vectors

N-grams

- Unigram: $P(w)$
  - Still does not take context into account
- Bigram: $P(w_1,\ w_2)$
  - $P(\text{“}I\text{”}, \text{“}am\text{”})$ and $P(\text{“}I\text{”}, \text{“}is\text{”})$
  - Takes one word context
- Trigram: $P(w_{1,} w_2, w_3)$
  - Takes two word context
- N-gram: $P(w_{1,} w_2,\ \ldots,\ w_n)$

N-grams

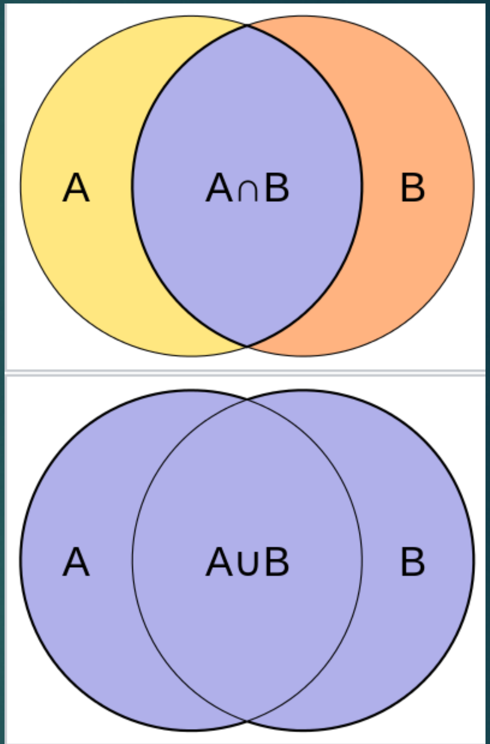
- Takes context into account
- You can set the decide the window size of context

Similarity Metric

- Jaccard Similarity Coefficient
- Cosine Similarity
- Euclidean Distance
- Pearson Similarity
- How it works:
  - A = representation of the input and
  - B = representation of the query in the database.
  - For each query in the database, we calculate similarity score and select the query which has max score.
  - We return the response of this query

Jaccard Similarity

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$



- measures similarity between finite sample sets
- $0 \leq J(A,B) \leq 1$
- Can be used when representations are words themselves.
- Cannot be used with vector representations.

Jaccard Similarity Example

- Input: What food do you like?
- Document 1: I like Indian food.
- Document 2: I hate Spanish food.
- $I = \{What, food, do, you, like, ?\}$
- $D1 = \{I, like, Indian, food, .\}$
- $D2 = \{I, hate, Spanish, food, .\}$
- $J(I, D1) = \dfrac{|\{like, food\}|}{|\{What, food, do, you, like, ?, I, Indian, .\}|} = \dfrac{2}{9} = 0.222$
- $J(I, D2) = \dfrac{|\{food\}|}{|\{What, food, do, you, like, ?, I, hate, Spanish, .\}|} = \dfrac{1}{10} = 0.1$

Cosine Similarity

$$\cos(\theta) = \frac{\boldsymbol{A}\,.\,\boldsymbol{B}}{\left|\left|\boldsymbol{A}\right|\right|_2 \left|\left|\boldsymbol{B}\right|\right|_2}$$

- Measures similarity between two vectors
- Values range between -1 and 1
- -1 is perfectly dissimilar
- 1 is perfectly similar

Cosine Similarity

- A = “I love dogs” = [0.6, 0.4, 0.9]
- B = “You are smart” = [0.5, 0.7, 0.1]
- $\cos(A, A) = \frac{0.6*0.6+0.4*0.4+0.9*0.9}{\sqrt{0.6*0.6+0.4*0.4+0.9*0.9} * \sqrt{0.6*0.6+0.4*0.4+0.9*0.9}} = 1.0$
- $\cos(A, B) = \frac{0.6*0.5+0.4*0.7+0.9*0.1}{\sqrt{0.6*0.6+0.4*0.4+0.9*0.9} * \sqrt{0.5*0.5+0.7*0.7+0.1*0.1}} = 0.6708$
- Note: If you use the TF-IDF representation then cos wont be negative.
- In case vectors are of different lengths then you pad the smaller length vector with 0s

Euclidean Distance

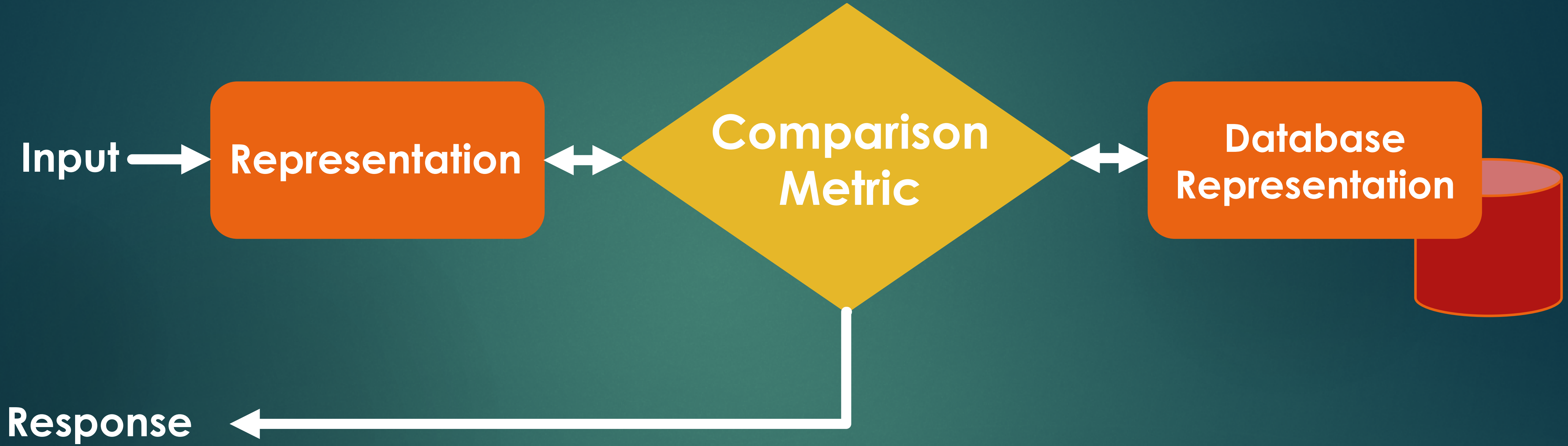
$$d(a,b) = \sqrt{(a_1 - b_1)^2 + \ldots + (a_n - b_n)^2} = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

- Measures similarity between two vectors
- Select the query with least distance from the input

Euclidean Distance

- Euclidean distance is large for different length vectors
- Example: Let a document $d = [d_1; d_1]$
- d is d_1 concatenated to itself
- d and d_1 have the same content
- The Euclidean distance between them can be quite large
- Angle between them is 0, corresponding to maximal similarity.

Retrieval Pipeline



Example

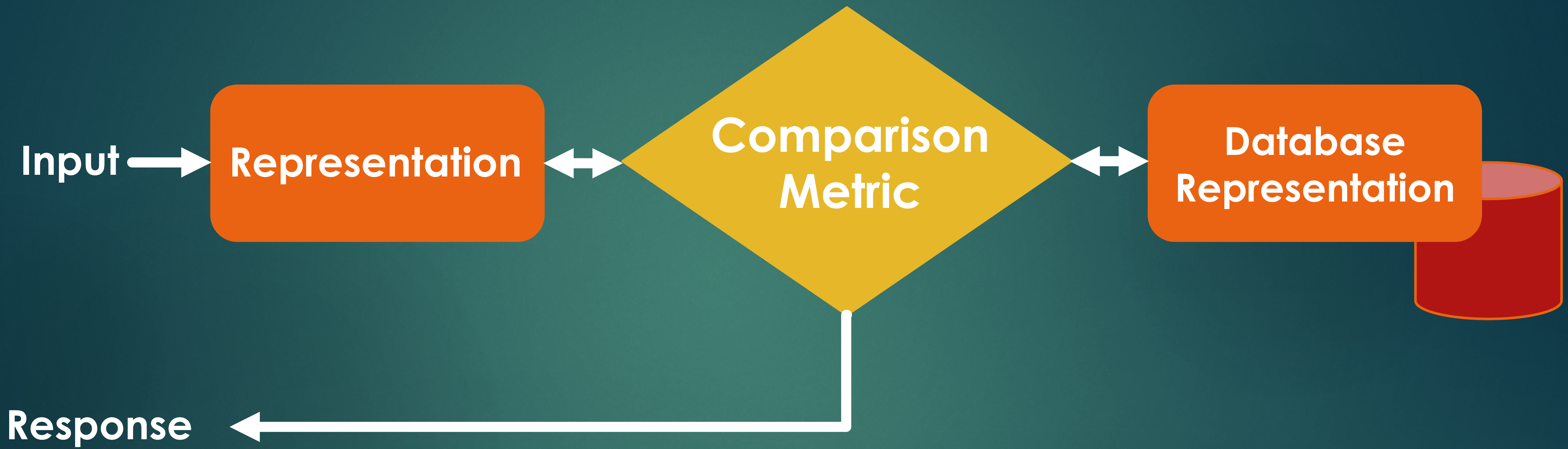
Database

Input

“How do I connect to WiFi”

“How can I connect to WiFi”	“Go to Settings → Wifi. Select …”
“How do I install Ubuntu 16.04”	“Download Ubuntu image …”
“How can I install Java”	“Download the jdk …”
“Which NVIDIA driver do I need for GTX 1080 Ti”	“sudo apt install nvidia-381”

Retrieval Pipeline



Database Representation

Database

"How can I connect to WiFi"
"Go to Settings → Wifi. Select …"

"How do I install Ubuntu 16.04"
"Download Ubuntu image …"

"How can I install Java"
"Download the jdk …"

"Which NVIDIA driver do I need for GTX 1080 Ti"
"sudo apt install nvidia-381"

Total Query Words = 22

Vocab	Tf-Idf	Vocab	Tf-Idf
How	3/22*log(1/2)	Java	
can		Which	
I		NVIDIA	
connect		driver	
to		need	
Wifi		for	
do		GTX	
install		1080	
Ubuntu		Ti	
16.04		UNK	

Database Representation

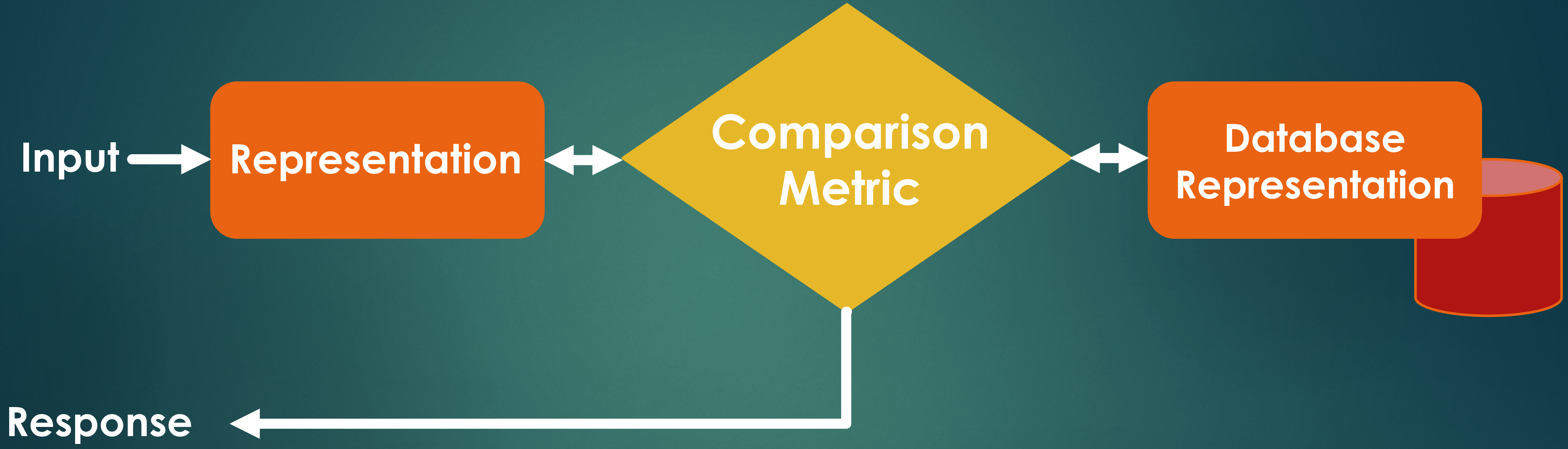
How	can	I	connect	to	WiFi
0.6	0.3	0.4	0.4	0.4	0.1

How	do	I	install	Ubuntu	16.04
0.6	0.2	0.4	0.35	0.1	0.05

How	can	I	install	Java
0.6	0.3	0.4	0.35	0.15

Which	NVIDIA	driver	do	I	need	for	GTX	1080	Ti
0.2	0.1	0.06	0.2	0.4	0.5	0.3	0.01	0.02	0.04

Retrieval Pipeline

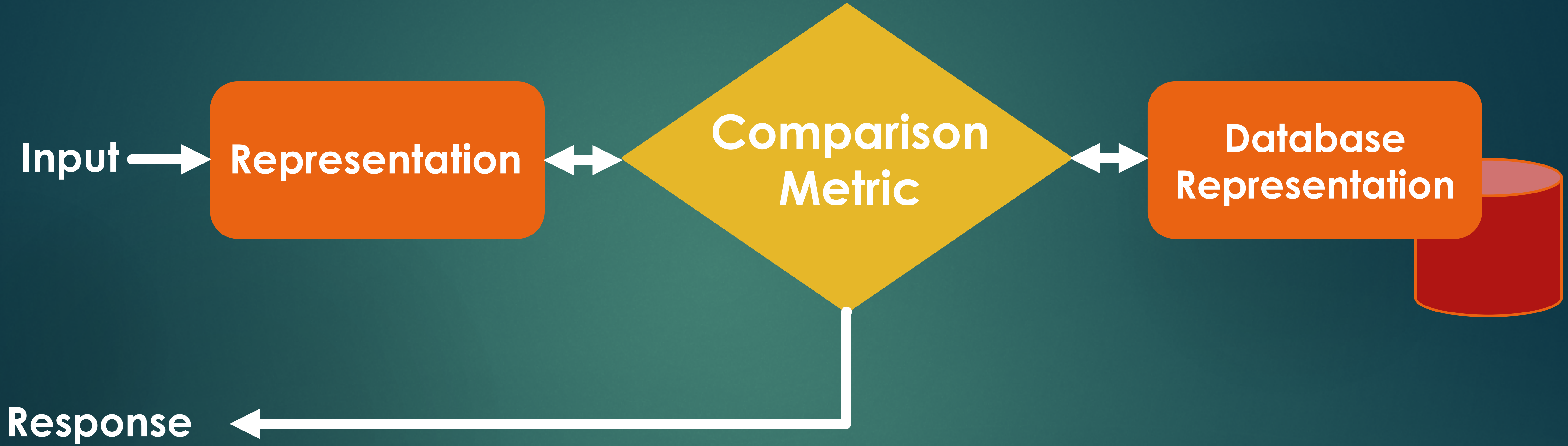


Input Representation

- Use the TF-IDF counts calculated over the database

How	do	I	connect	to	WiFi
0.6	0.2	0.4	0.4	0.4	0.1

Retrieval Pipeline



Compare

- Lets compare using cosine similarity
- $I = [0.6, 0.2, 0.4, 0.4, 0.4, 0.1]$
- $D1 = [0.6, 0.3, 0.4, 0.4, 0.4, 0.1]$
- $D2 = [0.6, 0.2, 0.4, 0.35, 0.1, 0.05]$
- $\cos(I, D1) = 0.9949069$
- $\cos(I, D2) = 0.9472593$
- $\cos(I, D1) > \cos(I, D2)$
- Hence, we select query $D1$ and return its response from the database "Go to Settings → Wifi. Select … "

How	do	I	connect	to	WiFi
0.6	0.2	0.4	0.4	0.4	0.1

How	can	I	connect	to	WiFi
0.6	0.3	0.4	0.4	0.4	0.1

How	do	I	install	Ubuntu	16.04
0.6	0.2	0.4	0.35	0.1	0.05

Advantages of Retrieval Systems

- No grammatical or meaning less errors as we store the answers
- Works very well for domain specific problems
  - Eg: chatbot for customer care for a business

Limitations of Retrieval Systems

- We have a constrained set of responses.
- No variance in the response.
- Cannot handle novel queries.

Summary

- Task Oriented
  - Intents, Slots, Responses. Evaluation by task completion.
- Non-Task oriented
  - Intents and evaluation are hard to define.
- Retrieval Techniques
  - TF-IDF representation and cosine similarity
- Limitations of Retrieval Techniques

Generative Models

- Next Class!