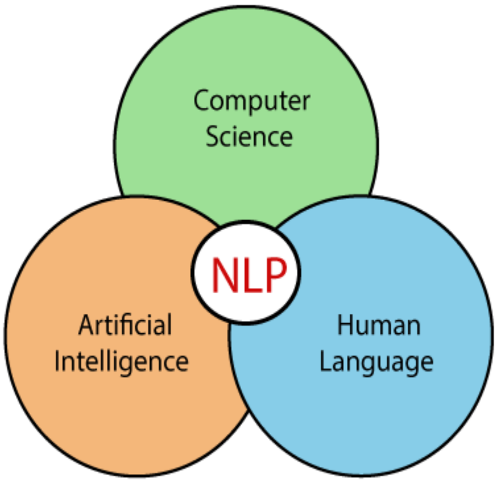
UNIT – 1 INTRODUCTION

Introduction to NLP, Machine Learning and NLP, Biology of Speech Processing; Place and Mannernof Articulation, Word Boundary Detection, Arg-Max Computation, Lexical Knowledge Networks.

1.1 INTRODUCTION TO NLP:

NLP stands for **Natural Language Processing**, which is a part of **Computer Science, Human language,** and **Artificial Intelligence**. It is the technology that is used by machines to understand, analyse, manipulate, and interpret human's languages. The goal of NLP is to enable computers to understand and interpret human language in a way that is similar to how humans process language. It helps developers to organize knowledge for performing tasks such as **translation, automatic summarization, Named Entity Recognition (NER), speech recognition, relationship extraction,** and **topic segmentation**.



1.1.1 Components of NLP

There are the following two components of NLP -

1. Natural Language Understanding (NLU)

Natural Language Understanding (NLU) helps the machine to understand and analyse human language by extracting the metadata from content such as concepts, entities, keywords, emotion, relations, and semantic roles.

NLU mainly used in Business applications to understand the customer's problem in both spoken and written language.

NLU involves the following tasks -

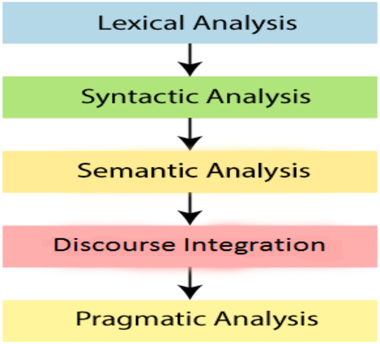
- It is used to map the given input into useful representation.
- It is used to analyze different aspects of the language.

2. Natural Language Generation (NLG)

Natural Language Generation (NLG) acts as a translator that converts the computerized data into natural language representation. It mainly involves Text planning, Sentence planning, and Text Realization.

1.1.2 Phases of NLP

There are the following five phases of NLP:



1. Lexical Analysis and Morphological

The first phase of NLP is the Lexical Analysis. This phase scans the source code as a stream of characters and converts it into meaningful lexemes. It divides the whole text into paragraphs, sentences, and words.

2. Syntactic Analysis (Parsing)

Syntactic Analysis is used to check grammar, word arrangements, and shows the relationship among the words.

Example: Agra goes to the Poonam
In the real world, Agra goes to the Poonam, does not make any sense, so this sentence is rejected by the Syntactic analyzer.

3. Semantic Analysis

Semantic analysis is concerned with the meaning representation. It mainly focuses on the literal meaning of words, phrases, and sentences.

4. Discourse Integration

Discourse Integration depends upon the sentences that proceeds it and also invokes the meaning of the sentences that follow it.

5. Pragmatic Analysis

Pragmatic is the fifth and last phase of NLP. It helps you to discover the intended effect by applying a set of rules that characterize cooperative dialogues.

For Example: "Open the door" is interpreted as a request instead of an order.

1.1.3 How to build an NLP pipeline

There are the following steps to build an NLP pipeline –

Step1: Sentence Segmentation
Sentence Segment is the first step for building the NLP pipeline. It breaks the paragraph into separate sentences.

Example: Consider the following paragraph -

Independence Day is one of the important festivals for every Indian citizen. It is celebrated on the 15th of August each year ever since India got independence from the British rule. The day celebrates independence in the true sense.

Sentence Segment produces the following result:

1. "Independence Day is one of the important festivals for every Indian citizen."
2. "It is celebrated on the 15th of August each year ever since India got independence from the British rule."
3. "This day celebrates independence in the true sense."

Step2: Word Tokenization

Word Tokenizer is used to break the sentence into separate words or tokens.

Example:

JavaTpoint offers Corporate Training, Summer Training, Online Training, and Winter Training.

Word Tokenizer generates the following result:

"JavaTpoint", "offers", "Corporate", "Training", "Summer", "Training", "Online", "Training", "and", "Winter", "Training", "."

Step3: Stemming

Stemming is used to normalize words into its base form or root form. For example, celebrates, celebrated and celebrating, all these words are originated with a single root word "celebrate." The big problem with stemming is that sometimes it produces the root word which may not have any meaning.

For Example, intelligence, intelligent, and intelligently, all these words are originated with a single root word "intelligen." In English, the word "intelligen" do not have any meaning.

Step 4: Lemmatization

Lemmatization is quite similar to the Stamming. It is used to group different inflected forms of the word, called Lemma. The main difference between Stemming and lemmatization is that it produces the root word, which has a meaning.

For example: In lemmatization, the words intelligence, intelligent, and intelligently has a root word intelligent, which has a meaning.

Step 5: Identifying Stop Words

In English, there are a lot of words that appear very frequently like "is", "and", "the", and "a". NLP pipelines will flag these words as stop words. **Stop words** might be filtered out before doing any statistical analysis.

Example: He **is a** good boy.

Step 6: Dependency Parsing

Dependency Parsing is used to find that how all the words in the sentence are related to each other.

Step 7: POS tags

POS stands for parts of speech, which includes Noun, verb, adverb, and Adjective. It indicates that how a word functions with its meaning as well as grammatically within the sentences. A word has one or more parts of speech based on the context in which it is used.

Example: "Google" something on the Internet.

In the above example, Google is used as a verb, although it is a proper noun.

Step 8: Named Entity Recognition (NER)

Named Entity Recognition (NER) is the process of detecting the named entity such as person name, movie name, organization name, or location.

Example: Steve Jobs introduced iPhone at the Macworld Conference in San Francisco, California.

Step 9: Chunking

Chunking is used to collect the individual piece of information and grouping them into bigger pieces of sentences.

1.1.4 How does natural language processing work?

NLP enables computers to understand natural language as humans do. Whether the language is spoken or written, natural language processing uses artificial intelligence to take real-world input, process it, and make sense of it in a way a computer can understand. Just as humans have different sensors -- such as ears to hear and eyes to see -- computers have programs to read and microphones to collect audio. And just as humans have a brain to process that input, computers have a program to process their respective inputs. At some point in processing, the input is converted to code that the computer can understand. There are two main phases to natural language processing: data preprocessing and algorithm development.

Data preprocessing involves preparing and "cleaning" text data for machines to be able to analyse it. preprocessing puts data in workable form and highlights features in the text that an algorithm can work with.

Once the data has been preprocessed, an algorithm is developed to process it. There are many different natural language processing algorithms, but two main types are commonly used:

- Rules-based system. This system uses carefully designed linguistic rules. This approach was used early on in the development of natural language processing, and is still used.
- Machine learning-based system. Machine learning algorithms use statistical methods. They learn to perform tasks based on training data they are fed, and adjust their methods as more data is processed. Using a combination of machine learning, deep learning and neural networks, natural language processing algorithms hone their own rules through repeated processing and learning.

1.1.5 Why NLP is difficult?

NLP is difficult because Ambiguity and Uncertainty exist in the language.

Ambiguity

There are the following three ambiguity -

- **Lexical Ambiguity**

  Lexical Ambiguity exists in the presence of two or more possible meanings of the sentence within a single word.

  Example:

  Manya is looking for a **match**.

  In the above example, the word match refers to that either Manya is looking for a partner or Manya is looking for a match. (Cricket or other match)

- **Syntactic Ambiguity**

  Syntactic Ambiguity exists in the presence of two or more possible meanings within the sentence.

  Example:

  I saw the girl with the binocular.

  In the above example, did I have the binoculars? Or did the girl have the binoculars?

- **Referential Ambiguity**

Referential Ambiguity exists when you are referring to something using the pronoun.

Example: Kiran went to Sunita. She said, "I am hungry."

In the above sentence, you do not know that who is hungry, either Kiran or Sunita.

1.1.6 NLP techniques are used in a wide range of applications, including:

- Speech recognition and transcription: NLP techniques are used to convert speech to text, which is useful for tasks such as dictation and voice-controlled assistants.
- Language translation: NLP techniques are used to translate text from one language to another, which is useful for tasks such as global communication and e-commerce.
- Text summarization: NLP techniques are used to summarize long text documents into shorter versions, which is useful for tasks such as news summarization and document indexing.
- Sentiment analysis: NLP techniques are used to determine the sentiment or emotion expressed in text, which is useful for tasks such as customer feedback analysis and social media monitoring.
- Question answering: NLP techniques are used to answer questions asked in natural language, which is useful for tasks such as chatbots and virtual assistants.
- NLP is a rapidly growing field and it is being used in many industries such as healthcare, education, e-commerce, and customer service. NLP is also used to improve the performance of natural language-based systems like chatbot, virtual assistants, recommendation systems, and more. With the advancement in NLP, it has become possible for computers to understand and process human languages in a way that can be used for various applications such as speech recognition, language translation, question answering, and more.

1.1.7 Advantages of Natural Language Processing:

1. **Improves human-computer interaction**: NLP enables computers to understand and respond to human languages, which improves the overall user experience and makes it easier for people to interact with computers.
2. **Automates repetitive tasks**: NLP techniques can be used to automate repetitive tasks, such as text summarization, sentiment analysis, and language translation, which can save time and increase efficiency.
3. **Enables new applications**: NLP enables the development of new applications, such as virtual assistants, chatbots, and question answering systems, that can improve customer service, provide information, and more.
4. **Improves decision-making**: NLP techniques can be used to extract insights from large amounts of unstructured data, such as social media posts and customer feedback, which can improve decision-making in various industries.
5. **Improves accessibility**: NLP can be used to make technology more accessible, such as by providing text-to-speech and speech-to-text capabilities for people with disabilities.

1.1.8 Disadvantages of Natural Language Processing:

1. **Limited understanding of context**: NLP systems have a limited understanding of context, which can lead to misinterpretations or errors in the output.

2. **Requires large amounts of data**: NLP systems require large amounts of data to train and improve their performance, which can be expensive and time-consuming to collect.
3. **Limited ability to understand idioms and sarcasm**: NLP systems have a limited ability to understand idioms, sarcasm, and other forms of figurative language, which can lead to misinterpretations or errors in the output.
4. **Limited ability to understand emotions**: NLP systems have a limited ability to understand emotions and tone of voice, which can lead to misinterpretations or errors in the output.

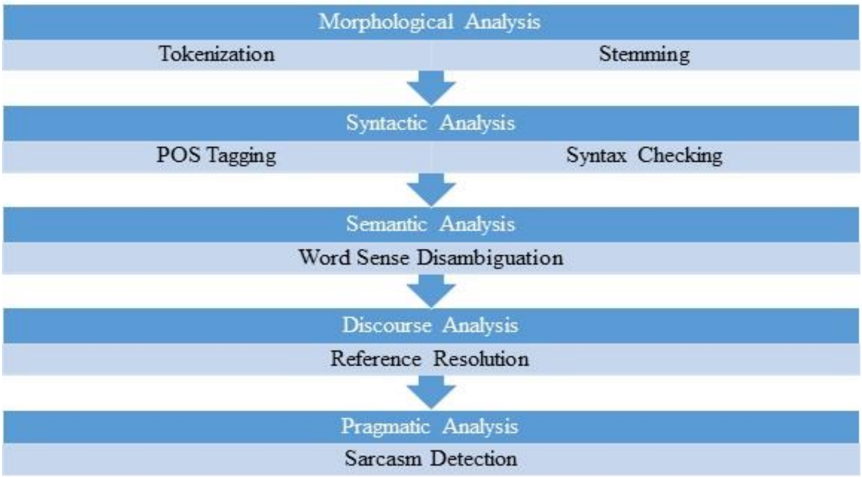
1.2 Machine Learning and NLP

- Machine Learning and Natural Language Processing are important subfields of Artificial Intelligence. Machine Learning and Natural Language Processing play a very important part in making an artificial agent into an artificial 'intelligent' agent. An Artificially Intelligent system can accept better information from the environment and can act on the environment in a user-friendly manner because of the advancement in Natural Language Processing.
- Artificially Intelligent System can process the received information and perform better predictions for its actions because of the adoption of Machine Learning techniques.
- Machine Learning gives the system the ability to learn from past experiences and examples. General algorithms perform a fixed set of executions according to what it has been programmed to do so and they do not possess the ability to solve unknown problems. And, in the real world, most of the problems faced contain many unknown variables which makes the traditional algorithms very less effective. This is where machine learning comes to the fore. With the help of past examples, a machine learning algorithm is far better equipped to handle such unknown problems.
- Some of the classic examples given include spam mail detection. To detect and classify if a mail is a legitimate one or spam includes many unknowns. There are many ways in which spam filters can be evaded. For a traditional algorithm to work, every feature and variable has to be hardcoded, which is extremely difficult, if at all possible. Whereas, a machine learning algorithm will be able to work in such an environment because of its ability to learn and form a general rule.
- Deep Learning is a specialization of machine learning algorithms, the Artificial Neural Network. In recent times it has been observed that deep learning techniques have been widely adopted and have produced good results as well. The flexibility provided by the deep learning techniques in deciding upon the architecture is one of the important reasons for the success of these techniques. Deep learning techniques have been at the forefront of machine learning techniques used for research in natural language processing.
- Natural Language Processing, on the other hand, is the ability of a system to understand and process human languages. A computer system only understands the language of 0's and 1's, it does not understand human languages like English or Hindi. Natural Language Processing gave the computing system the ability to understand English or the Hindi language.
- Natural Language Processing has seen large-scale adaptation in recent times because of the level of user-friendliness it brings to the table. From choosing your choice of music to controlling your electronic appliances like Air conditioners, and ovens, in fact even the ceiling fans and light bulbs, everything and anything can now be done using your voice, thus making these electronic items smart…!!. This is all possible because of Natural Language Processing.

- Even as NLP has made it easier for the users to interact with the complex electronics, on the other side there is a lot of processing happening behind the scenes which makes this interaction possible. Machine learning has played a very important role in this processing of the language.
- Apart from playing a role in the proper processing of natural language Machine Learning has played a very constructive role in important applications of natural language processing. Important NLP applications like Sentiment Analysis, Chatbot Systems, Question Answering Systems, Information Retrieval Systems, Machine Translation, and Email Classification, among others have all included machine learning techniques for better working.

1.2.1 Role of Machine Learning in Natural Language Processing

- Processing of natural language so that the machine can understand the natural language involves many steps. These steps include Morphological Analysis, Syntactic Analysis, Semantic Analysis, Discourse Analysis, and Pragmatic Analysis, generally, these analysis tasks are applied serially. Machine Learning acts as important value addition in all these processes in some form or the other.



1. Morphological Analysis:

The data received by the computing system is in the form of 0s and 1s. These 0s and 1s can be converted into alphabets using the ASCII code. So, it can be said that a machine receives a bunch of characters when a sentence or a paragraph has been provided to it. At the level of morphological analysis, the first task is to identify the words and the sentences. This identification is called tokenization. Many Different Machine Learning and Deep Learning algorithms have been employed for tokenization including Support Vector Machine and Recurrent Neural Network.

Once the tokenization is complete the machine has with it a bunch of words and sentences. Most of the sentences which are formed contain affixes. These affixes complicate the matter for the machines as, having a word meaning dictionary containing all the words with all its possible affixes is almost impossible. So, the next task that the morphological analysis level is removing these affixes. These affixes can be removed either using stemming or lemmatization. Machine Learning algorithms like the random forest and decision tree have been quite successful in performing the task of stemming.

2. Syntactic Analysis

The next task in natural language processing is to check whether the given sentence follows the grammar rule of a language. To do this the words are first tagged with their part of speech. This helps the syntactic parsers in checking the grammar rules. Machine learning and Deep learning algorithms like the random forest and the recurrent neural network has been successfully used implemented for this task. Machine learning algorithms like K- nearest neighbour have been used for implementing syntactic parsers as well.

3. Semantic Analysis

At this level, the word meanings are identified using word-meaning dictionaries. The problem encountered here is, the same word might have different meanings according to the context of the sentence. For example, the word 'Bank' might mean a Blood Bank or a Financial Bank, or even a River Bank / Shore, this creates ambiguity. So, removing this ambiguity is one of the important tasks at this level of natural language processing called Word Sense Disambiguation.

Word sense disambiguation is one of the classical classification problems which have been researched with different levels of success. Machine learning like the random forest, gradient boosting and decision trees have been successfully employed. But, in recent times it is the deep learning algorithms like the recurrent neural network, long short term memory based recurrent neural network, gated recurrent unit based recurrent neural network and convolution neural network have been researched and have produced very good results.

4. Discourse Analysis

There instances where pronouns are used or certain subjects/objects are referred to, which are outside of the current preview of the analysis. In such cases, the semantic analysis will not be able to give proper meaning to the sentence. This is another classical problem of reference resolution which has been tackled by machine learning and deep learning algorithms.

5. Pragmatic Analysis

Many a time sentences convey a deeper meaning than what the words can describe. That is, the machine has to discard the word meaning understood after semantic analysis and capture the intended or the implied meaning. It is easier said than done. For many years now this is of natural language process has intrigued researchers. One of the classic examples of pragmatic analysis is sarcasm detection.

Many different machine learning and deep learning algorithms have been employed with varied success for performing sarcasm detection for performing pragmatic analysis in general.

Role of Machine Learning in the applications of Natural Language processing

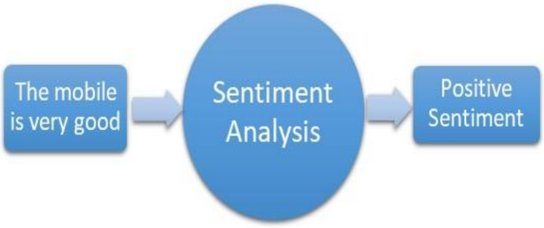
As with the processing task of the natural language, machine learning and deep learning algorithms have played a very important role in almost all of the applications of natural language processing. Almost all the deep learning techniques including, Deep Neural Network, Autoencoders, Restricted Boltzmann Machine, Recurrent Neural Network, and Convolution Neural Network have been experimented with to get good accuracy in the different applications of Natural Language Processing.

1.2.3 Applications

1. Sentiment Analysis

Sentiment Analysis strives to analyse the user opinions or sentiments on a certain product. Sentiment analysis has become a very important part of Customer Relationship

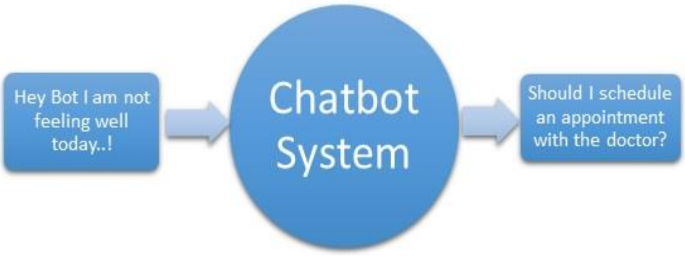
Management. Even a single negative opinion can be disastrous for the product. Recent times have seen greater use of deep learning techniques for sentiment analysis. An interesting fact to note here is that new deep learning techniques have been quipped especially for analysis of sentiments that is the level of research that is being conducted for sentiment analysis using deep learning.



2. Chatbot Systems

Chatbot systems are conversational agents or dialog systems that try to engage the user in a conversation. This conversation can be through voice or text. Personal assistants like Amazon's Alexa and Google Assistant have popularised the chatbot systems and have also showcased the level of ease through which user interaction can be carried out.As easy as it may sound, the development of a true chatbot system that can replace a human agent is an extremely difficult task. Which requires Natural Language Understanding and also Natural Language Generation.

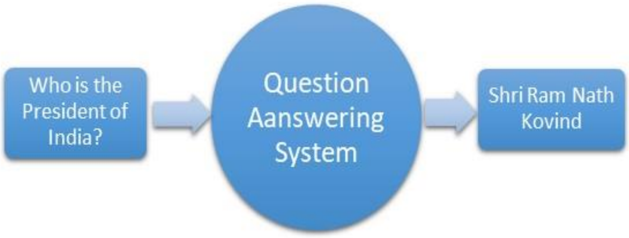
Recent frameworks like Google's DialogFlow, IBM's Watson AI, and Amazon's Alexa AI provide an easy way of developing a chatbot system. And, all these frameworks employ complex and proprietary deep learning architectures.


Hey Bot I am not feeling well today..!
Chatbot System
Should I schedule an appointment with the doctor?

3. Question Answering Systems

As the name suggests, a question answering system is a system that tries to answer user's questions. Recent times have seen the thin line separating a dialog system and a question answering system getting blurred and most of the time a chatbot system performs the question answering task and it is true the other way round as well. So, the research works which pledge to develop a chatbot system will, in all probability, be developing a question answering system within it as well.

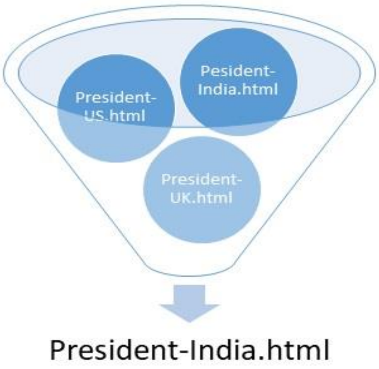
A question answering system has three important components, Question Processing, Information Retrieval, and Answer Processing. Machine Learning and Deep Learning techniques have played a crucial role in all these three components. Especially, Question Processing has attracted quite a few research. The idea here is that understanding the question is extremely important for better answer retrieval. The question processing task is taken as a classification problem and many research works have experimented with deep learning techniques for better question classification.



4. Information Retrieval Systems

Information Retrieval is another important application of Natural Language Processing that tries to retrieve relevant information. Information retrieval systems act as the backbone of the systems like the chatbot systems and question answering systems.

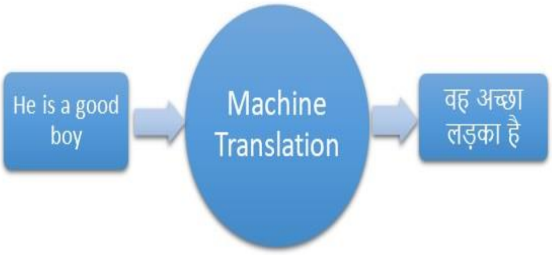
The most basic way of retrieving any information is using the frequency method where the frequency of keywords determines if a particular data is retrieved or not. But, smart systems process the required query as well as the present large data to retrieve only the relevant information. This process is carried out using deep learning techniques.



5. Machine Translation

A machine translation system is striving to translate a text from one language to another with minimum or no human intervention. Applications like Google Translate are one of the best examples of the machine translation system.

Have a translation system that translates word to word is not enough as the construction of a sentence might vary from one language to another. For example, English follows the Subject-Verb-Object format whereas Hindi follows Subject -Object-Verb form for sentence construction. Apart from this, there are many different rules which need to be followed. All these things make the task of machine translation difficult.



The Recurrent Neural Network Deep learning technique along with its variants, Long Short Term Memory and Gated Recurrent Unit, with their Bi-directional forms, have been extensively experimented with for better machine translation. The reason for this is the ability of these neural networks in holding on to the contextual information, which is very crucial in proper translation. Even, Convolution Neural networks have experimented with varied success.

So, it can be observed that Machine Learning and Deep Learning techniques are being extensively researched for their employment in the field of Natural Language Processing.

it can be seen that these learning techniques are playing an important role in almost all of the processing of natural language tasks as well as in almost all the applications of natural language processing.

All the different processing of natural language tasks and the different applications of natural language processing are different fields of research by themselves. And currently, in all these fields of research Machine Learning and Deep Learning techniques are being researched extensively with an exceeding level of success. In conclusion, it can be said that Machine Learning and Deep Learning techniques have been playing a very positive role in Natural Language Processing and its applications.

1.3 Biology of Speech Processing

- Speech processing and NLP allow intelligent devices, such as smartphones, to interact with users via verbal language.

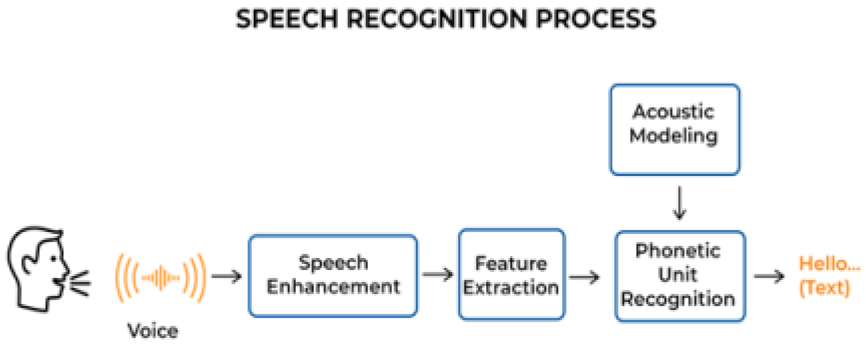
- Speech processing is the process by which speech signals are interpreted, understood, and acted upon. It specifically refers to the processing of human speech by computerized systems, as in **voice recognition** software or voice-to-text programs.

- Speech processing is important to many fields for both theoretical and practical uses, ranging from voice activation and control in phones to development of functional artificial intelligence in computer science.

- Interpretation and production of coherent speech are both important in the processing of speech. The application needs of speech processing are very diverse.

- Speech recognition is one of the most important aspects of speech processing because the overall aim of processing speech is to comprehend and to act on **spoken language**.

- Speech processing is a complex and highly coordinated process that involves multiple areas of the brain working together. The biology of speech processing can be divided into two main categories: the production of speech and the perception of speech.

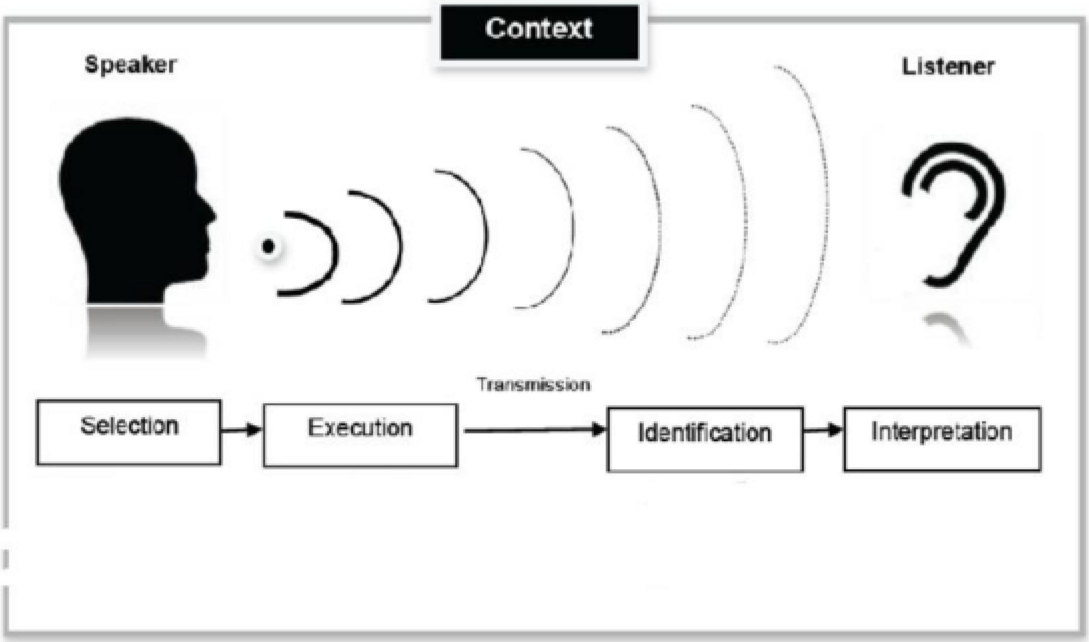


- Production of Speech: The production of speech involves the coordination of many different muscles and structures, including the tongue, lips, jaw, vocal cords, and respiratory system. The brain sends signals to these muscles to control their movements and create the sounds of speech. The motor cortex, located in the frontal lobe of the brain, plays a key role in controlling these movements.

- The production of speech involves the coordination of many different muscles and structures in the vocal tract, including the lips, tongue, jaw, vocal cords, and respiratory system. The process can be divided into three main stages: air pressure, sound generation, and speech articulation. 1. Air pressure: The first stage of speech production involves the respiratory system. The diaphragm and intercostal muscles work together to create airflow, which is then regulated by the larynx and vocal cords. As air is expelled from the lungs, it passes through the larynx and the vocal cords, causing them to vibrate. 2. Sound generation: The vibration of the vocal cords produces a fundamental frequency, which is then modified by the resonant properties of the vocal tract. This modification creates different sounds that are then further modified by the movement of the tongue, lips, and jaw. 3. Speech articulation: The final stage of speech production involves the movement of the tongue, lips, and jaw to produce specific sounds. The tongue is a particularly important organ in speech production, as it can be positioned in different ways to produce a wide range of sounds. The lips and jaw also play a role in modifying the sounds produced by the vocal cords. The production of speech is a complex and highly coordinated process that involves the integration of multiple muscles and structures in the vocal tract. The process is controlled by the motor cortex in the frontal lobe of the brain, which sends signals to the various muscles involved in speech production. Perception of Speech: The perception of speech involves the brain's ability to process and interpret the sounds of speech. This process begins in the auditory system, which consists of the ear and various brain regions responsible for processing auditory information. The auditory cortex, located in the temporal lobe of the brain, plays a key role in processing and interpreting speech sounds.
- Additionally, other brain regions are involved in the perception of speech, including the Broca's area and Wernicke's area. These areas are located in the left hemisphere of the brain and are responsible for language production and comprehension, respectively. The biology of speech processing is complex and involves the coordination of many different brain regions and processes. Research in this area continues to shed light on the intricate workings of the brain and how they relate to speech and language processing. The perception of speech involves the brain's ability to interpret the sounds of speech. This process begins with the auditory system, which consists of the ear and various brain regions responsible for processing auditory information.
- The perception of speech can be divided into two main stages: the detection of sound and the interpretation of meaning. 1. Detection of sound: The first stage of speech perception involves the detection of sound by the ear. Sound waves travel through the ear canal and cause the eardrum to vibrate, which then causes the bones in the middle ear to vibrate. This vibration is then transmitted to the cochlea in the inner ear, which contains hair cells that convert the vibrations into electrical signals that are sent to the brain. 2. Interpretation of meaning: The second stage of speech perception involves the brain's ability to interpret the sounds of speech and assign meaning to them. This process begins in the auditory cortex, located in the temporal lobe of the brain, which processes and interprets auditory information. The auditory cortex is able to distinguish between different sounds and can identify specific phonemes, the smallest units of sound in language.
- Other brain regions are also involved in the interpretation of speech, including the Broca's area and Wernicke's area. Broca's area, located in the left frontal lobe, is responsible for language production, while Wernicke's area, located in the left temporal lobe, is

responsible for language comprehension. These areas work together to help the brain interpret the sounds of speech and assign meaning to them.

- The perception of speech is a complex process that involves the integration of multiple brain regions and processes. Research in this area continues to shed light on how the brain is able to process and interpret language, which has important implications for the diagnosis and treatment of speech and language disorders.



- The most well-known example of speech recognition technology on a mobile device is Apple's voice-recognition service, *Siri*. Siri uses onboard microphones to detect speech (e.g., commands, questions, or dictations) and Automatic Speech Recognition (ASR) to transcribe it into text. The software then translates the transcribed text into "parsed text" and then evaluates it locally on the device. If the request cannot be handled on the device, Siri communicates with servers in the Cloud (web services) to help process the request. Once the command is executed, Siri presents the information and/or provides a verbal response back to the user. Siri also makes use of ML methods to adapt to the user's individual language usage and individual searches and returns personalized results.

1.4 Manner of Articulation

Manner of articulation is nothing but comparative study of position of articulators and also nature of obstruction in the air passage. In this the study of how exactly the consonant sound is produce is considered.

Manner of articulation	How it is created
Plosive	A short, quick release of air after closed stricture.
Fricative	Close stricture that creates friction when air is released.
Affricate	Start with producing a plosive and blending immediately into a fricative.

Nasal	Air is released through the nasal passages.
Liquids	Air is released from the sides of tongue.
Glides	Mouth is move continuously from a articulation to a vowel sound

Different ways of Manner of articulation :

1. **Stop or Plosive -**In this we built pressure of air & then release it. This is sudden burst of air.

   Eg.
   /p/ /b/ /t/ /d/ /k/ /g/
   Initial sound : pig, tall, kick, bag, dad, girl

2. **Fracative-** In this we stop part of air so it cant come through everywhere. It is a stream of air.
   Eg.
   / v /: vat, van / ð /: then, them / z /: zip, zoom / ʒ /: casual, treasure / f /: fat, far

   / s /: site, cycle / h /: help, high / ʃ /: ship, she / θ /: think, north

   Initial sound : four, van ,sun , zip, hello, ship, jaw, thin
   Final sound : half, have, pass, has, wash, rough, bathe,bath

3. **Affricate** : stop + fracatives = Affricate
   Affricates are also known as semi-plosives and are created by combining a plosive and a fricative consonant. There are two affricatives: / t ʃ / and / dʒ /.
   Eg.
   / tʃ /: chair, choose / dʒ /: jump, jet

4. **Nasal :** Nasal consonants, also known nasal stops, are made by blocking the airflow from the mouth, so it comes out of the nose instead. In nasal vowels, by contrast, the sound is generated by lowering the soft palate to allow the airflow out of both mouth and nose.

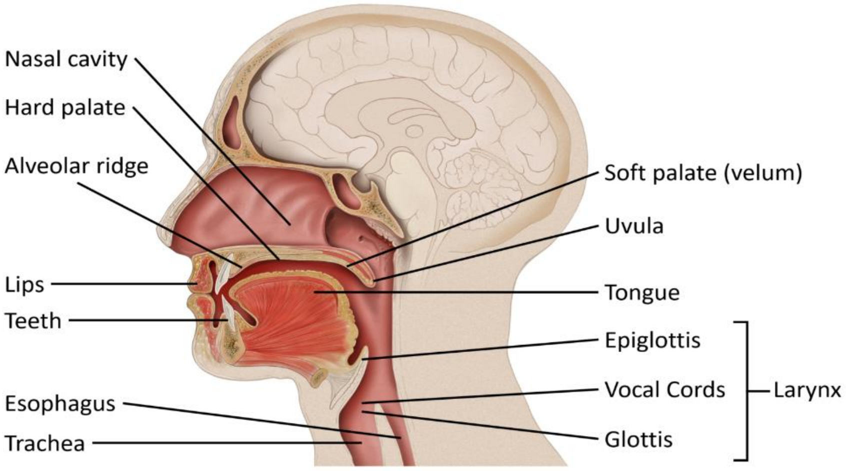
   There are three nasal consonants: / m, n, ŋ /.
   Eg.
   / m /: mirror, melody / n /: name, nose / ŋ /: working, long

5. **Liquids** : In this air stream outs on the sides of our tongue
   Eg.
   Initial sound : /ɹ/ - red /L/ - light
   Final sound : /ɹ/ - dear /L/ - full

6. **Glides** : in this mouth is moving constantly from a articulation to a vowel sound.
   Eg.
   /ɷ- win /ȳ/ - yellow

1.4 Place of Articulation

(i) While manner of articulation and voicing partition phonemes into the broad categories used by most languages, it is the place of articulation (point of narrowest vocal tract constriction) that enables finer discrimination of phonemes.
(ii) Consonants: Place of articulation is most often associated with consonants, rather than vowels because consonants use a relatively narrow constriction.
(iii) Along the vocal tract, approximately light regions or points, as shown in the figure, are traditionally associated with constant constriction, as follows:



Vowels are voiced components of the sound, that is |a|,|e|,|i|,|o|,|u|. The excitation is periodic excitation generated by the fundamental frequency of the vocal cords and the sound gets modulated when it passes via the vocal tract.

Place of articulation	How it is created
Bilabial	Contact between the lips.
Labio-dental	Contact between the lower lip and the upper teeth.
Dental	Contact between the lower lip and the upper teeth.
Alveolar	Contact between the tongue and the alveolar ridge (this is the ridged area between the upper teeth and the hard palate).
Palatal	Contact between the tongue and the hard palate or alveolar ridge.
Post-alveolar	The tongue makes contact with the back of the alveolar ridge.
Velar	The back of the tongue makes contact with the soft palate (velum).

Glottal	A restriction of the airflow at the glottis.

1.5 Word Boundary Detection

- Word Boundary Detection (WBD) is defined as identifying the start and the end of each word in a spoken utterance. Detecting the word boundary is used in many applications like keyword spotting, speech recognition system etc.
- Two types of speech are considered for the word boundary detection :
  Constrained Speech: In the constrained speech, the utterance and words has well defined boundaries with well defined pauses between words.
  Unconstrained Speech: Speech with no well defined boundaries, grammar or pauses is called unconstrained speech. Word boundary detection is challenging in the unconstrained speech and without a good detection algorithm word boundary detection can result in the many false alarms and misses.
- Word boundary detection can also be classified as speaker dependent and speaker independent. Speaker dependent systems are easy to develop when compared to speaker independent systems as one can have the information about utterance characteristics of the single speaker like pitch, but speaker independent systems have to be developed based on many speakers which makes it challenging due to different prosody, pitch etc.
- Word boundary detection is used by automatic speech recognition systems to isolate useful speech from background noise in order to extract speech patterns that further recognition stages can recognize.

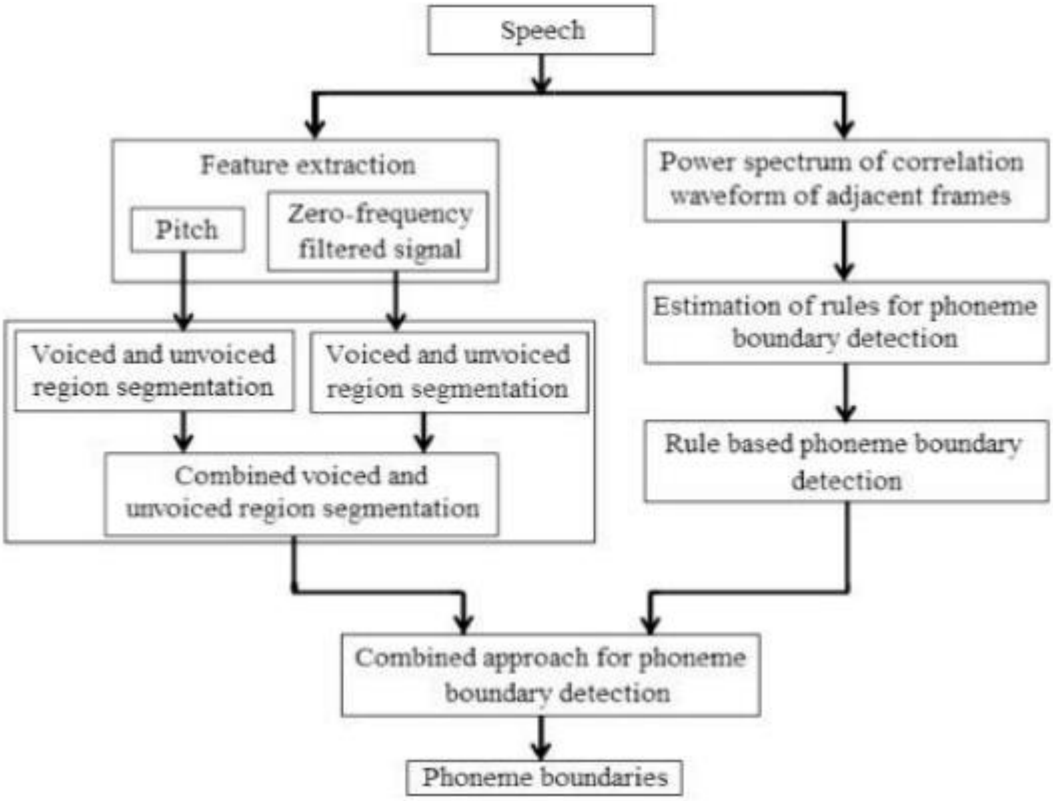
1.5.1 Need For Accurate Boundary Detection:
Word boundary detection is an integral party of an Automatic Speech Recognition System (ASRS). Accurate word boundary detection in an ASRS is important for three main reasons.
• Accurate word boundary detection lessens the computational load on further recognition stages.
• Greater accuracy in word boundary detection translates into greater accuracy in the overall speech recognition system.
• Word boundary detection techniques may incorporate word recognition techniques, in which case further recognition stages in the speech recognizer may be easier to implement.
For speech produced in a relatively noise-free environment, boundary detection is a simple problem. However, high levels of noise, be it background noise or noise in the transmission system, make word boundary detection difficult.

1.5.2 Approaches to Word Boundary Detection:

Rabiner and Juang broadly classified endpoint detection approaches as either explicit, implicit, or hybrid depending upon the degree of interaction between the endpoint detection and word recognition stages of the automated speech recognition system (ASRS).

1. **Rule-based approach**: In this approach, a set of rules is used to define word boundaries based on patterns in the text. For example, a rule might state that a word boundary occurs when there is a space, punctuation mark, or change in capitalization. In a rule-based approach to word boundary detection, a set of rules is created based on

the characteristics of the language in question, and these rules are used to identify the boundaries between words in a given text.


Speech
Feature extraction
Pitch
Zero-frequency filtered signal
Power spectrum of correlation waveform of adjacent frames
Voiced and unvoiced region segmentation
Voiced and unvoiced region segmentation
Estimation of rules for phoneme boundary detection
Combined voiced and unvoiced region segmentation
Rule based phoneme boundary detection
Combined approach for phoneme boundary detection
Phoneme boundaries

Some common rules that can be used for word boundary detection include: Spaces: In languages that use spaces between words, such as English, word boundaries can be identified by looking for spaces between words.

Punctuation: Punctuation marks, such as periods, commas, and semicolons, can also indicate word boundaries.

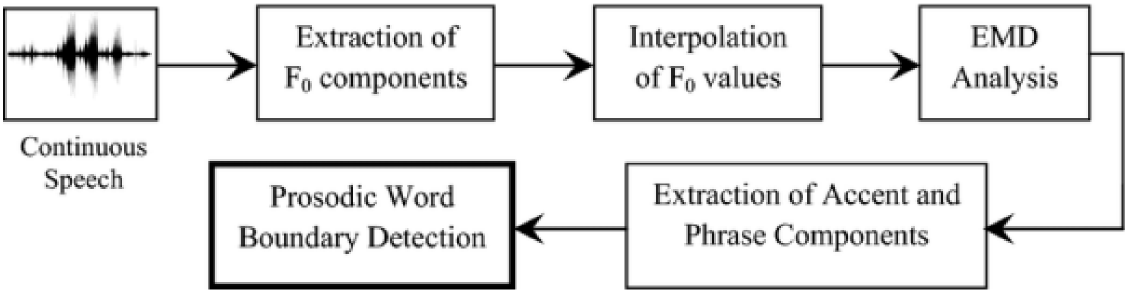
Capitalization: In some languages, such as German, the first letter of each noun is capitalized, which can be used to identify word boundaries.

Morphology: In languages where words are inflected or have prefixes or suffixes, such as Turkish or Arabic, word boundaries can be identified by looking for changes in the form of the word. Pronunciation: In some languages, such as Chinese, word boundaries can be identified by changes in pronunciation or intonation.

Once the rules have been established, they can be applied to a given text to identify word boundaries. For example, in English, the rule might be to look for spaces between words and punctuation marks, while in Arabic, the rule might be to look for changes in the form of the word.

While a rule-based approach can be effective in some cases, it may not be able to handle all possible cases, such as when a language has no spaces between words or has irregular spellings. In these cases, a machine learning approach may be more effective, as it can learn from large amounts of data and adapt to different types of texts.

2. **Statistical approach:** In this approach, a statistical model is trained on a large corpus of text to learn patterns in the language and predict word boundaries. For example, a model might learn that certain combinations of letters are more likely to occur at the beginning or end of a word.


Continuous Speech
Extraction of F_0 components
Interpolation of F_0 values
EMD Analysis
Extraction of Accent and Phrase Components
Prosodic Word Boundary Detection

Statistical approaches to word boundary detection involve using statistical models to identify boundaries between words in a text. One common approach is to use a language model, which is a probabilistic model that assigns probabilities to sequences of words. Language models are typically trained on large amounts of text data and use this data to learn the probability distribution of words in a language.

Once a language model has been trained, it can be used to identify word boundaries by calculating the probability of a given sequence of letters or sounds being a word. This is typically done by using the language model to calculate the probability of each possible word boundary location and selecting the location with the highest probability.

Another statistical approach to word boundary detection is to use a hidden Markov model (HMM), which is a statistical model that can be used to model sequences of observations. In the case of word boundary detection, the observations are typically sequences of letters or sounds, and the HMM is used to model the probability distribution of words in a language.

To use an HMM for word boundary detection, the model is first trained on a large corpus of text data. The training process involves estimating the probabilities of each possible sequence of letters or sounds, as well as the probabilities of transitioning from one sequence to another.

Once the HMM has been trained, it can be used to identify word boundaries by calculating the most likely sequence of hidden states (i.e., word boundaries) that corresponds to a given sequence of observations (i.e., letters or sounds).

Statistical approaches to word boundary detection are useful for identifying word boundaries in a variety of languages and text types. These approaches can be particularly useful in cases where linguistic knowledge of the language being analyzed is limited or incomplete. However, they may not always be accurate in cases where there are unusual or rare word formations that the statistical models have not encountered before.

3. **Machine learning approach:** In this approach, a machine learning algorithm is trained on labeled data to predict word boundaries. The algorithm might be trained on a combination of linguistic features, such as part-of-speech tags, and acoustic features, such as pitch and duration.

   Word boundary detection is a fundamental task in natural language processing (NLP) that involves identifying the boundaries between words in a text. In recent years, machine learning techniques have been widely used to perform this task with high accuracy. Here's an overview of how a machine learning approach to word boundary detection works:

   Data collection and preparation: The first step is to collect a large dataset of text, which includes both correctly segmented text (with word boundaries marked) and unsegmented text. The dataset is then preprocessed by removing any noise or irrelevant information and converting the text into a format suitable for machine learning algorithms.

   Feature extraction: The next step is to extract features from the text that can be used to train a machine learning model. Common features used for word boundary detection include character n-grams (sequences of characters of length n), word shape (the pattern of uppercase and lowercase letters in a word), and part-of-speech tags (the grammatical category of a word, such as noun or verb).

Model training: Once the features are extracted, a machine learning model is trained on the labeled dataset to learn the patterns that distinguish word boundaries from non-boundary positions. Various machine learning algorithms can be used for this task, such as decision trees, support vector machines (SVM), and neural networks.

Model testing and evaluation: The trained model is then tested on a separate, held-out dataset to evaluate its performance in identifying word boundaries. The performance of the model is typically measured using metrics such as precision, recall, and F1 score, which reflect the accuracy of the model in identifying word boundaries.

Deployment: Finally, the trained model can be deployed to identify word boundaries in new, unseen text. This can be done by running the text through the model and using the predicted labels to segment the text into words.

A machine learning approach to word boundary detection involves using a labelled dataset to train a model that can learn to distinguish word boundaries from non-boundary positions. With the availability of large datasets and powerful machine learning algorithms, this approach has achieved high accuracy in identifying word boundaries in a variety of languages and contexts.

4. **Hybrid approach:** In this approach, a combination of rule-based, statistical, and machine learning techniques is used to improve word boundary detection accuracy. Word boundary detection is the process of identifying the boundaries between words in a text. This is an important task in natural language processing, as many NLP applications rely on the ability to accurately identify the words in a sentence. There are several approaches to word boundary detection, including rule-based, statistical, and hybrid approaches.

   A hybrid approach to word boundary detection combines the strengths of rule-based and statistical approaches. In this approach, a set of rules is used to identify word boundaries in a text, but these rules are augmented with statistical information that is learned from a large corpus of annotated text. The rule-based component of the hybrid approach involves using linguistic rules to identify common patterns that indicate word boundaries. For example, in English, spaces are typically used to separate words, and punctuation marks such as periods, commas, and question marks often occur at the end of a word.

   Rules can be created to detect these patterns and use them to identify word boundaries. The statistical component of the hybrid approach involves using machine learning algorithms to learn patterns from a large corpus of annotated text. This approach involves training a machine learning model on a corpus of text that has been manually annotated with word boundaries. The model then uses this information to make predictions about the location of word boundaries in new, unseen text.

   By combining rule-based and statistical approaches, the hybrid approach to word boundary detection is able to take advantage of the strengths of both methods. Rule-based methods are precise and can be tailored to the specific characteristics of a language, while statistical methods are more flexible and can learn patterns that are difficult to capture with rules alone. The hybrid approach to word boundary detection is a promising area of research in NLP and has the potential to improve the accuracy and efficiency of word boundary detection in a wide range of applications.

   Word boundary detection is an important task in NLP that has many real-world applications. Improving the accuracy of word boundary detection can help to improve the performance of downstream NLP tasks

1.6 Arg-Max Computation

Argmax is a mathematical function.

- Argmax is an operation that finds the argument that gives the maximum value from a target function.
- Argmax is most commonly used in machine learning for finding the class with the largest predicted probability.
- Argmax can be implemented manually, although the argmax() NumPy function is preferred in practice.
- It is typically applied to another function that takes an argument. For example, given a function *g()* that takes the argument *x*, the *argmax* operation of that function would be described as follows:
  result = argmax(g(x))
  The *argmax* function returns the argument or arguments (*arg*) for the target function that returns the maximum (*max*) value from the target function.
  Consider the example where *g(x)* is calculated as the square of the *x* value and the domain or extent of input values (*x*) is limited to integers from 1 to 5:
- g(1) = 1^2 = 1
- g(2) = 2^2 = 4
- g(3) = 3^2 = 9
- g(4) = 4^2 = 16
- g(5) = 5^2 = 25
  We can intuitively see that the argmax for the function *g(x)* is 5.
  That is, the argument (*x*) to the target function *g()* that results in the largest value from the target function (25) is 5. Argmax provides a shorthand for specifying this argument in an abstract way without knowing what the value might be in a specific case.
- argmax(g(x)) = 5
  Note that this is not the *max()* of the values returned from function. This would be 25.
  It is also not the *max()* of the arguments, although in this case the argmax and max of the arguments is the same, e.g. 5. The *argmax()* is 5 because g returns the largest value (25) when 5 is provided, not because 5 is the largest argument.
  Typically, "*argmax*" is written as two separate words, e.g. "*arg max*". For example:
- result = arg max(g(x))
  It is also common to use the arg max function as an operation without brackets surrounding the target function. This is often how you will see the operation written and used in a research paper or textbook. For example:
- result = arg max g(x)
  You can also use a similar operation to find the arguments to the target function that result in the minimum value from the target function, called *argmin* or "*arg min*."

Arg-max computation is a mathematical operation used in machine learning and other fields to find the argument or input value that maximizes a given function or probability distribution.

The arg-max of a function is the input value that produces the maximum output value of the function. In machine learning, arg-max is commonly used to make predictions based on a trained model.

For example, in a classification problem where we want to predict the class label of a given input, the arg-max of the output probabilities is used to select the class with the highest probability.

This can be represented mathematically as follows:

Given a probability distribution p(y|x) that represents the probability of output y given input x, the arg-max of p(y|x) is: argmax_y p(y|x)

This means that we select the output y that maximizes the probability p(y|x) for a given input x.

In practice, the arg-max computation can be performed using numerical methods such as gradient descent or brute-force search. However, these methods can be computationally expensive for high-dimensional input spaces or complex models. Therefore, efficient algorithms such as dynamic programming and message passing have been developed to perform arg-max computations more efficiently. arg-max computation is an important operation in machine learning and other fields that allows us to select the most likely output given a specific input and probability distribution.

1.6.1 How Argmax Used in Machine Learning?

The argmax function is used throughout the field of mathematics and machine learning. Nevertheless, there are specific situations where you will see argmax used in applied machine learning and may need to implement it yourself.

The most common situation for using argmax that you will encounter in applied machine learning is in finding the index of an array that results in the largest value.An array is a list or vector of numbers.

It is common for multi-class classification models to predict a vector of probabilities (or probability-like values), with one probability for each class label. The probabilities represent the likelihood that a sample belongs to each of the class labels.

The predicted probabilities are ordered such that the predicted probability at index 0 belongs to the first class, the predicted probability at index 1 belongs to the second class, and so on.

Often, a single class label prediction is required from a set of predicted probabilities for a multi-class classification problem.

This conversion from a vector of predicted probabilities to a class label is most often described using the argmax operation and most often implemented using the argmax function.

Consider a multi-class classification problem with three classes: "*red*", "*blue*," and "*green*." The class labels are mapped to integer values for modeling, as follows:

- red = 0
- blue = 1
- green = 2

Each class label integer values maps to an index of a 3-element vector that may be predicted by a model specifying the likelihood that an example belongs to each class.

Consider a model has made one prediction for an input sample and predicted the following vector of probabilities:

- yhat = [0.4, 0.5, 0.1]

We can see that the example has a 40 percent probability of belonging to red, a 50 percent probability of belonging to blue, and a 10 percent probability of belonging to green.

We can apply the argmax function to the vector of probabilities. The vector is the function, the output of the function is the probabilities, and the input to the function is a vector element index or an array index.

- arg max yhat

We can intuitively see that in this case, the argmax of the vector of predicted probabilities (yhat) is 1, as the probability at array index 1 is the largest value. Note that this is not the max() of the probabilities, which would be 0.5. Also note that this is not the max of the arguments, which would be 2. Instead it is the argument that results in the maximum value, e.g. 1 that results in 0.5.

- arg max yhat = 1

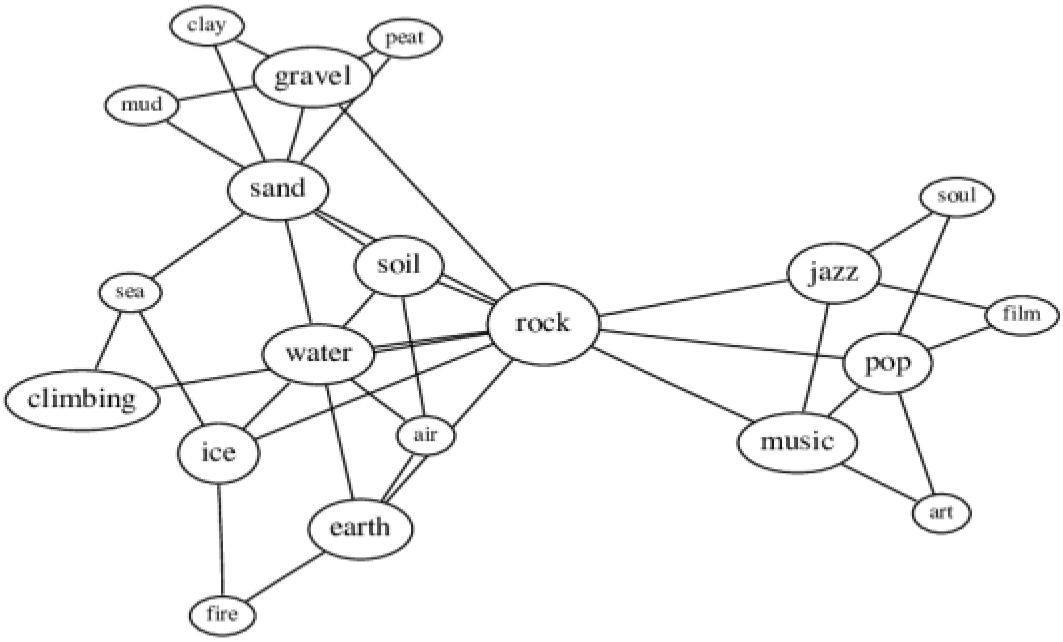
We can then map this integer value back to a class label, which would be "*blue*."

- arg max yhat = "blue"

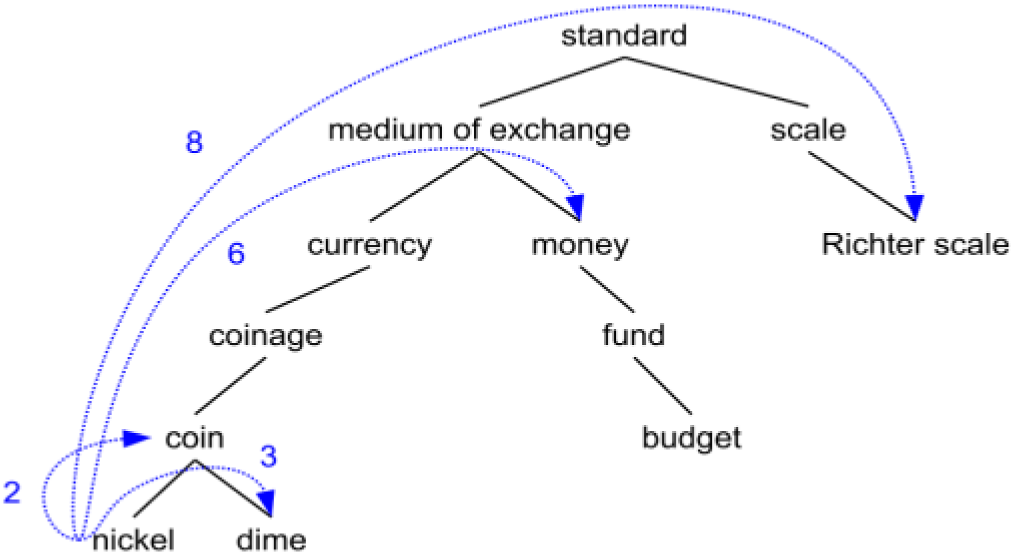
1.7 Lexical Knowledge Networks

Lexical resources play an important role in natural language processing tasks. Past couple of decades have shown an immense growth in the development of lexical resources such as wordnet, Wikipedia, ontologies etc. These resources vary significantly in structure and representation formalism.
In order to develop applications that can make use of different resources, it is essential to link these heterogeneous resources and develop a common representation framework.

Lexical knowledge networks, also known as lexical semantic networks, are a type of knowledge representation used in natural language processing (NLP) and computational linguistics. They represent the relationships between words based on their meaning or semantic content.
In a lexical knowledge network, words are represented as nodes, and the relationships between words are represented as edges. The edges may be labeled with a specific relationship type, such as "synonym", "antonym", "hypernym" (a word that is more general than another word), or "hyponym" (a word that is more specific than another word).
There are several different types of lexical knowledge networks, each with their own specific characteristics and uses. Some of the most well-known lexical knowledge networks include WordNet, FrameNet,ConceptNet.



WordNet is a lexical database of English that groups words into sets of synonyms called synsets, each expressing a distinct concept. It also includes relationships between words, such as hypernymy and meronymy (a word that refers to a part of another word).



WordNet is a lexical database of English words and their relationships, developed at Princeton University's Cognitive Science Laboratory. It is widely used in natural language processing and computational linguistics for tasks such as word sense disambiguation, text classification, and machine translation. WordNet groups English words into sets of synonyms, called synsets, which are defined by a common sense or meaning. Each synset contains one or more words that are related in meaning and can be used interchangeably in certain contexts. For example, the synset for "car" includes the words "automobile", "vehicle", and "motorcar". WordNet also includes relationships between words, such as hypernymy (a word that is more general than another word), hyponymy (a word that is more specific than another word), and meronymy (a word that refers to a part of another word). These relationships are represented by links between the synsets, forming a network of interrelated words and meanings. One of the key advantages of WordNet is its extensive coverage of the English language. It includes over 155,000 words and phrases, organized into more than 117,000 synsets. In addition, WordNet is freely available for use and has been integrated into many NLP applications and tools. WordNet has also inspired the development of similar lexical databases in other languages, such as EuroWordNet and BalkaNet, which have similar structures and relationships between words

WordNet is a valuable resource for natural language processing and computational linguistics, providing a structured representation of the English language and its many meanings and relationships. WordNet is a lexical database for the English language that was developed at Princeton University.

It is a widely-used resource in natural language processing and computational linguistics, and has several key characteristics that make it a useful tool for these fields:

1. Hierarchical structure: WordNet organizes words into a hierarchy of semantic relationships, with each word belonging to a synset (a set of synonyms) that expresses a distinct concept. Synsets are linked together by relationships such as hypernymy (a word that is more general than another word), hyponymy (a word that is more specific than another word), meronymy (a word that refers to a part of another word), and holonymy (a word that refers to a whole that contains another word).

2. Large coverage: WordNet includes over 155,000 words and 117,000 synsets, making it a comprehensive resource for lexical information about the English language.

3. Part-of-speech information: WordNet provides information about the part-of-speech (POS) of each word, including nouns, verbs, adjectives, and adverbs. This allows NLP systems to disambiguate between different senses of a word based on its POS.

4. Rich linguistic information: WordNet includes additional linguistic information about each word, such as examples of usage, collocations, and derivational relationships (relationships between words formed from the same root).

5. Multilingual support: WordNet has been translated into several other languages, including Spanish, Italian, and Korean, making it a valuable resource for cross-lingual applications. WordNet's hierarchical structure, large coverage, part-of-speech information, rich linguistic information, and multilingual support make it a valuable resource for natural language processing and computational linguistics. Its structure and information can be used to support various NLP applications, such as word sense disambiguation, semantic role labeling, and information retrieval.

FrameNet is a lexical database that organizes words based on the underlying concepts or frames they represent. It includes information about the syntactic and semantic relationships between words and the frames they belong to.

FrameNet is a lexical database and knowledge representation framework that was developed at the International Computer Science Institute in Berkeley, California. It is similar to WordNet in that it organizes words into sets of synonyms, but it differs in its focus on the underlying concepts or "frames" that these words represent. In FrameNet, each frame represents a conceptual structure that describes a particular situation or event. Frames are composed of a set of semantic roles, which represent the different participants and their relationships in a given situation. For example, the "buying" frame might include roles such as "buyer", "seller", "product", and "price". Each semantic role is associated with a set of words or phrases that can fill that role in a given context. FrameNet includes a large number of annotated examples of natural language text that illustrate how words and frames are used in context. This makes it a valuable resource for natural language processing applications such as semantic role labeling, which involves identifying the semantic roles that are played by different words in a given sentence. One advantage of FrameNet over other lexical databases is that it captures the way that different words can be used to express the same frame. For example, in the "buying" frame, words such as "purchase", "acquire", and "obtain" might all be included as options for the "buy" semantic role. This allows NLP systems to better handle variation in language use and to make more accurate predictions about the meaning of words in context. FrameNet provides a valuable resource for natural language processing and computational linguistics, particularly in its focus on the underlying conceptual structures that are expressed by language. It is widely used in research and has been incorporated into various NLP tools and applications. FrameNet is a lexical database that organizes words based on the underlying concepts or "frames" they represent. Frames are representations of typical situations, events, or scenarios, and they consist of several components, including core elements, peripheral elements, and the relations between them. Some key characteristics of FrameNet are:

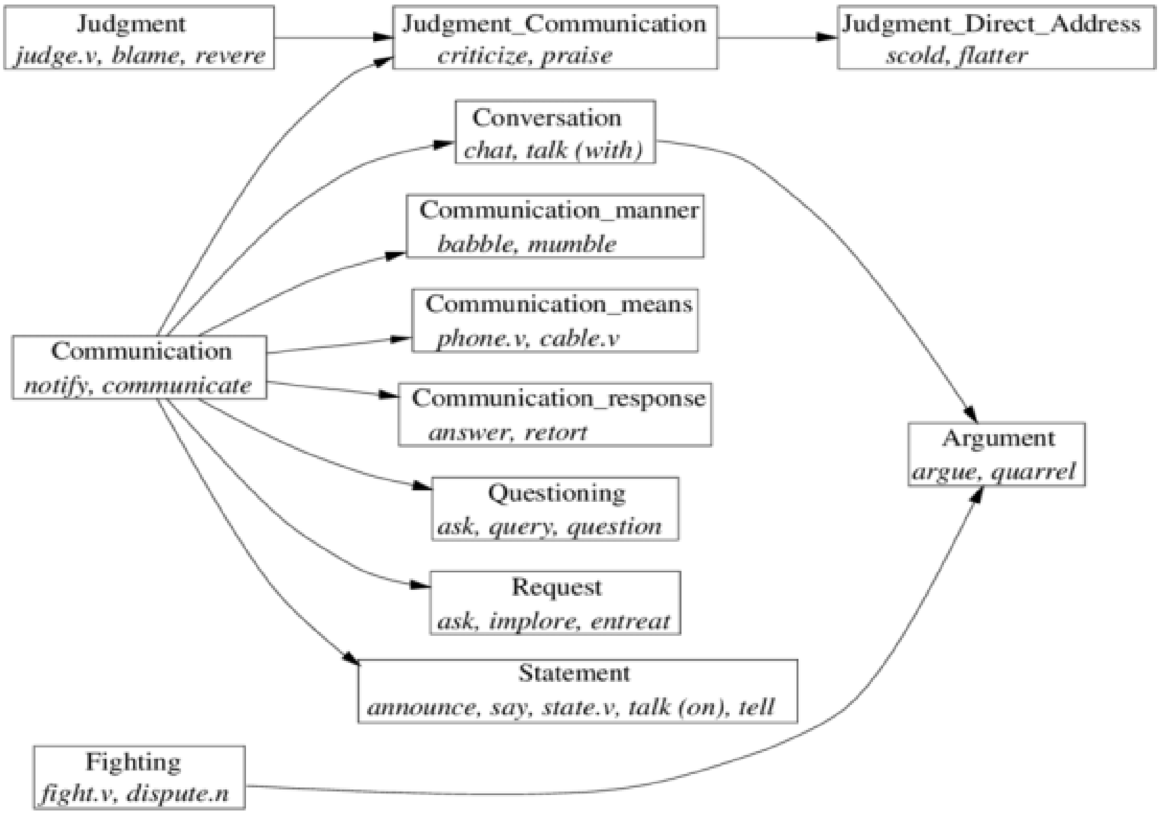
1. Focus on semantic frames: Unlike other lexical databases that focus on individual words and their relationships, FrameNet focuses on the underlying concepts or "frames" that words represent. This allows for a more comprehensive understanding of the semantic content of a text.

2. Annotations of words in context: FrameNet includes annotations of words in context, which means that each word is linked to a specific frame and a set of semantic roles that

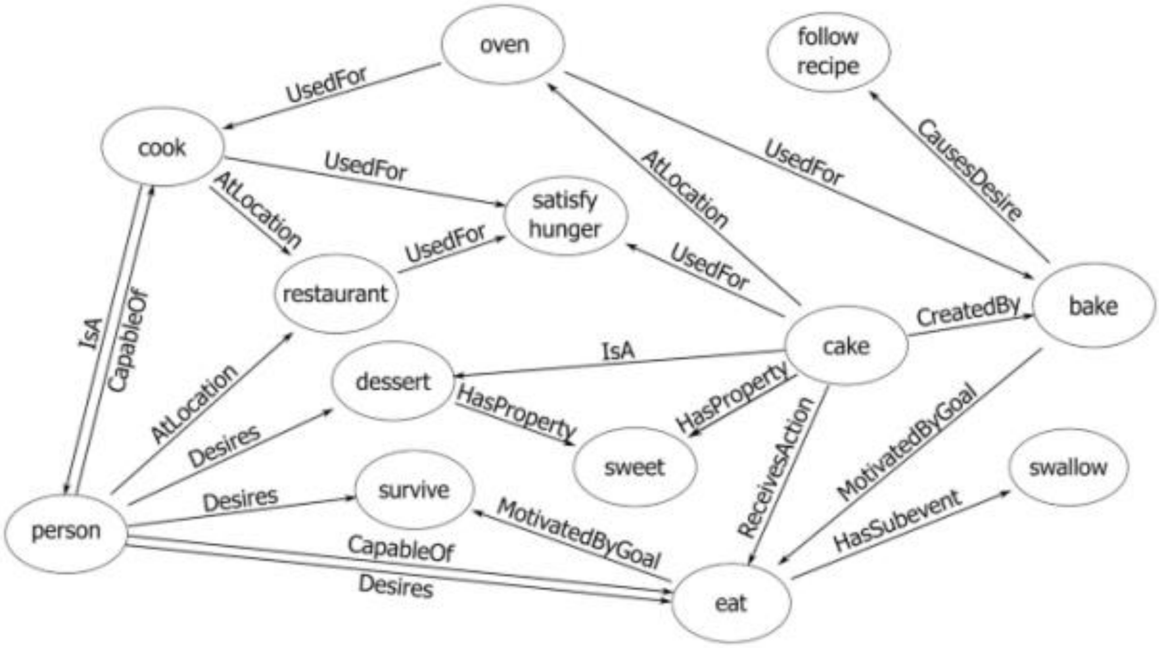
it plays in that frame. This allows NLP systems to identify the appropriate meaning of a word based on its context.

3. Linguistic resources: FrameNet includes a range of linguistic resources, including examples of usage, annotations of syntactic and semantic structures, and relations between frames. These resources allow for more detailed analyses of language use and the relationships between words. 4. Multiword expressions: FrameNet includes information about multiword expressions (MWEs), such as idioms and phrasal verbs. This allows NLP systems to recognize and interpret these expressions more accurately.

5. Rich metadata: FrameNet includes metadata about the sources of its data, such as the genre and domain of the text, which can be used to support more nuanced analyses of language use. FrameNet's focus on semantic frames, annotations of words in context, linguistic resources, support for multiword expressions, and rich metadata make it a valuable resource for natural language processing and computational linguistics. Its structure and information can be used to support various NLP applications, such as semantic role labeling, information extraction, and machine translation.



ConceptNet is a lexical knowledge network that represents concepts and their relationships to one another. It includes relationships such as synonymy, antonymy, hypernymy, hyponymy, and more.



ConceptNet is a knowledge graph that captures commonsense knowledge and relationships between concepts. It was developed by researchers at the Massachusetts Institute of Technology (MIT) and is designed to support natural language processing applications and research. ConceptNet contains a large collection of concepts, represented as nodes in a graph, and the relationships between these concepts, represented as edges. The concepts in ConceptNet are organized into a hierarchy of categories, such as "animals", "food", and "music". The relationships between concepts include "is-a" relationships (e.g., "cat" is-a "animal"), "partof" relationships (e.g., "wheel" is part-of "car"), and "related-to" relationships (e.g., "coffee" is related-to "caffeine"). One notable feature of ConceptNet is its focus on capturing commonsense knowledge and relationships between concepts that are not typically captured in other knowledge graphs or lexical resources. For example, ConceptNet includes information about cultural associations between words, such as "football" being associated with "tailgating" and "beer". It also includes information about the emotional valence of words, such as "happy" being associated with positive emotions and "sad" being associated with negative emotions. ConceptNet has been used in various natural language processing applications, such as sentiment analysis, named entity recognition, and semantic similarity analysis. It has also been used to support research in cognitive science and artificial intelligence. The open-source nature of ConceptNet allows it to be extended and improved by the research community, making it a valuable resource for natural language processing and computational linguistics. ConceptNet is a semantic network and knowledge graph that represents concepts and their relationships using a set of nodes and edges. It was developed at the Massachusetts Institute of Technology (MIT) and is an open-source project that is freely available to the public. Here are some of the key characteristics of ConceptNet:

1. Multilingual support: ConceptNet supports multiple languages, including English, Chinese, and Spanish, among others. This allows it to be used in cross-lingual natural language processing applications.

2. Broad coverage: ConceptNet includes a wide range of concepts from various domains, such as common objects, actions, events, and emotions, among others. It also includes

relationships between these concepts, such as synonymy, hypernymy, meronymy, and entailment, among others.
3. Open-source: ConceptNet is an open-source project that is freely available to the public, which allows researchers and developers to access the underlying data and build applications based on it.
4. Machine-readable format: ConceptNet is available in a machine-readable format, which allows it to be easily integrated into natural language processing and machine learning systems.
5. Crowdsourcing: ConceptNet is built using a combination of automated methods and crowdsourcing, which involves asking humans to provide annotations or corrections to the data. This allows it to be continually updated and improved over time.
6. Linked data: ConceptNet is part of the Linked Open Data (LOD) initiative, which aims to make data available on the web in a standardized and interconnected way. This allows it to be easily integrated with other semantic web resources.
ConceptNet provides a valuable resource for natural language processing and computational linguistics, particularly in its broad coverage of concepts and relationships across multiple languages. Its open-source and machine-readable format make it easy to integrate into NLP and machine learning systems, while its crowdsourcing and linked data approach ensures that it is continually updated and improved over time. Lexical knowledge networks are useful in NLP applications such as semantic role labeling, word sense disambiguation, and machine translation. They allow for a more nuanced understanding of the relationships between words and help NLP systems make more accurate predictions about the meaning of words and their context.

Hownet- HowNet is an online common-sense knowledge base unveiling inter-conceptual relations and inter-attribute relations of concepts as connoting in Chinese and English bilingual lexicons.
MindNet- MindNet represents a general methodology for acquiring, structuring, accessing, and exploiting semantic information from natural language text.
VerbNet- VerbNet (VN)is the largest on-line network of English verbs that links their syntactic and semantic patterns. It is a hierarchical, domain-independent, broad-coverage verb lexicon with mappings to other lexical resources, such as WordNet, PropBank and FrameNet.

-----X-----

Prepared By: Prof Srushti Raut