Unit 1

Introduction to Natural Language Processing

Prof. Pranjal Pandit
CSE AI

CAUA32201: NATURAL LANGUAGE PROCESSING

Teaching Scheme	Examination Scheme				
Credits: 04 **Lecture (L):** 03 hrs / week **Practical (P):** 02 hr / week	CIE	SCE	ESE	PR	Total
	20	20	40	20	100
Prerequisite: Theory of Computation, Compiler Design and Basic Understanding of Probability Theory					

Course Objectives: -

1. To be familiar with fundamental concepts and techniques of natural language processing (NLP).
2. To acquire the knowledge of various morphological, syntactic, and semantic NLP tasks
3. To develop the various language modelling techniques for NLP
4. To use appropriate tools and techniques for processing natural languages
5. To Describe Applications of NLP and Machine Translations
6. To comprehend the advance real-world applications in NLP domain

NLP-6 Units

1. Introduction to Natural Language Processing
2. Language Syntax and Semantics
3. Language Modelling
4. Information Retrieval using NLP
5. Machine Translation
6. NLP Tools and Techniques

Unit 1: Introduction to Natural Language Processing

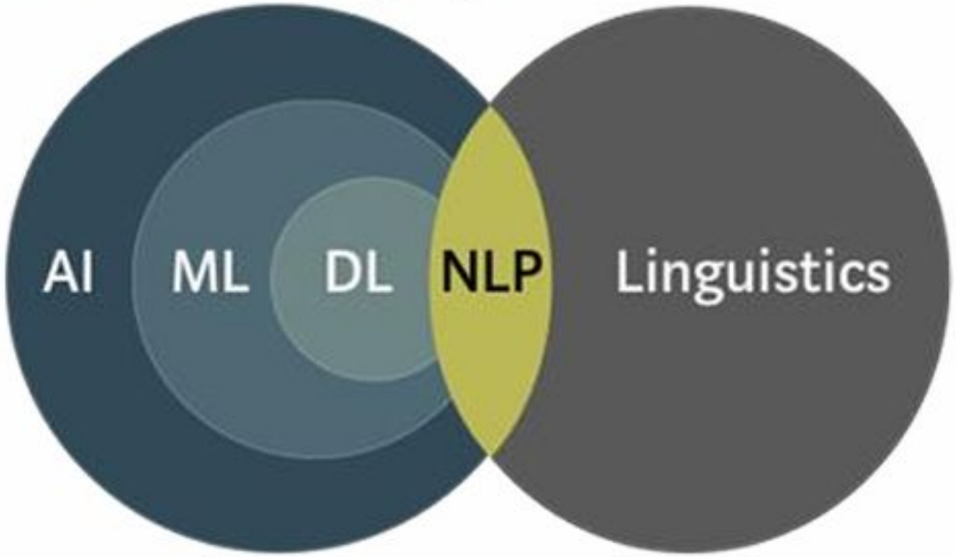
- Introduction: Scope, Applications and Challenges.
- Brief history and evolution of NLP: Programming languages Vs Natural Languages, Are Natural Languages Regular? Finite automata for NLP
- Stages of NLP
- NLP Basics of text processing: Tokenization, Stemming, Lemmatization,
- Part of Speech Tagging;
- NLP Components: Syntax, Semantics, and Pragmatics;
- Introduction to Regular Expressions and their Applications in NLP.

Outline

- What is NLP?
- What makes NLP challenging?
- Some common NLP tasks
- NLP Methods
- Some practical advice
- Code walk-through

What is NLP?

Natural Language Processing (NLP) is all about making computers understand and interact with humans, in their language(s).



source

Other related disciplines: cognitive science, human-computer interaction

Cognitive science is the interdisciplinary study of the **mind and its processes**, encompassing **psychology, neuroscience, linguistics, philosophy, artificial intelligence, and anthropology.**

It explores **how humans perceive, think, learn, and solve problems**, aiming to understand the **nature of intelligence and consciousness** through both biological and computational approaches.

Where is NLP useful?

It is a part of many day to day applications we use now

- Email filters, virtual assistants, information seeking, language translation etc.

There are so many business use cases where NLP plays a prominent role.

- Customer support, analytics, content generation, data protection etc.

It is also used across various disciplines to address domain specific questions

- E.g., analyzing political speeches for social scientists to protecting enterprise communications in cybersecurity experts

A Brief Timeline of NLP

1950
- Interest in Translation
- "Syntactic Structures" by Chomsky
- Generative Grammars

1960
- ELIZA
- ALPAC Report and First AI Winter

1970
- Case Grammars, Semantic Networks, and Conceptual Dependency Theory

1980
- Ontologies
- Expert Systems (e.g. MYCIN)

1990
- Statistical Models
- RNNs and LSTMs

2000
- Language Modelling
- Word Embeddings
- Google Translate

2010
- Word2Vec
- Rise of LSTMs and CNNs
- Encoder-Decoder
- Attention and Transformers
- Pre-trained Models and Transfer Learning

2020
- GPT3 and Large Language Models

1940s–1950s: The Foundations

- **Early Computational Linguistics:** Inspired by cryptography during World War II, initial efforts in language processing were rooted in computational analysis.

- **Turing Test (1950):** Alan Turing **proposed a test to determine a machine's ability** to exhibit intelligent behavior indistinguishable from a human, influencing NLP's goals.

1960s: Rule-Based Systems

- **First Machine Translation (MT):** The Georgetown-IBM experiment (1954) translated Russian to English, marking an early NLP success.

- **Chomsky's Influence:** Noam Chomsky’s transformational grammar introduced structural rules, laying the groundwork for syntactic parsing.

1970s: Knowledge-Based Approaches

- **SHRDLU (1970):** Terry Winograd developed this system, which could understand and manipulate objects in a virtual world using natural language.

- **Semantic Networks and Ontologies:** Early systems like ELIZA used predefined scripts, and more advanced methods began to encode knowledge semantically.

1980s: Statistical Methods

- **Shift to Probability:** With the availability of larger datasets and computing power, NLP shifted from rule-based systems to statistical models.

- **Hidden Markov Models (HMMs):** These were widely used for tasks like speech recognition and part-of-speech tagging.

1990s: Machine Learning

- **Corpus-Based NLP:** Large annotated corpora like the Penn Treebank became standard for training and evaluating models.

- **Support Vector Machines (SVMs) and Naive Bayes:** These algorithms improved text classification and other NLP tasks.

2000s: Data-Driven NLP

- **Rise of the Web:** The explosion of digital text provided abundant data for training models.

- **Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA):** Techniques for topic modeling and dimensionality reduction emerged.

2010s: Deep Learning Revolution

- **Neural Networks:** Techniques like Word2Vec, GloVe, and recurrent neural networks (RNNs) transformed how NLP models learned contextual relationships.
- **Transformers:** The introduction of the Transformer architecture in 2017 (Vaswani et al.) and models like BERT and GPT revolutionized NLP with superior performance on complex tasks.

- **Applications:** Chatbots, sentiment analysis, machine translation, and summarization saw significant advancements.

2020s: Large Language Models (LLMs)

- **GPT-3 and Beyond:** OpenAI's GPT series showcased the potential of large-scale pre-trained models for generating human-like text.
- **Multimodal and Cross-Lingual NLP:** Models like GPT-4 and CLIP integrate text with images and expand capabilities across languages.
- **Ethics and Fairness:** The field grapples with challenges like bias, misinformation, and ethical AI use.

NLP continues to evolve, pushing the boundaries of machine understanding and interaction with human language.

What makes NLP challenging?

Language is Ambiguous

See these newspaper headlines:

- *"Children make delicious snacks"*
- *"Dead expected to rise"*
- *"Republicans grill IRS chief over lost emails"*

Normal, grammatical sentences can be ambiguous too:

- *"I saw a man on a hill with a telescope."*
- *"Look at the man with one eye"*

We are not even talking about ambiguities involving speech or alternative interpretations due to stress/emphasis on some word.

There are many forms of ambiguity

1. Lexical ambiguity: *I am at a bank vs I am at a river bank*
2. Structural ambiguity: *I saw the man on the hill with a telescope.*
3. Semantic ambiguity: *John and Mary are married (to each other? or to different people?)*
4. Referential ambiguity: *She dropped the plate on the table and broke it*
5. Ambiguity from non-literal language use: *Time flies like an arrow.*

World Knowledge

What is common knowledge for humans may not be so for a computer.

Dog bit man.
Man bit dog.

Linguistically, both of them are similar. But, we know only the first one is "normal" English sentence because we have "world knowledge". How can an NLP system/a computer know that?

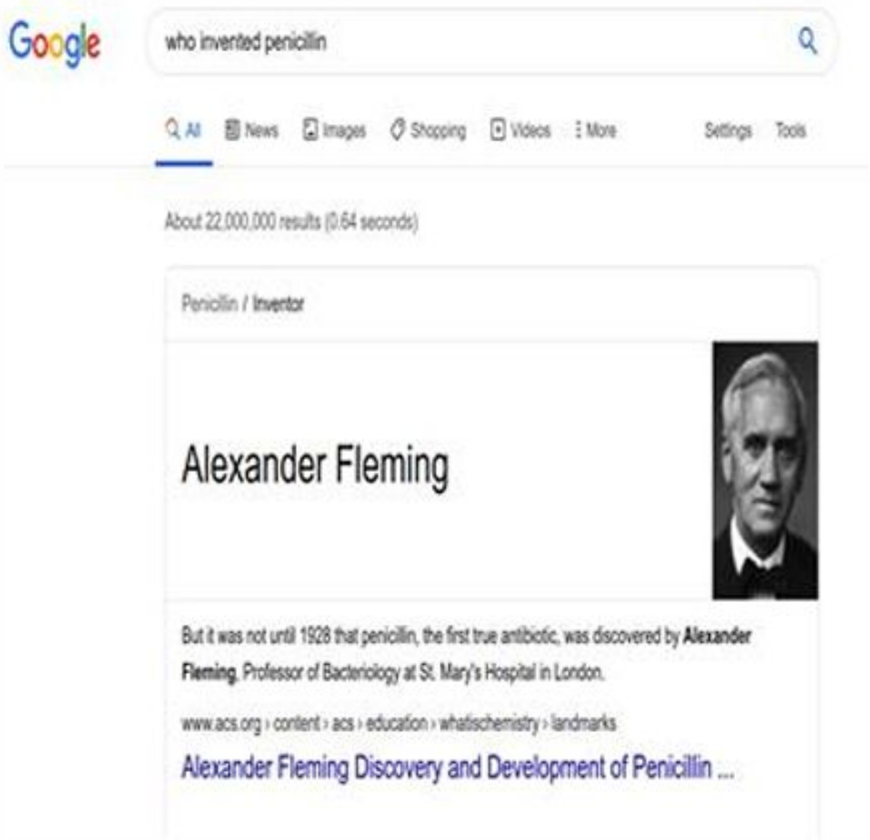
Language is Diverse

What is “language?”

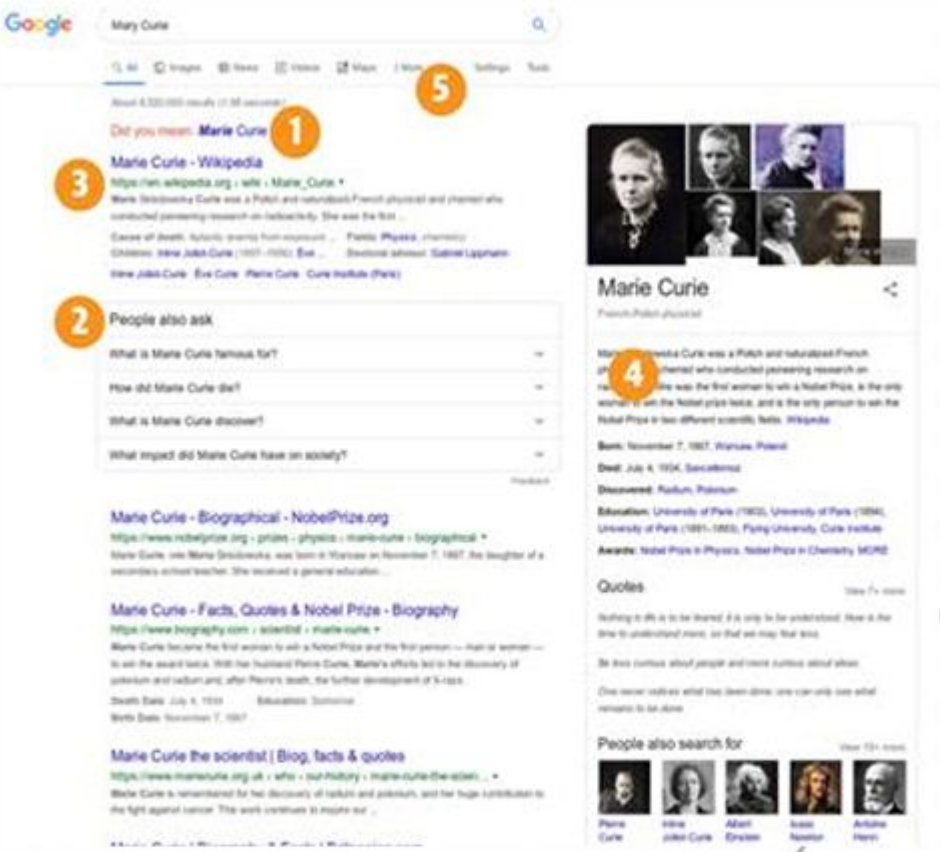
- Many different forms: News articles, tweets, logs, legal texts, chats, etc
- Creative use, and keeps changing over time.
- Many spelling variations, slangs, dialects, styles etc.
- Above all, thousands of languages in the world.

Some common NLP tasks

Search

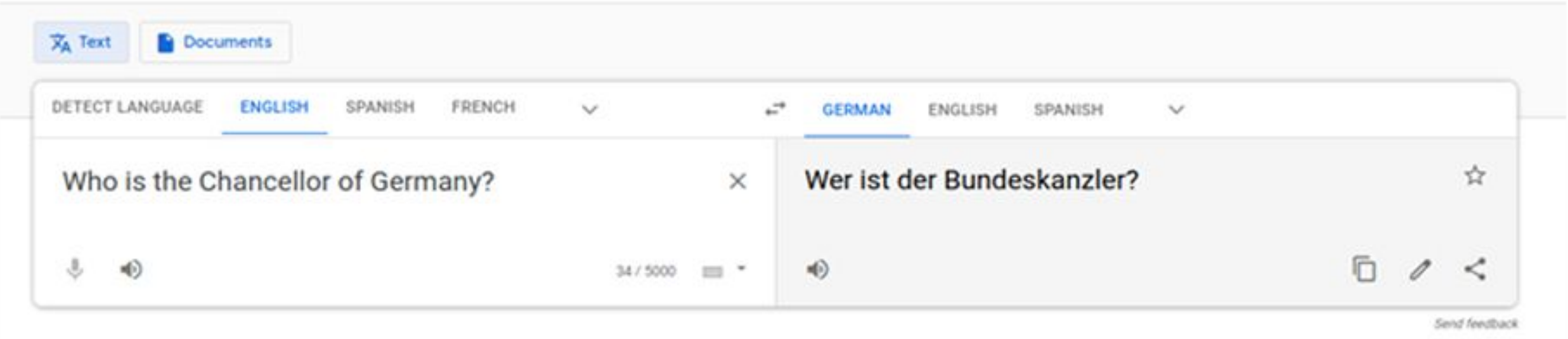


Google
who invented penicillin
All
News
Images
Shopping
Videos
More
Settings
Tools
About 22,000,000 results (0.64 seconds)
Penicillin / Inventor
Alexander Fleming
But it was not until 1928 that penicillin, the first true antibiotic, was discovered by Alexander Fleming, Professor of Bacteriology at St. Mary's Hospital in London.
www.acs.org › content › acs › education › whatischemistry › landmarks
Alexander Fleming Discovery and Development of Penicillin ...

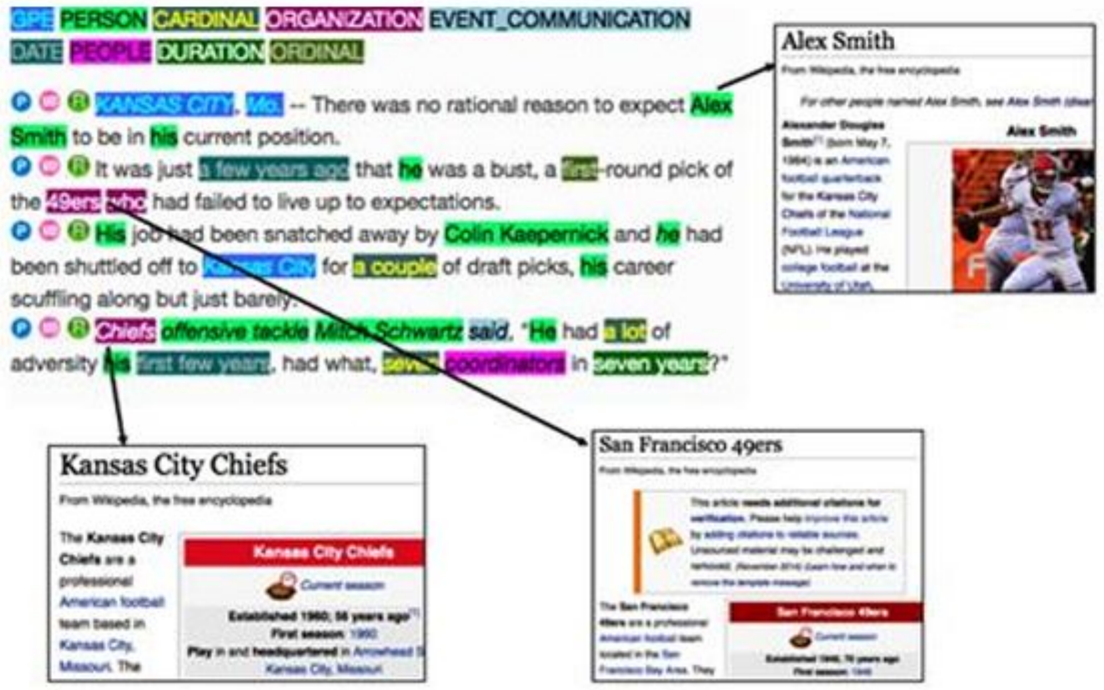


Google
Mary Curie
5
Did you mean: Marie Curie
1
Marie Curie - Wikipedia
3
2
People also ask
Marie Curie - Biographical - NobelPrize.org
Marie Curie - Facts, Quotes & Nobel Prize - Biography
Marie Curie the scientist | Blog, facts & quotes
Marie Curie
4
Quotes
People also search for

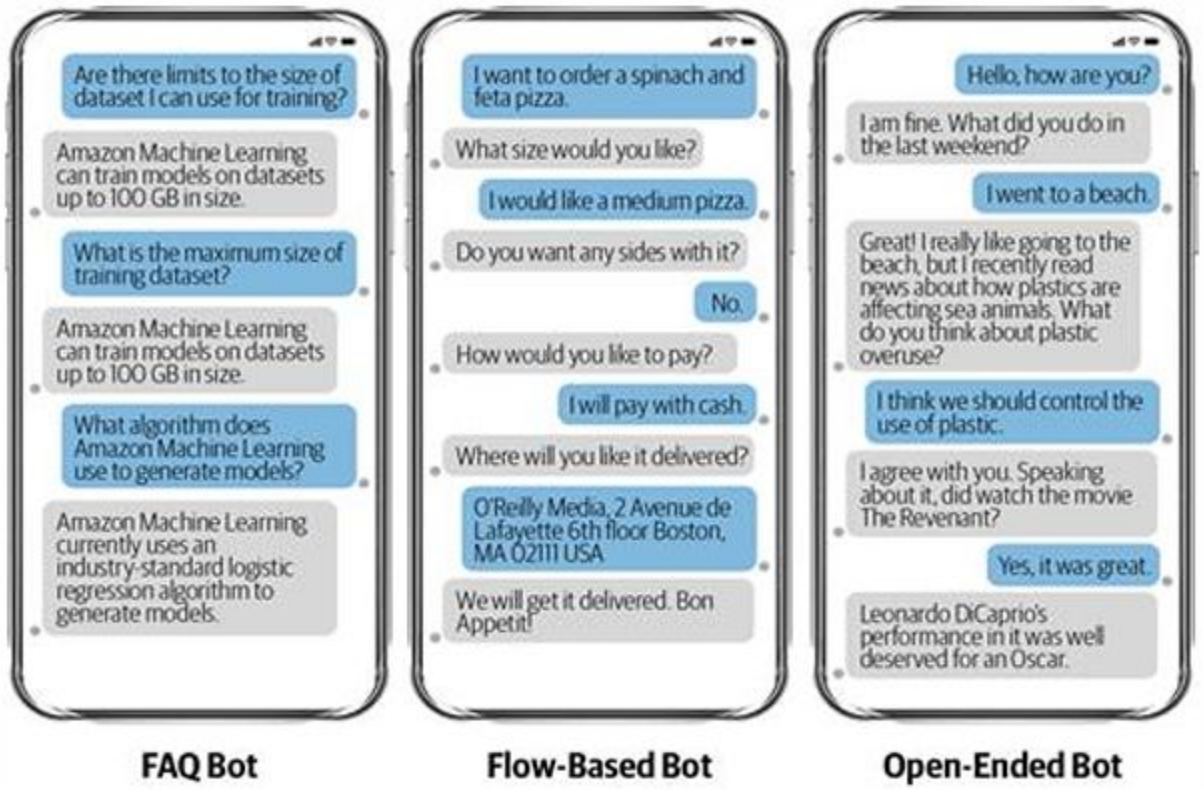
Machine Translation



Information Extraction

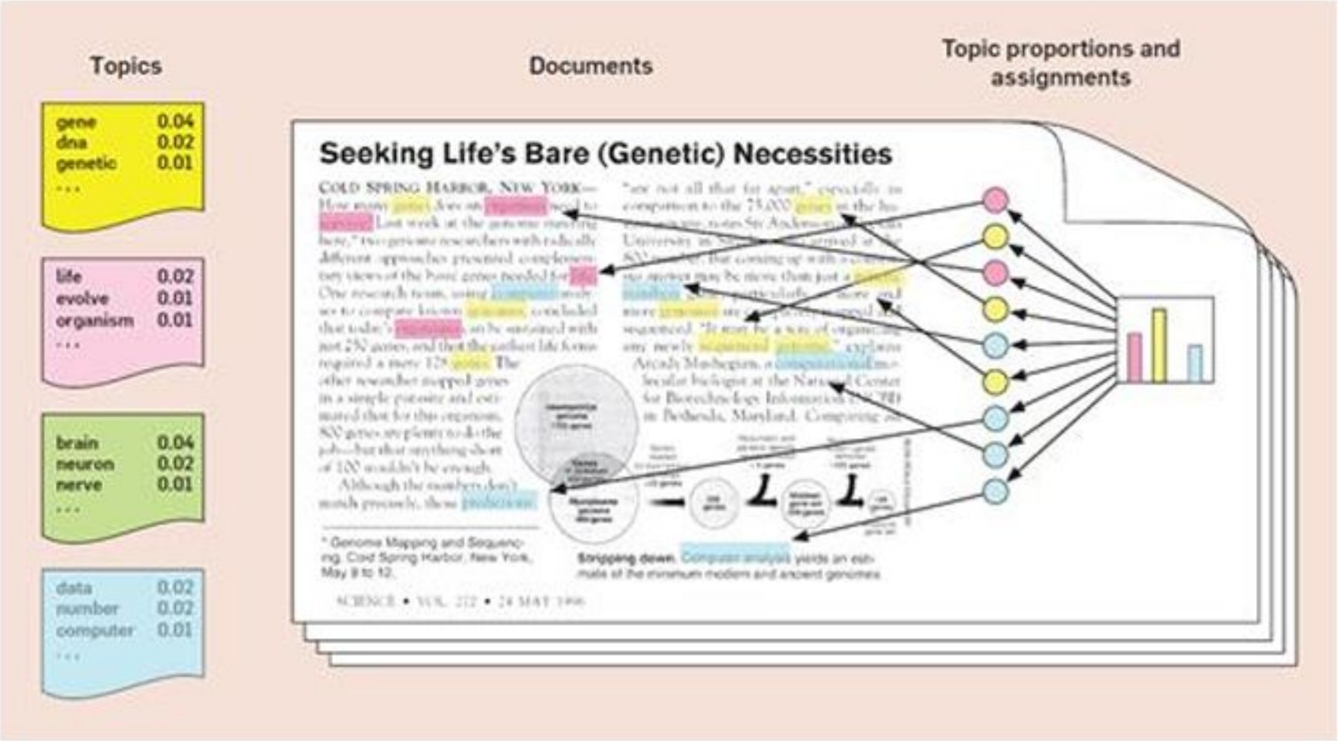


Chatbots



Are there limits to the size of dataset I can use for training?
Amazon Machine Learning can train models on datasets up to 100 GB in size.
What is the maximum size of training dataset?
Amazon Machine Learning can train models on datasets up to 100 GB in size.
What algorithm does Amazon Machine Learning use to generate models?
Amazon Machine Learning currently uses an industry-standard logistic regression algorithm to generate models.
FAQ Bot
I want to order a spinach and feta pizza.
What size would you like?
I would like a medium pizza.
Do you want any sides with it?
No.
How would you like to pay?
I will pay with cash.
Where will you like it delivered?
O'Reilly Media, 2 Avenue de Lafayette 6th floor Boston, MA 02111 USA
We will get it delivered. Bon Appetit!
Flow-Based Bot
Hello, how are you?
I am fine. What did you do in the last weekend?
I went to a beach.
Great! I really like going to the beach, but I recently read news about how plastics are affecting sea animals. What do you think about plastic overuse?
I think we should control the use of plastic.
I agree with you. Speaking about it, did watch the movie The Revenant?
Yes, it was great.
Leonardo DiCaprio's performance in it was well deserved for an Oscar.
Open-Ended Bot

Topic Modeling



Uncovering hidden themes from large datasets- By clustering words into topics.

It uses algorithms like Latent Dirichlet Allocation (LDA) or Non-Negative Matrix Factorization (NMF).

Applications include analysing customer reviews, organizing research articles, content recommendation, social media trend analysis, and automated document categorization.

Programming Language Vs Natural Language

Aspect	Programming Language	Natural Language
Purpose	Communication with computers	Communication between humans
Ambiguity	None	Often ambiguous
Structure	Fixed syntax	Flexible grammar and vocabulary
Evolution	Slow and deliberate	Rapid and ongoing
Error Handling	Explicit errors	Contextual clarification
Interactivity	Predefined	Dynamic and spontaneous
Learning Curve	Systematic learning	Natural acquisition

Are Natural Languages Regular?

What Are Regular Languages?

- Regular languages are a **class of formal languages** that can be expressed using **regular expressions** or modeled by **finite-state automata (FSA)**.

Regular Expression (Regex)

A **regular expression** is a sequence of characters defining a search pattern, typically used for pattern matching and text manipulation.

Key Features:

- **Purpose**: Identify, search, or replace specific text patterns in strings.
- **Syntax**: Includes special symbols like *, +, ?, [], and () to define rules.

Imagine you’re **trying to find something specific** in a large book or a file—like a phone number, a date, or just a word. Regular expressions are like a **super-smart magnifying glass** that helps you do that. 🚀

What Is It?

A **regular expression** is just a **fancy set of rules** you write **to describe what you're looking for**. These rules are made of special symbols and characters that tell the computer how to find patterns.

Examples:

1. Finding a Word
You can search for "cat" in a book. A regex for this is simply cat

2. **Finding Any Word That Starts with "c"**
Imagine you’re not sure what comes after "c"—it could be "cat," "car," or "cup." You can write: c[a-z] This means: Find a "c" followed by **any letter** (a to z).

3. **Finding Numbers**
Let’s say you want to find numbers like "123" or "4567." You can write \d+

Rule	What It Does	Example
.	Matches any single character (except a newline).	c.t matches "cat" or "cut".
[ ]	Matches any character inside the brackets.	[aeiou] matches vowels.
*	Matches zero or more of the character before it.(Kleen Clouser)	ba* matches "b" or "baa".
+	Matches one or more of the character before it. (Positive Clouser)	ba+ matches "ba" or "baa".
\d	Matches any digit (0-9).	\d matches "5" or "42".

Example:

- Regex: \bcat\b
  - Matches the word "cat" as a whole word.
- Application in NLP: Tokenization, searching for specific patterns like email addresses

([a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,})

Regular languages **are simple and have limited expressive power**. They cannot handle nested structures or dependencies.

Complexity of Natural Languages

Natural languages exhibit properties that exceed the capabilities of regular languages:

- **Nested Dependencies**: Sentences in natural languages often have hierarchical structures that require context-free or context-sensitive grammars.
  - Example: "The cat that chased the mouse that ate the cheese is black."
    - This sentence has nested clauses, which a finite automaton cannot process.
- **Long-Distance Dependencies**: Words and phrases in a sentence can depend on each other across large distances.
  - Example: "The book that John said Mary borrowed from the library is on the table."
    - The subject "book" is linked to the verb "is," skipping intermediate words.
- **Ambiguity and Context Dependence**: Natural languages rely heavily on context, tone, and real-world knowledge.
  - Example: "Visiting relatives can be tiresome."
    - Ambiguous without additional context.

How Natural Languages Are Modeled

- While natural languages are not strictly regular, they can be **approximated** as regular languages for certain applications like keyword matching or tokenization.
- More sophisticated grammars, such as **context-free grammars (CFGs)** or **context-sensitive grammars**, are often used for parsing and understanding natural languages.

In practical natural language processing:

- Regular expressions are used for simple text matching tasks (e.g., email validation).
- Parsing (Breaking down and analyzing) and semantic analysis require more powerful models, such as **dependency parsers** or **neural networks**.

Finite Automaton – To model Formal Languages

A **finite automaton** is a very simple kind of machine used in computer science and math. It helps us solve problems by following a set of rules to decide if something is valid or not, step by step.

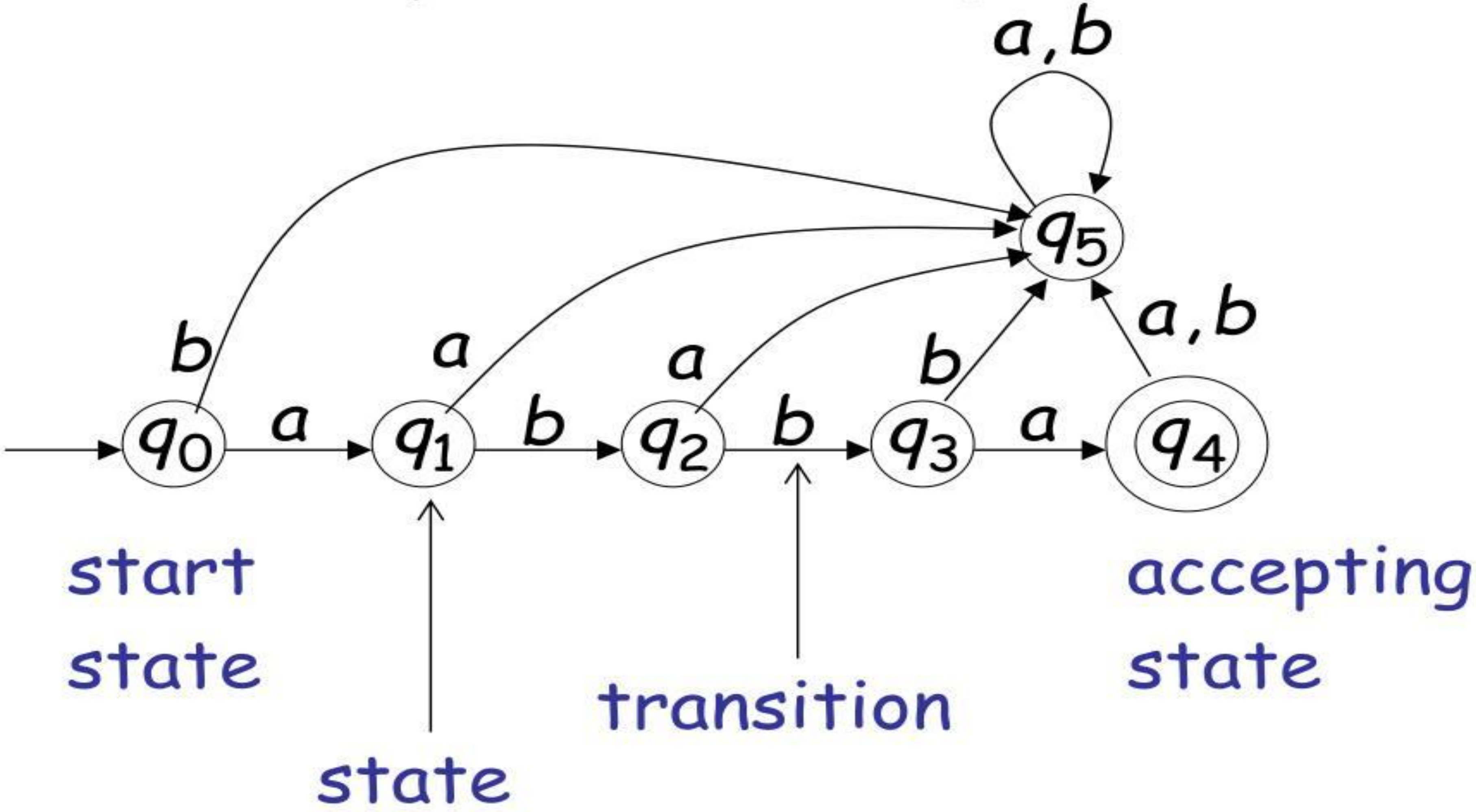
Think of a finite automaton as a **robot on a map** with some **rules** about where it can go. The robot:

1. Starts at a specific place (called the **start state**).
2. Moves from one place to another based on what it "reads" (these are the **inputs**).
3. Ends up in a place that tells us whether the journey was correct or not (called the **accept state** or **reject state**).

A Simple Example: Checking for the Word "cat"

1. The robot starts at the beginning (start state).
2. It reads each letter of a word one by one:
   1. If it sees "c," it moves to the next state.
   2. If it sees "a" after "c," it moves to the next state.
   3. If it sees "t" after "a," it moves to the final "accept" state.
3. If the robot successfully finishes at the "accept" state, the word is "cat." Otherwise, it rejects it.

One finite automaton



Finite-State Automata (FSA) in NLP

An **FSA** is a specific application of finite automata used to model linguistic phenomena in NLP. It simplifies certain language processes by treating them as a series of finite states.

Applications:

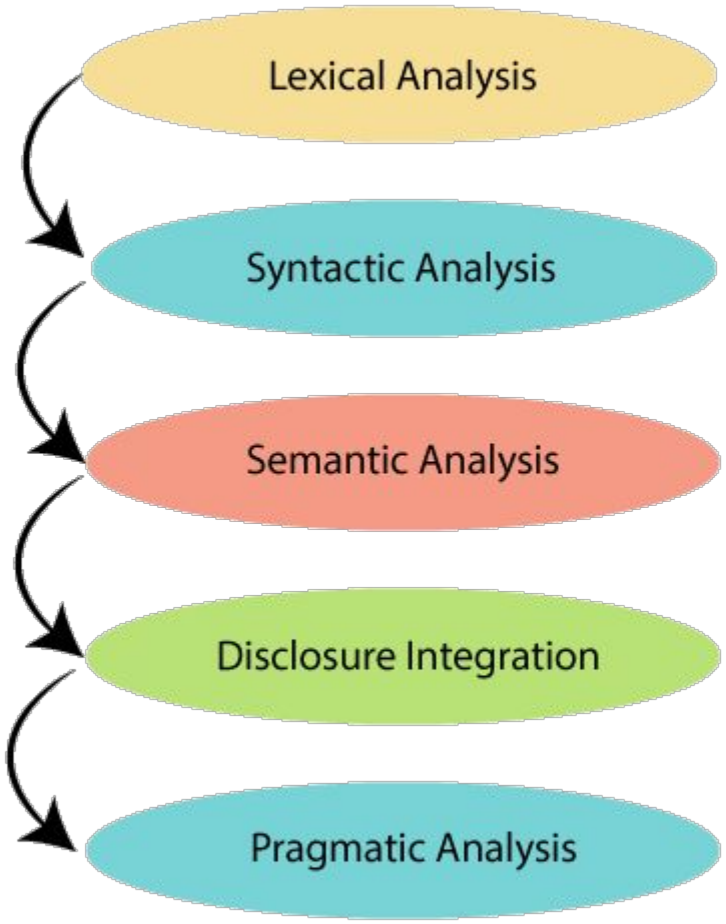
1. **Tokenization**: Recognizing words, punctuation, or other units in text.
   - Example: FSA recognizes sequences like dog, cat, or . based on defined states.
2. **Morphological Analysis**: Analyzing word forms and their inflections.
   - Example: Recognizing that "cats" consists of a root "cat" and suffix "s."
3. **Named Entity Recognition (NER)**: Identifying entities like names or dates in text.
4. **Spell Checking**: Recognizing valid word sequences.
5. **Regular Grammar Recognition**: Processing text using simple grammatical rules.

Limitation in NLP:

- FSAs can model regular languages but **cannot handle natural language's complexity** (e.g., nested dependencies or long-distance relations).
- Advanced techniques like **context-free grammars** or **neural networks** are often needed for complex tasks

Concept	**Purpose**	**Example for NLP**
Regular Expression	Text pattern matching	Tokenizing sentences, searching emails
Finite Automata	Recognizing regular languages	Simple text pattern recognition
FSA in NLP	Simplifying language tasks using FSAs	Tokenization, morphological analysis

Stages in NLP



Stages in NLP

Natural Language Processing (NLP) involves a series of stages that transform human language into a format that machines can understand and process. Here are the typical stages:

1. Text Preprocessing

- **Tokenization:** Breaking text into smaller units like words or sentences.
- **Stop word Removal:** Removing common words (e.g., "the," "is") that do not contribute significant meaning.
- **Stemming and Lemmatization:** Reducing words to their base or root forms.
- **Lowercasing:** Converting all text to lowercase for uniformity.
- **Text Cleaning:** Removing noise like punctuation, numbers, or special characters.

2. Lexical Analysis

- Identifying and analyzing the **structure of words**, including their **meanings and relationships**.
- Morphological analysis identifies **prefixes, suffixes, and root words**.

3. Syntactic Analysis (Parsing)

- Analyzing sentence structure **using grammar rules**.
- Identifying **parts of speech** and **dependencies between words**.
- Producing a syntax tree or dependency graph.

4. Semantic Analysis

- Understanding the meaning of individual words and sentences.
- Resolving **ambiguities (e.g., polysemy**) and **recognizing named entities (e.g., "John" as a person or "Apple" as a company).**
- Building representations like **word embeddings** (e.g., Word2Vec, GloVe).

5. Discourse Analysis

- Examining **relationships between sentences** in a text.
- Understanding the **context and structure of dialogue or** passages.

6. Pragmatic Analysis

- Understanding the **intended meaning or use of language in a given context**.
- Incorporating real-world knowledge, sarcasm, idioms, and cultural nuances.

7. Text Representation

- Converting text into numerical formats for machine processing:
  - **Bag of Words (BoW)**
  - **TF-IDF (Term Frequency-Inverse Document Frequency)**
  - **Word Embeddings (e.g., FastText, BERT)**

8. Modelling and Machine Learning

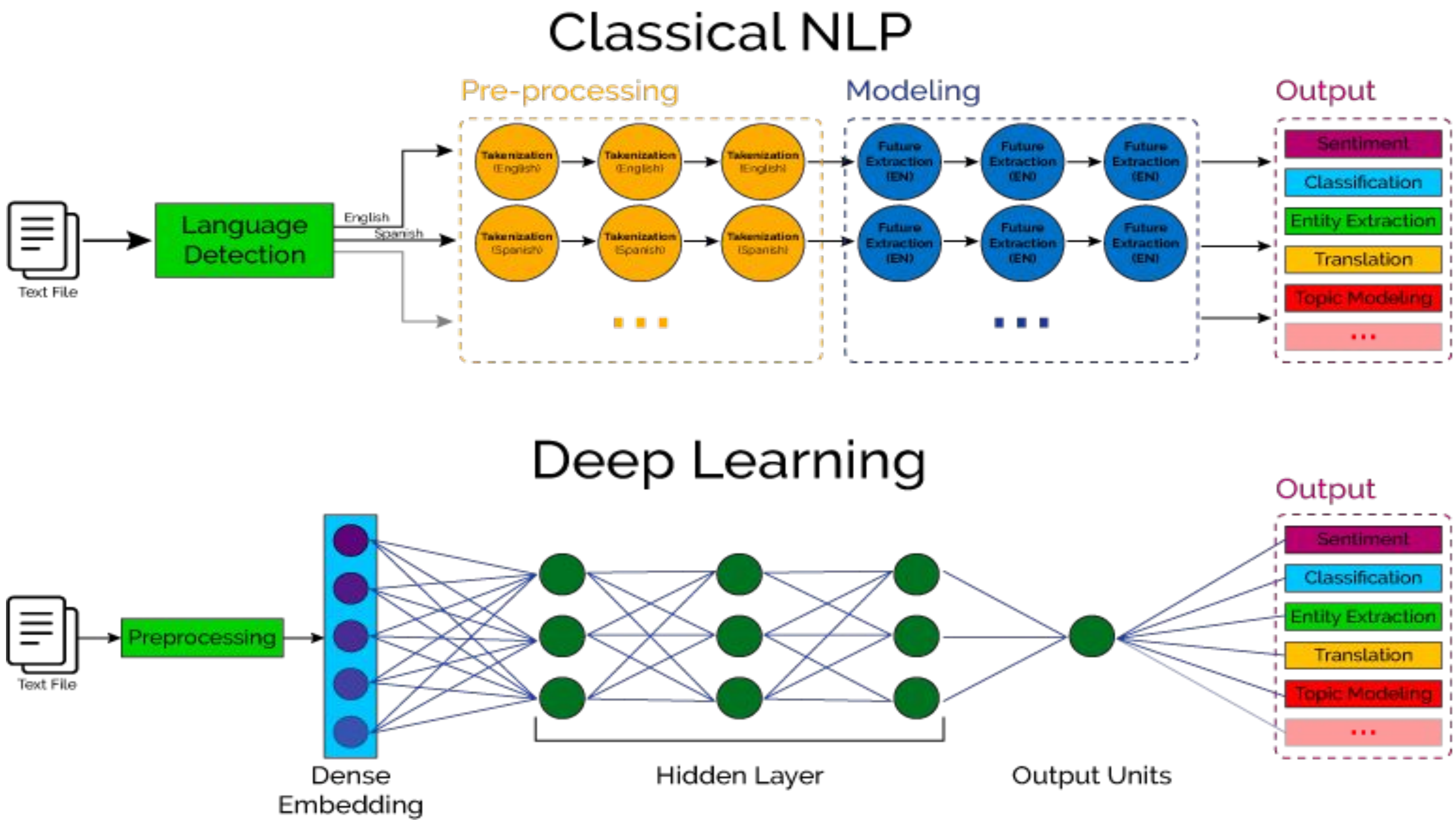
- Applying **algorithms to tasks like classification, translation, sentiment analysis**, etc.
- Using supervised, unsupervised, or reinforcement learning methods.

9. Evaluation

- Measuring model performance using **metrics like accuracy, precision, recall, F1 score, BLEU score** (for translation), etc.

10. Post-Processing

- Refining outputs (e.g., **fixing grammatical errors** in generated text).
- Formatting and presenting the final results for end-users.
- Each stage may involve different tools, techniques, and algorithms depending on the specific application (e.g., chatbots, sentiment analysis, machine translation).


Classical NLP
Pre-processing
Modeling
Output
Text File
Language Detection
English
Spanish
Takenization (English)
Takenization (English)
Takenization (English)
Takenization (Spanish)
Takenization (Spanish)
Takenization (Spanish)
Future Extraction (EN)
Future Extraction (EN)
Future Extraction (EN)
Future Extraction (EN)
Future Extraction (EN)
Future Extraction (EN)
Sentiment
Classification
Entity Extraction
Translation
Topic Modeling
...
Deep Learning
Output
Text File
Preprocessing
Dense Embedding
Hidden Layer
Output Units
Sentiment
Classification
Entity Extraction
Translation
Topic Modeling
...