# References and Links

[1] Scisharp STACK, “Data Science, Machine Learning, and AI Open Source Ecosystem for .NET”

<https://scisharp.github.io/SciSharp/>

[2] Catalyst nlp, “C# Natural Language Processing Library”

<https://github.com/curiosity-ai/catalyst>

# Terms

|  |  |
| --- | --- |
| **Term** | **Description** |
| Adverb | A word or phrase that modifies or qualifies an adjective, verb, or other adverb or a word group, expressing a relation of place, time, circumstance, manner, cause, degree, etc. (e.g., gently, quite, then, there). |
| Noun | A word (other than a pronoun) used to identify any of a class of people, places, or things (common noun), or to name a particular one of these (proper noun). |
| Pronoun | A word that can function by itself as a noun phrase and that refers either to the participants in the discourse (e.g., I, you) or to someone or something mentioned elsewhere in the discourse (e.g., she, it, this). |
| Verb | A word used to describe an action, state, or occurrence, and forming the main part of the predicate of a sentence, such as hear, become, happen. |
| Proper noun | A name used for an individual person, place, or organization, spelled with initial capital letters, e.g., Larry, Mexico, and Boston Red Sox. |
| LDA | Latent Dirichlet Allocation (LDA) is used for Topic Modeling to discover topics hidden (latent) in a set of text documents. |
| Corpus | In the context of artificial intelligence, a “corpus” (plural: “corpora”) refers to a large and structured set of texts or data. AI systems use these corpora for training, learning from the patterns, structures, and information contained within the data. |
| Corpora | Plural of corpus. |

# Annotated Glossary

Here are various Terms as gathered using “Bing Chat” and other sources for the convenience of our discussion.

## Information Extraction

Information Extraction (IE) is a subfield of Natural Language Processing (NLP) that involves the automated identification and extraction of structured information from unstructured or semi-structured text data. The goal of information extraction is to transform unstructured text data into structured data that can be easily analyzed, searched, and visualized.

For example, consider we’re going through a company’s financial information from a few documents. Usually, we search for some required information when the data is digital or manually check the same. But with information extraction NLP algorithms, we can automate the data extraction of all required information such as tables, company growth metrics, and other financial details from various kinds of documents (PDFs, Docs, Images etc.).

Information Extraction from text data can be achieved by leveraging Deep Learning and NLP techniques like Named Entity Recognition. However, if we build one from scratch, we should decide the algorithm considering the type of data we’re working on, such as invoices, medical reports, etc. This is to make sure the model is specific to a particular use case.

To understand the mechanics of Information Extraction NLP algorithms, we should understand the kind of data we are working on. This will help us to sort out the information we want to extract from the unstructured data.

## Named Entity Recognition (NER)

Named Entity Recognition (NER) is a subtask of Natural Language Processing (NLP) that involves identifying and categorizing named entities in unstructured text. Named entities are words or phrases that refer to specific types of objects, such as people, organizations, locations, dates, etc. NER is used in various NLP applications such as information retrieval, sentiment analysis, question-answering, and recommendation systems.

NER involves several steps, including tokenization, part-of-speech tagging, chunking, and entity recognition. Tokenization involves breaking down the input text into individual words or tokens. Part-of-speech tagging involves labeling each word in the text with its corresponding part of speech. Chunking involves grouping the words together into meaningful phrases based on their part of speech and syntactic structure. Finally, entity recognition involves identifying and classifying the named entities in the text.

There are several mathematical concepts involved in NER, including probability theory, machine learning, and deep learning. Hidden Markov Models (HMM) is a statistical model used for sequence classification tasks, such as NER.

## Part of Speech

In Natural Language Processing (NLP), identifying parts of speech is a fundamental task. The eight parts of speech in English are noun, pronoun, verb, adjective, adverb, preposition, conjunction, and interjection. Here is a brief description of each part of speech:

* Noun: A noun is a word that represents a person, place, thing, or idea. For example, “dog”, “house”, “love”, and “happiness” are all nouns.
* Pronoun: A pronoun is a word that takes the place of a noun. For example, “she”, “we”, “they”, and “it” are all pronouns.
* Verb: A verb is a word that expresses an action or state of being. For example, “run”, “sleep”, “is”, and “become” are all verbs.
* Adjective: An adjective is a word that describes or modifies a noun or pronoun. For example, “pretty”, “old”, “blue”, and “smart” are all adjectives.
* Adverb: An adverb is a word that describes or modifies a verb, adjective, or other adverb. For example, “quickly”, “very”, “barely”, and “happily” are all adverb.

To identify these parts of speech in NLP, one can use various techniques such as part-of-speech tagging. This technique involves labeling each word in a sentence with its corresponding part of speech. There are several libraries available in Python such as spaCy and NLTK that provide pre-trained models for part-of-speech tagging.

## Stemming

Stemming is the process of reducing inflected words to their word stem, base or root form—generally a written word form. It is a natural language processing technique that is used to reduce words to their base form, also known as the root form. The goal of stemming is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. Stemming is a technique used to extract the base form of the words by removing affixes from them. It is just like cutting down the branches of a tree to its stems. Search engines use stemming for indexing the words. Lemmatization is closely related to stemming. The difference is that a stemmer operates on a single word without knowledge of the context, and therefore cannot discriminate between words which have different meanings depending on part of speech. However, stemmers are typically easier to implement and run faster, and the reduced accuracy may not matter for some applications.

## Lemmatization

Lemmatization is a process in linguistics and computational linguistics where the inflected forms of a word are grouped together to be analyzed as a single item, identified by the word’s lemma, or dictionary form.

The goal of lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. Unlike stemming, which is a crude heuristic process that chops off the ends of words, lemmatization does things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma.

For example, the word “better” has “good” as its lemma. This link is missed by stemming, as it requires a dictionary look-up. The word “walk” is the base form for the word “walking”, and hence this is matched in both stemming and lemmatization. The word “meeting” can be either the base form of a noun or a form of a verb (“to meet”) depending on the context. Unlike stemming, lemmatization attempts to select the correct lemma depending on the context.

In computational linguistics, lemmatization is the algorithmic process of determining the lemma of a word based on its intended meaning. It depends on correctly identifying the intended part of speech and meaning of a word in a sentence, as well as within the larger context surrounding that sentence, such as neighboring sentences or even an entire document. As a result, developing efficient lemmatization algorithms is an open area of research.

### Is Lemmatization Context Sensitive?

Yes, lemmatization can be context-sensitive. While the basic form of lemmatization involves reducing a word to its base or dictionary form (lemma), some advanced lemmatization techniques take into account the context in which a word is used. This can be particularly helpful for words that have multiple meanings or forms depending on their usage. However, these context-sensitive approaches often require fully lemma-annotated sentences for training, which may not always be available, especially for low-resource languages.

## Topic Modeling

Topic modeling is a type of statistical modeling that uses unsupervised Machine Learning to identify clusters or groups of similar words within a body of text. It allows companies to harness the power of big data rather than be overwhelmed by it. Topic modeling is an unsupervised machine learning technique that’s capable of scanning a set of documents, detecting word and phrase patterns within them, and automatically clustering word groups and similar expressions that best characterize a set of documents. It refers to the process of logically selecting words that belong to a certain topic from within a document. From a business standpoint, topic modeling provides great time- and effort-saving benefits. For example, a company that wants to identify areas of improvement can run a survey asking users to rate their services and explain each rating. Topic modeling can fast-track this analysis by categorizing information into a topic like “most common reasons for low ratings”.

## Context Analysis

Context analysis in AI refers to the process of interpreting data in a specific context to provide meaningful and actionable insights. It involves understanding the nuances of language, user behavior, time, and place among other factors. Here are some key aspects:

1. Understanding Language: AI systems use Natural Language Processing (NLP) to understand the context of words and phrases in a sentence. This helps in understanding user queries, sentiment analysis, and language translation.
2. User Behavior Analysis: By analyzing the past behavior of a user, AI can provide personalized recommendations and predictions. For example, e-commerce platforms use this to recommend products based on browsing history.
3. Time and Place: The context of time and place is crucial in many applications. For example, a weather prediction AI needs to understand the geographical location and time to provide accurate forecasts.
4. Domain-Specific Context: In certain fields like healthcare or finance, AI needs to understand the specific jargon and norms of the field to provide accurate analysis and predictions.

By effectively analyzing context, AI systems can provide more accurate, personalized, and relevant results. However, it’s a challenging task due to the complexity and variability of data. AI researchers are continuously working on improving the contextual understanding of AI systems.

## Domain Specific Context

In the field of Artificial Intelligence (AI), domain-specific context refers to the application of specialized knowledge or expertise in a particular area or field. This can significantly enhance the performance and accuracy of AI models by providing them with a deeper understanding of the specific area they are designed to operate in.

For instance, in healthcare, an AI model might be trained to recognize patterns in medical imaging data. The “domain-specific context” in this case would include knowledge about human anatomy, the nature of different diseases and conditions, and how these are typically manifested in medical images. This contextual knowledge can help the AI model make more accurate diagnoses or predictions.

Similarly, in finance, an AI model might use domain-specific context about economic indicators, market trends, and financial regulations to make more accurate predictions about future market movements.

In summary, domain-specific context in AI involves tailoring AI models to specific fields or industries, leveraging specialized knowledge to improve their performance and accuracy. It’s about making AI more intelligent and useful in specific contexts.

## Applying Domain-Specific Context

Applying domain-specific context in Natural Language Processing (NLP) can be achieved through several methods:

1. Adding Domain-Specific Vocabulary to Existing Models: You can enrich the vocabulary of an already trained natural language model with that from a specialized domain (medicine, law, etc.) in order to perform new tasks (classification, NER, summary, translation, etc.). This can be done by adding new tokens to the vocabulary of an existing tokenizer like BERT WordPiece.
2. Fine-Tuning on a New Corpus: An intermediate method consists in fine-tuning on a new corpus an already trained natural language model by adding to its existing tokenizer vocabulary that specific to the domain of this new corpus.
3. Creating Domain-Specific Models: One of the lines of research is creating domain-specific models for a specific universe, for a diverse set of downstream tasks, like sentiment analysis, NER, etc. This involves creating a BERT-based language model using your own corpus of data.
4. Domain Adaptation of Language Models: An alternative approach for transferring pretrained language models to new domains is by adapting their tokenizers. Domain-specific subword sequences can be efficiently determined directly from divergences in the conditional token distributions of the base and domain-specific corpora.

Remember, the choice of method depends on your specific use case and the resources available to you. It’s always a good idea to experiment with different approaches and see which one works best for your needs.

## Latent Semantic Indexing

Latent Semantic Indexing (LSI) is a technique that uses singular value decomposition (SVD) to analyze the relationships between terms and concepts in unstructured data. It is a natural language processing (NLP) approach that was created in the 1980s. It helps with categorizing and information retrieval by projecting queries and documents into a lower-dimensional space with latent semantic dimensions. In this space, a query and a document can have high similarity even if they do not share any terms, as long as their terms are semantically similar. LSI is used by search engines to provide more accurate and relevant results by understanding the context and synonyms of words.

## AI Inference Methods

In AI, inference methods are techniques for using existing knowledge or models to make predictions, classifications, or recommendations on new data. There are different types of inference methods, such as deductive, inductive, abductive, probabilistic, and causal inference. Here is a brief overview of each type:

* Deductive inference involves reasoning from general principles to specific conclusions, based on logical rules. For example, if A implies B and A is true, then B must also be true. This is also known as modus ponens1.
* Inductive inference involves inferring general principles or rules based on specific observations or data, using statistical methods or heuristics. For example, if all the observed birds can fly, then one may infer that all birds can fly. This is also known as generalization or learning from examples.
* Abductive inference involves finding the best explanation for a set of observations, based on prior knowledge or hypotheses. For example, if one observes smoke, then one may infer that there is fire. This is also known as inference to the best explanation or diagnosis.
* Probabilistic inference involves computing the likelihood or probability of an event or a hypothesis, given some evidence or data, using probability theory or Bayesian networks. For example, if one knows the probability of rain and the probability of clouds, then one can infer the probability of rain given clouds. This is also known as inference under uncertainty or belief updating.
* Causal inference involves identifying the causal relationships or effects between variables, given some data and assumptions, using graphical models or causal calculus. For example, if one knows the causal structure of a system and the interventions on some variables, then one can infer the effects of those interventions on other variables. This is also known as inference of causation or causal reasoning.

If you want to learn more about inference methods in AI, you can check out these resources:

* What is AI inferencing?: An IBM Research blog post that explains what inference is, how it differs from training, and how to speed up inference performance.
* What is Machine Learning Inference?: A DataCamp blog post that introduces inference approaches, benefits, challenges, and applications in machine learning.
* Rules of Inference in AI: A Scaler Topics page that lists and explains some common deductive inference rules in AI.
* Inference Engine in AI: A Scaler Topics page that describes how an inference engine works and the two modes of inference: forward chaining and backward chaining.

## Guan (Prolog like language)

Guan (to observe in Mandarin) is a general-purpose logic programming system written in C# and built as a .NET Standard Library. It has been tested in both Windows and Linux environments.

Guan employs Prolog style syntax for writing logic rules. It enables easy interop between such rules with regular C# code and the vast .NET Base Class Library. External Predicates are written in C# and logic rules can be housed in simple text files or as string variables in your consuming program. These logic rules will be parsed and executed by Guan, which provides imperative, procedural, and even functional programming idioms the expressive power of logic programming for use in several novel contexts.

## Sentence Transformers / Sentence Similarity

Sentence Transformers are a type of deep learning model that can be used to compare the semantic similarity between two sentences. They convert input texts into vectors (embeddings) that capture semantic information and calculate how close (similar) they are between them.