

Natural-language processing (NLP) and indexing, Part 1. Overview

Donald Howes

This article discusses the application of natural language processing (NLP) in book indexing, highlighting its potential to assist in extracting information from unstructured text. It explains key NLP concepts such as topic modeling, semantic search, and text representation techniques like bag-of-words and TF-IDF. While NLP shows promise, current systems are not yet capable of producing high-quality indexes, and further research is needed to explore its full applicability to indexing.

The text discusses the fundamental components and types of neural networks, including neurons, input weights, biases, and activation functions. It explains the structure of neural networks, comprising input, hidden, and output layers, and highlights the perceptron as an early model. The text also covers various activation functions like sigmoid, tanh, ReLU, and softmax, and compares shallow and deep learning networks, noting their characteristics, advantages, and limitations. Additionally, it introduces popular deep learning models such as RNNs, LSTMs, transformers, BERT, and GPT, emphasizing their applications in NLP tasks.

Every model is wrong, but some are useful
George Box, statistician (Box, 1976)

Introduction

Interest by book indexers in artificial intelligence (AI) in general, and natural-language processing (NLP) in particular, has spiked over the past few years.¹ Many are wondering just how this technology can assist in the production of a back-of-book index, and there are increasing worries about AI systems replacing human book indexers altogether in index creation.

Donald Howes has an academic background in archaeology. He worked as a software engineer for companies large and small and has owned and operated an online antiquarian bookstore. He currently works as a freelance indexer, specializing in back-of-book and embedded indexes for scholarly and trade publications. He is a member of the Indexing Society of Canada/Société canadienne d'indexation (ISC/SCI), and of the American Society for Indexing (ASI).
Email: dwhowes@shaw.ca

At present, NLP applications show the most promise as tools for the extraction of information from the unstructured text of a manuscript.² Future articles in this series will explore two areas where NLP can assist the indexer: topic modeling³ and semantic search.⁴

This first article provides an overview of natural-language processing. It explains how NLP works and examines the tasks that fall under NLP and its recognized subset, natural-language understanding (NLU). The benefits and disadvantages of NLP are discussed, including chatbot hallucinations, and the applicability of NLP/NLU tasks to indexing is considered.

What is natural-language processing?

NLP is a subset of artificial intelligence that uses techniques of machine learning (ML) to enable computers and devices to understand and generate human language.⁵ It does this by combining computational linguistics, statistical modeling, shallow learning, and deep learning techniques (Stryker, 2024). See the Appendix for more about these topics.

NLP has the following benefits:

- **Automation of repetitive tasks:** NLP has been usefully applied to such areas as customer support, data entry, and document handling.
- **Improved data analysis:** NLP enables the extraction of insights from unstructured data (customer reviews, social media posts, news articles, and so on), performing what is known as sentiment analysis.
- **Enhanced search:** NLP can enhance search engines by allowing them to understand the intent of a user's query through the analysis of words and phrases.
- **Content generation:** large language model systems can generate text in response to a user request.

It is important to remember that NLP applications do not work with language per se, they work with vectors (Jha, 2023).⁶ In order to create reliable vectors, applications use a pipeline approach to cleaning and standardizing the input text data (Ali, 2023; Premadasa, 2024). NLP pipelines will be covered in detail in Part 2 of this series, which is on topic modeling.

The foundation of all modern NLP applications is the neuron and the neural network.⁷ These models, often statistical in nature, are abstractions that attempt to understand and interpret their real-world inputs. A degree of error is unavoidable, and expected, in even the best-trained model. Readers interested in understanding these models better can dip their toes into the mathematics underlying these models in the Appendix. While some mathematical fluency is required to understand the explanation, it is not overly complex.

Data preprocessing

When dealing with unstructured text, a number of preprocessing steps must be undertaken in order to convert the text into a regularized form suitable for vectorization.⁸ For the example text used here (below), these steps include:

- 1 removal of specialized characters, URLs, HTML, and leading or trailing white space
- 2 tokenization⁹
- 3 conversion to lower case and deletion of punctuation
- 4 lemmatization¹⁰
- 5 deletion of stop words¹¹
- 6 deletion of unneeded parts of speech.

To exemplify this process, the text used is the initial paragraph from the introduction to *Who owns outer space?* (Byers and Boley, 2023: 1):

The asteroid 101955 Benu is just a pile of rubble, weakly held together by its own gravity, the remnants of a catastrophic event that occurred a billion years ago. But Benu is also a bearer of both life and death, containing clues about the origins of life on Earth while, at the same time, having the potential to destroy humanity. For over time, the agencies of physics and chance have brought the 500-metre-wide asteroid onto an orbit very near to Earth.

After steps 1–3, the text is as follows:

the asteroid 101955 benu is just a pile of rubble weakly held together by its own gravity the remnants of a catastrophic event that occurred a billion years ago but benu is also a bearer of both life and death containing clues about the origins of life on earth while at the same time having the potential to destroy humanity for over time the agencies of physics and chance have brought the 500 metre wide asteroid onto an orbit very near to earth

After lemmatization and stop word removal (steps 4–5), the text is:

asteroid 101955 benu pile rubble weakly hold gravity remnant catastrophic event occur billion year ago benu bearer life death contain clue origin life earth time potential destroy humanity time agency physics chance bring 500 metre wide asteroid orbit near earth

Finally, the text is trimmed to show only selected parts of speech. In this case, proper nouns, nouns, verbs, adjectives, and adverbs:

asteroid benu pile rubble weakly hold gravity remnant catastrophic event occur year ago benu bearer life death contain clue origin life earth time potential destroy humanity time agency physics chance bring metre wide asteroid orbit earth

While no longer particularly readable, the text is in a form that can be easily converted into a vector.

Text representation techniques

Now that we have processed the text, what techniques are used to convert it into a machine-readable form?¹² Note that the following discussions comprise only an introduction to each technique. They will be addressed in more detail in Part 2 of this series, dealing with topic modeling.

Bag-of-words (BoW)

The BoW model works by creating a vocabulary of all words represented in the documents being analyzed,¹³ then takes each word in the documents as a separate feature, counting the number of times that word appears. The BoW model disregards grammar and word order. Documents can be compared by word occurrence and frequency.¹⁴

For example, with these documents:

Doc 1: 'I love dogs and cats'

Doc 2: 'I hate dogs but love cats'

Doc 3: 'Dogs are my favorite animal'

the vocabulary is:

['I', 'love', 'dogs', 'and', 'cats', 'hate', 'but', 'are', 'my', 'favorite', 'animal'].

In practice, a number of these words ('I', 'and', 'but', 'are', 'my') would be removed during preprocessing, so the BoW vector matrix for these documents would be as shown in Table 1.

Table 1. BoW vector matrix representation

	love	dogs	cats	hate	favorite	animal
Doc 1	1	1	1	0	0	0
Doc 2	1	0	1	1	0	0
Doc 3	0	1	0	0	1	1

Advantages

- It is simple to implement.
- It works well when word frequency is the main classifier.

Disadvantages

- Since word order is ignored, information regarding the structure and meaning of a document is lost.
- For large vocabularies the resulting vectors can become very large and sparse (each vector contains many zeros).

Term frequency-inverse document frequency (TF-IDF)

TF-IDF is a measure of the importance of a word within a collection of documents. It is useful for such tasks as information retrieval and keyword extraction.

TF-IDF is calculated by:

$$\text{TF-IDF} = \text{TF} \cdot \text{IDF}$$

where

TF = the frequency of a word in a document.

IDF = the inverse frequency of a word in the collection of documents. The higher the occurrence of a word, the lower the IDF score. IDF is calculated by

$$\log \left(\frac{\sum \text{documents}}{\sum \text{documents with term}} \right).$$

For example, with the documents:

Doc 1: 'The cat jumped'

Doc 2: 'The white tiger roared'

Doc 3: 'Bird flying in the sky'

the TF-IDF matrix is as shown in Table 2.

Table 2. Matrix of TD-IDF vectors

	bird	cat	flying	in	jumped	roared	sky	the	tiger	white
Doc 1	0	0.5844	0	0	0.5844	0	0	0.3452	0	0
Doc 2	0	0	0	0	0	0.5464	0	0.3227	0.5464	0.5464
Doc 3	0.5046	0	0.5046	0.3838	0	0	0.5046	0.2980	0	0

Advantages

- It is simple and easy to use.
- It provides a clear starting point for similarity analysis.

Disadvantages

- It cannot assist in the analysis of semantic meaning.

- It disregards word order, so it cannot distinguish compound nouns or instances where word order is important (for example, ‘friendly’ and its negation ‘not friendly’).
- It can suffer from the same problem of dimensionality and sparse vectors as BoW.

Text embedding

Text embeddings are dense, low-dimensionality vector representations that capture semantic relationships and contextual information (Van Otten, 2023e; Yadav, 2024b). These techniques are central to deep learning models in general, and LLMs in particular. For an overview of embedding techniques, see Palachy (2020). See the Appendix for more about deep learning models.

Embedding can be done at one of three levels – word, sentence, or document.¹⁵ Table 3 gives the advantages and disadvantages of each of these types of embedding.

Table 3. Text embedding advantages and disadvantages

Type	Advantages	Disadvantages
Word	Efficient representation of words in isolation. Captures the meaning and relationship between words.	Struggles to understand polysemy. Has word-level granularity and cannot capture the meaning of longer phrases or sentences.
Sentence	Can capture the context of words in a sentence. Can handle polysemy. Can leverage transfer learning.	Computationally expensive. Memory-intensive.
Document	Captures the semantic meaning of the document. Can handle unseen words by leveraging the context of their appearance in the set of documents. Scalable to very large data sets. Flexible and easily customizable.	Computationally expensive. Requires a large data set.

Sources: GeeksForGeeks (2025a), Van Otten (2023e), Yadav (2024b)

- **Word embedding.** Word embeddings represent individual words as vectors in a continuous multidimensional vector space. They capture the syntactic and semantic relationships between words and, since each word is a point in the vector space, the distance between words has meaning. Common models for word embedding are Word2Vec, GloVe, and FastText.

- **Sentence embedding.** Sentence embeddings represent a whole sentence as a vector, summarizing the meaning of the sentence in vector form by taking into account word order and the structure of the sentence. Common models for sentence embedding are USE (Universal Sentence Encoder), BERT, and SBERT (Sentence-BERT).
- **Document embedding.** Document embeddings represent whole documents as fixed-length vectors in a continuous vector space, summarizing the document as a point in that space. Models for document embedding are PV-DM (Paragraph Vector – Distributed Memory) and PV-DBOW (Paragraph Vector – Distributed Bag of Words).

NLP categories

NLP covers a number of distinct task categories (Figure 1), usually with multiple tasks included in a production NLP application. A generally recognized subset of NLP tasks has to do with natural-language understanding. Applications such as ChatGPT fall under NLU.

Named-entity recognition (NER)

NER extracts important information from unstructured text without the need for human intervention. Named entities refer to specific things in the real world (Van Otten, 2022b; Hashemi-Pour and Barney, 2024). Table 4 lists the default named entities that can be identified by SpaCy.¹⁶

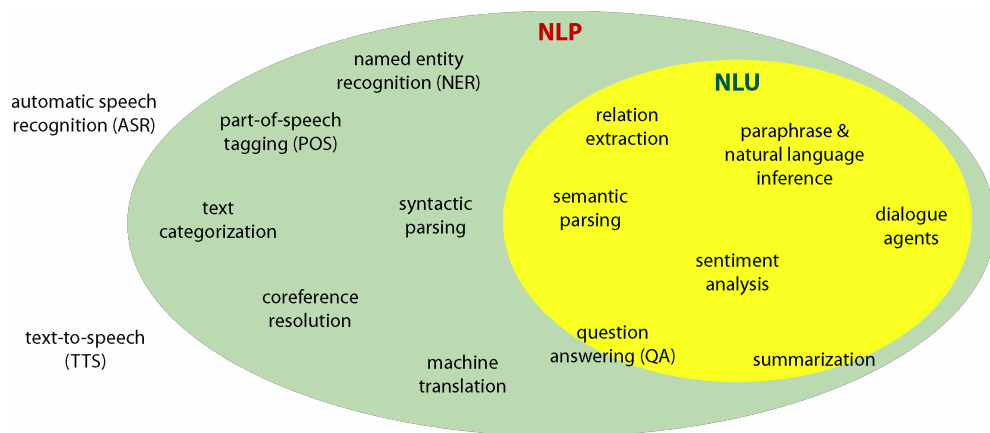


Figure 1. Natural-language processing categories

Table 4. Default SpaCy named entities

Named entity labels	Description
CARDINAL	Numerals that do not fall under another type
DATE	Absolute or relative dates or periods
EVENT	Named hurricanes, battles, wars, sports events, etc.
FAC	Buildings, airports, highways, bridges, etc.
GPE	Countries, cities, states
LANGUAGE	Any named language
LAW	Named documents made into laws
LOC	Non-GPE locations, mountain ranges, bodies of water
MONEY	Monetary values, including units
NORPs	Nationalities or religious or political groups
ORDINAL	'first,' 'second,' etc.
ORG	Companies, agencies, institutions, etc.
PERCENT	Percentage, including '%'
PERSON	People, including fictional
PRODUCT	Objects, vehicles, foods, etc. (not services)
QUANTITY	Measurements, as of weight or distance
TIME	Times smaller than a day
WORK_OF_ART	Titles of books, songs, etc.

Generally, the accuracy of NER increases with the size/complexity of the model used. In situations where domain-specific terms are found, NER can struggle with identification either by not tagging the term or by misidentifying it. These problems can be reduced by retraining the model using domain-specific texts.

Part-of-speech tagging (POS)

POS is one of the fundamental NLP tasks, where each word in a sentence is assigned to a grammatical category (Figure 2). The goal of POS is to understand the syntactical structure of a sentence and identify the grammatical role of individual words (Mudadla, 2023).

Text categorization

Also known as text classification, categorization is used to turn unstructured data into structured data. It does this by assigning the unstructured text to predefined categories. Once categorized, information can be extracted from the structured data and used in such NLP tasks as sentiment analysis, spam detection, and topic labeling (Van Otten, 2022g).

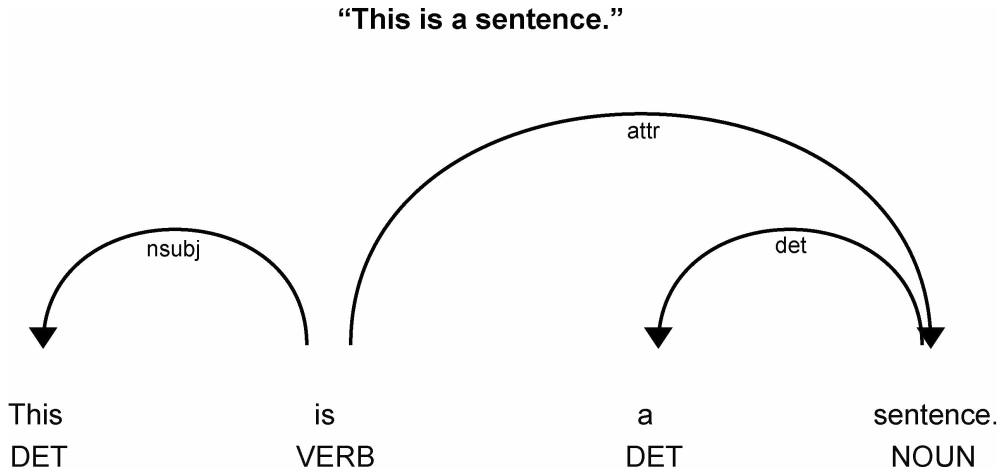


Figure 2. SpaCy POS and dependency tagging of a simple sentence. DET = determiner

Syntactic parsing

Syntactic parsing is a computational linguistic technique that performs the automatic analysis of the syntactic structure of a text. The structure can be displayed visually as a dependency graph (Figure 2).¹⁷ Syntactic parsing is a fundamental step in such NLP tasks as sentence parsing, ambiguity resolution,¹⁸ grammar and language generation, and information extraction (Van Otten, 2023f; Al-Ma’arif, 2024a).

Coreference resolution

Coreference resolution (Al-Ma’arif, 2024b; Van Otten, 2024d) ‘is the task of finding all linguistic expressions (called mentions) in a given text that refer to the same real-world entity. After finding and grouping these mentions we can resolve them by replacing ... pronouns with noun phrases’ (Maślankowska and Mielniczuk, 2022). Coreference resolution provides unambiguous sentences for use in other NLP tasks (Figure 3). Coreference resolution can be applied to such NLP tasks as text understanding, information extraction, machine translation, sentiment analysis, and document summarization.

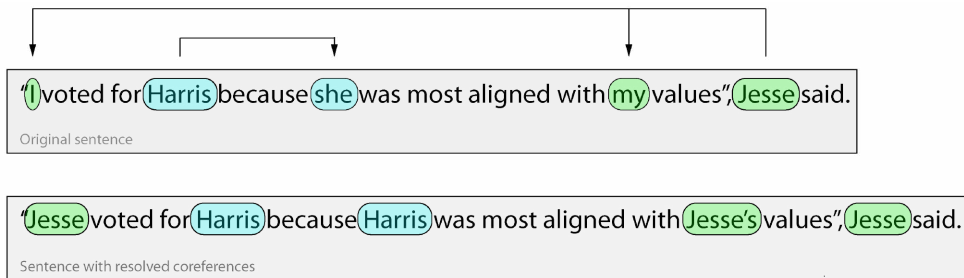


Figure 3. Example of coreference resolution

Machine translation

Machine translation (MT) ‘is the task of automatically converting one natural language into another, preserving the meaning of the input text, and producing fluent text in the output language’ (Stanford Natural Language Processing Group, n.d.). The output of an MT application can be either text or spoken language. MT is one of the oldest areas of study in AI, but the change from rules-based systems to deep learning neural networks has significantly increased the quality of the translation (Van Otten, 2022d, 2023m; GeeksForGeeks, 2024d). MT has applications in diverse areas, including cross-border communication, localization, business, education, and government. See the Appendix for more about deep learning.

NLU categories

NLU tasks generally use many of the previously described NLP tasks to produce their output. The NLU subset of NLP provides tools that are of interest to indexers.

Relation extraction

Relation extraction¹⁹ ‘automatically identifies and categorizes the connections between entities within natural language text. These entities can encompass individuals, organizations, locations, dates, or any other nouns or concepts mentioned in the text’ (GeeksForGeeks, 2024f). It allows the extraction of structured data from unstructured text. Figure 4 is an example of binary relation extraction from a simple sentence using NER. There are a number of other types of extraction that can resolve more nuanced relationships (Zhao et al., 2023; GeeksForGeeks, 2024f). Relation extraction is one of a class of techniques used to extract information from unstructured documents (Van Otten, 2023l).

Paraphrase generation

‘Paraphrase generation is the task of generating an output sentence that preserves the meaning of the input sentence with variations in word choice and grammar’ (Yse, 2022). Paraphrasing is used as an aid to comprehension, for knowledge stimulation, or skills development. Paraphrasing was originally done using rules-based or machine-learning systems.²⁰ Modern paraphrase systems use transformer-based

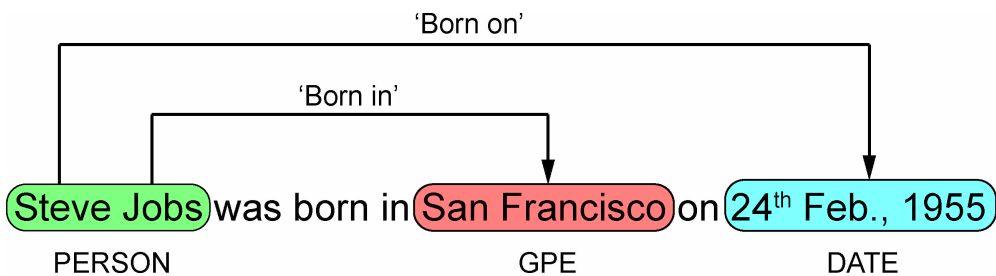


Figure 4. Binary relation extraction using NER

models such as GPT, BERT, or PEGASUS (Mathur, 2023). For more about BERT and GPT, see the Appendix.

Natural-language inference (NLI)

NLI is a fundamental NLP task that determines the logical relationship between two texts. That relationship can be one of three categories: entailment, contradiction, or neutral (Table 5). NLI is used in the understanding of textual relationships, QA chatbots, sentiment analysis, and machine translation.

Table 5. NLI relationships

Relationship	Description	Example
Entailment	The meaning of the hypothesis can be inferred or logically deduced from the premise	Premise: The cat is on the table Hypothesis: The cat is on a piece of furniture
Contradiction	The meaning of the hypothesis contradicts or is in opposition to the premise	Premise: The sun rises in the east Hypothesis: The sun never rises
Neutral	The relationship between the premise and the hypothesis is neither entailment or contradiction	Premise: I like to eat ice cream Hypothesis: Eating ice cream is enjoyable

Source: data from Shariq (2023)

Semantic parsing

Semantic parsing converts natural language into a logical form or structured query that is a machine-compatible representation of its meaning (Al-Ma'arif, 2024a). Rules-based or statistical systems were commonly used to perform the conversion, but modern semantic parsing components use deep learning models such as recursive neural networks (RNNs) or transformer-based models. These systems are better able to deal with issues of language ambiguity and context dependency. Applications that benefit from semantic parsing are machine translation, QA systems, automated reasoning, and code generation. See the Appendix for more about these deep learning models.

Sentiment analysis

'Sentiment analysis uses natural-language processing and machine-learning techniques to identify and extract "feeling" information from text. Sentiment information can be positive, negative, or neutral and is often used to gauge a text's overall emotion' (Van Otten, 2022c).²¹ Because sentiment analysis requires an understanding of the context and meaning of a text, it is typically a supervised learning task.²² Texts are labeled as to their overall sentiment (positive, negative, neutral). Sentiment analysis is used in such areas as customer feedback for a product or service, or the analysis of social-media discussions.

Dialogue agents

‘Dialogue systems, or conversational agents, are programs that allow a person and a computer to communicate in a way that is similar to a conversation between two people’ (Hyperskill, n.d.). Dialogue agents are divided into two types:

- Task-oriented: these are agents designed for a specific task, such as hotel bookings.
- Chatbot: a computer program that can simulate human speech patterns and engage in a conversation with a user. They use deep learning models in order to understand questions and frame an appropriate response. Most chatbots fall into the categories of declarative bots or predictive bots.²³

Figure 5 shows the general structure of user interaction with a dialogue agent. Chatbots are created using transformer-based generative language models such as GPT (Yücel, 2023). See the Appendix for more about transformer-based deep learning models.

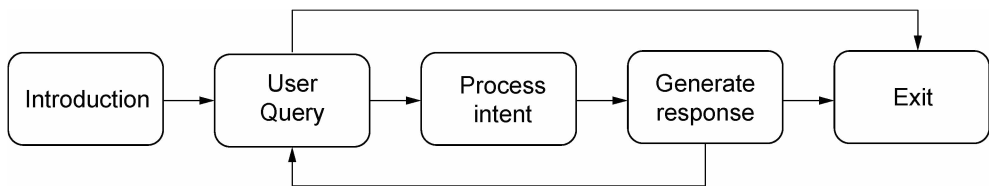


Figure 5. General structure of a dialogue agent session

Question answering (QA)

QA systems have the ability to understand and answer queries posed to them in natural language (Van Otten, 2023j). In general, a QA system includes the following steps:

- Pre-processing: the query is cleaned (tokenization, lemmatization, stop word removal, etc.) and converted to a standardized form.
- Question understanding: the standardized query is analyzed to extract entities, concepts, and type of query. The QA system uses NLP tasks such as NER, dependency parsing, and POS tagging to accomplish this.
- Information retrieval: the transformed query is used to search the QA data set to find relevant information. Information retrieval techniques such as keyword or semantic search are used for retrieval.
- Answer generation: the information retrieved by the system is analyzed to construct a set of answers to the query.
- Ranking: the answer set is ranked based on relevance and confidence scores.

QA systems are created using transformer-based extractive language models such as BERT or a BERT-derived model (Yücel, 2023). See the Appendix for more about transformer-based deep learning models.

Summarization

Text summarization techniques process a document's text and provide a concise, clear summary of its contents (Van Otten, 2022e). There are two main methods for summarizing text:

- **Extractive summarization:** a summary is created using the most pertinent key phrases from the source document. Document integrity is maintained during summarization.
- **Abstractive summarization:** a summary is created using a new set of sentences, not present in the source document, for the summary. Abstractive summarization synthesizes the essential information of the document, as in the abstract generated for this article.

Prior indexer interest in NLP

There has been limited interest over the years in the application of NLP to book indexing. However, in the last few years, with the development of chatbot technology, this interest has seen a significant upswing.

Articles in The Indexer

In April 1990, Korycinski and Newell published an article in *The Indexer* in which they examined the syntactic and semantic requirements for the automatic generation of an index (Korycinski and Newell, 1990). In the following issue, Jones (1990) pointed out that their analysis focused primarily on the form of index creation and ignored the mechanisms that human indexers use to derive an index entry (something that large language model applications struggle with to this day).

Both these articles are firmly rooted in the assumption of a rules-based expert system performing index creation. Rules-based systems remained the norm for decades and still have specific uses. It was not until well into this century that deep learning neural networks replaced rules-based systems for NLP tasks.

Most recently, Tanya Izzard (2024) evaluated the use of LLM chatbots (specifically Claude and Adobe AI Assistant) in the performance of indexing tasks, up to and including the creation of sample indexes. She found that there were issues regarding intellectual property, algorithmic judgment, accuracy, and reliability (in terms of the repeatability of output) that currently weigh against the use of these tools in indexing.²⁴

Online discussions

Prior to 2023, the oldest online thread dealing with any topics related to artificial intelligence appears to be from 2015. That discussion centred on indexing an unstructured document (Rooney, 2015). There is no further discussion of AI until 2023, but during 2023 and 2024 it became a recurring topic in the IDG discussion list:²⁵

- the application of AI to publishing (Rising, 2023; Connolly, 2024; Moffat, 2024)
- automated indexing using large language models (LLMs) (Peace, 2023; Reed, 2023; Saigh, 2023; Ullstrom, 2023; Unger, 2024)
- controlled vocabularies and LLM hallucinations (Browne, 2024)
- training AI systems to create an index (Ching, 2023)
- text summarization using Adobe AI Assistant (Modi, 2024; Winston, 2024).

In December 2024, the American Society for Indexing (ASI) conducted a special event online entitled ‘All about Indexing: From Thesauri to AI, Tools and Tips for Working Smarter,’ which was summarized in the January 2025 issue of the online ASI newsletter *See Also*.²⁶ In that seminar, the keynote address examined AI in the publishing industry,²⁷ and one of the sessions demonstrated the use of TExtract²⁸ in semi-automated index creation.

Chatbot hallucinations

Among the real-world hazards of LLM models are hallucinations, which Izzard (2024: 384, 398) briefly touched on. This issue is addressed in greater detail here since it is a particular hazard when using LLM models for indexing.

Hallucinations in QA systems

A hallucination is when the QA system returns a response to a query that is both incorrect and plausible (Huang et al., 2024). Hallucinations (Figure 6) can be divided into factual (Li et al., 2024) and non-factual (Yu et al., 2024).

Factual hallucinations

Factual hallucinations occur when the QA system produces factually incorrect content. Table 6 details factual hallucinations types.

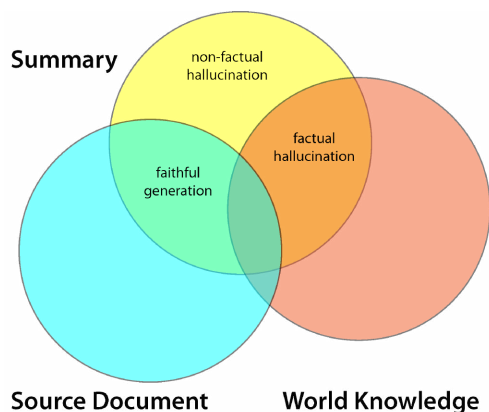


Figure 6. Hallucinations in GPT QA systems. Source: original figure by Yuxuan Wang, copyright © 2023. Released under MIT license, used with permission

Table 6. Types of factual hallucination produced by LLMs

Type of factual hallucination	Explanation
Factual inconsistency	The LLM provides a response that is inconsistent with the verifiable truth
Factual fabrication	The LLM response is entirely fictional

Source: table abstracted from Shah (2023)

Non-factual hallucinations

Non-factual hallucinations (also known as faithfulness hallucinations) occur when the LLM produces unfaithful content, or content that is inconsistent with the source material. Table 7 gives the types of non-factual hallucination.

Table 7. Types of non-factual hallucination produced by LLMs

Type of non-factual hallucination	Explanation
Instruction inconsistency	The LLM ignores specific instructions given by the user in their query
Context inconsistency	The LLM response includes information not present in the provided context, or information that contradicts it
Logical inconsistency	The LLM response includes a logical error, despite starting correctly

Source: table abstracted from Shah (2023)

Consequences

The fact that generative QA systems like ChatGPT can hallucinate has real-world consequences when the user does not understand the limitations of the system. An example of this comes from Vancouver, BC, where a lawyer in a child custody case used ChatGPT as a research tool when constructing her brief. Not until opposing

council requested that she provide them with the case law cited in her brief was it discovered that the cited cases did not exist. ChatGPT had constructed them out of thin air (Proctor, 2024) and the lawyer had not verified their existence. The lawyer was publicly reprimanded by the judge and is facing an investigation by the Law Society of British Columbia (Lazaruk, 2024). A similar case occurred in Manhattan in 2023 (Parnaby, 2023).

Hallucinations by ChatGPT when it is applied to the legal field are alarmingly common. A 2024 study (Dahl et al., 2024) shows that ChatGPT hallucinates minimally 58 percent of the time when answering legal questions. Efforts to study how to mitigate hallucinations are active in other fields as well (Kang and Liu, 2023; Shah, 2024). Hallucinations remain a significant problem for all large language model generative QA systems (Ji et al., 2023; Snyder et al., 2024).

Conclusion

This overview of machine learning and natural-language processing has aimed to provide the reader with an increased understanding of the mechanics underlying NLP applications, and a greater appreciation of their strengths and weaknesses.

While NLP applications show promise as tools for the extraction of information from the unstructured text of a manuscript, current transformer-based systems are not yet able to produce a valid, high-quality index, as demonstrated by Izzard (2024). However, since index creation is a specialized process, this is possibly due to no transformer model having been trained in index creation, rather than any incapability of the model.

In Part 2, we will examine topic modeling, using both the traditional methods of LDA (latent Dirichlet allocation) and TF-IDF, and a transformer-based approach using BERTopic. Part 3 will examine the area of semantic search using several different models. Semantic search tools allow the indexer to enter natural-language queries and extract content from the manuscript that matches the concept(s) expressed in the query.

The author welcomes reader input on other NLP areas that they would like to see investigated for their applicability to indexing.

Acknowledgements

I would like to thank Alison Craig, who read and commented on earlier versions of this article. If the reader wishes to further explore the fields of machine learning and natural-language processing, I recommend two excellent resources, both of which I have drawn on extensively. They are the section of the GeeksforGeeks portal on machine learning (<https://www.geeksforgeeks.org/machine-learning/>), and the blog by Neri van Otten (<https://spotintelligence.com/blog/>) at Spot Intelligence. Any remaining errors are my own.

Appendix: the neuron and the neural network

Neurons

A neuron is the basic unit of a neural network (Agrawal, 2019) and is composed of two parts (Figure A1).²⁹ First is a summation function that uses the neuron inputs,³⁰ input weights,³¹ and the bias value³² to calculate the summed input value of the neuron. Second is an activation function³³ that determines the output value of the neuron. Changes to the input weights can have a dramatic effect on output. Neural networks are trained by modifying these values.³⁴

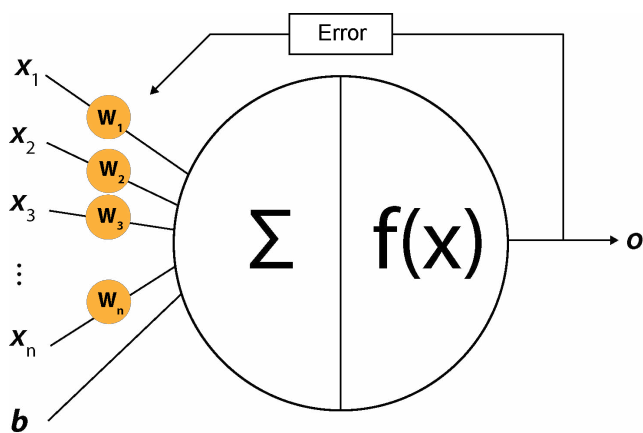


Figure A1. Schematic of an artificial neuron, where x_1-x_n are the input values, w_1-w_n are the input weights, b is the bias value, and o is the neuron output. The 'Error' loop demonstrates backpropagation for the adjustment of input weights during network training

Neural networks

A neural network is a type of machine-learning program that mimics the structure of neurons in the human brain, but processes data far more quickly. Neural networks are made up of three parts: (1) the input layer, (2) one or more hidden layers, and (3) the output layer (see Figures A6 and A7 below).

Perceptron

The perceptron is one of the earliest neural-network-based machine-learning models (Rosenblatt, 1958). It is a linear binary classifier that uses supervised learning to adjust the weight values of its inputs in order to more accurately reflect its training data. Since the output of a perceptron is a binary number (either (0, 1) or (-1, 1) depending on the activation function), the model can only classify linearly separable sets of input vectors. It cannot reliably classify non-linear data (Banoula, 2024).

The perceptron can stand as a simple proxy for all neural networks. While other types of network vary as to methods of activation, training, and network architecture, they all follow the same basic principles.

Step activation function

The following demonstrates the use of a step threshold activation function in a perceptron (Figure A2).

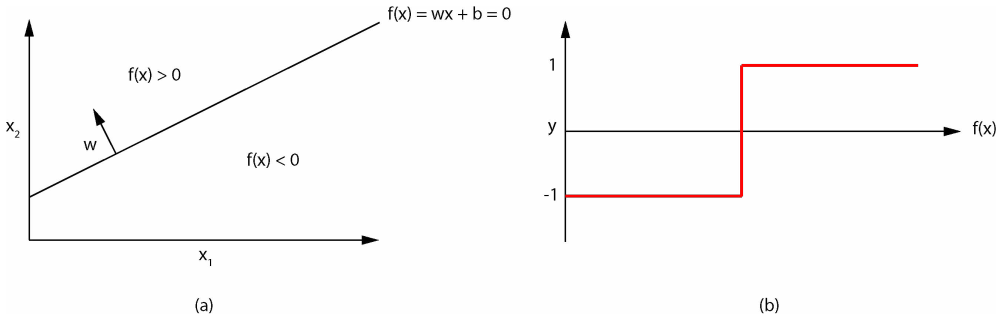


Figure A2. Activation of a perceptron using a step function: (a) linear classifier for $x = (x_1, x_2)$, (b) shows the all-or-nothing nature of the perceptron activation threshold

First, calculate the summation of the inputs by:

$$Y = f(\sum x_i \cdot w_i + b)$$

where:

w = input vector weight

x = input vector

b = bias.

Next, calculate whether the neuron activation threshold is met by:

$$f(x) = \begin{cases} 1, & w \cdot x + b > 0 \\ 0, & \text{otherwise} \end{cases}$$

where:

w = input vector weight

x = input vector

b = bias.

Activation is all or none.

Training using the perceptron learning rule (PLR)

Rosenblatt (1958) used what he termed the ‘perceptron learning rule’ to train the model. The error is the difference between the desired response and the actual response. The PLR is calculated by:

$$w_{new} = w_{old} + \eta(t - o)x_i$$

where:

t = target value

o = output of the perceptron

η = the learning rate.³⁵

Other training algorithms, such as stochastic gradient descent (GeeksForGeeks, 2024g; Van Otten, 2024c), may also be used.

Non-linear activation functions

Non-linearity can be introduced into a neural network by changing the activation function used in the hidden layer(s). These non-linear functions allow the model to capture much more intricate patterns in the input data. Common non-linear activation functions are sigmoid, hyperbolic tangent (tanh), rectified linear unit (ReLU)/Leaky ReLU, and softmax. Both the sigmoid and tanh functions suffer from the vanishing gradient problem,³⁶ which has led to the increased popularity of such functions as ReLU.

Non-linear activation functions solve the following problems (Baheti, 2021):

- **Backpropagation:** backpropagation can be accomplished because (1) the derivation function is related to the input, and (2) it is possible to understand which weights in the input neurons most optimally effect the prediction.
- **Layer stacking:** multiple neuron layers can be stacked as the output is the non-linear combination of inputs passed through multiple layers. Output is a functional computation of the model.

Sigmoid function

This function takes an input value and outputs a value in the range (0, 1). The larger the input value, the closer to 1 the output is, and vice versa. The output of the sigmoid function asymptotically approaches its minimum and maximum values but does not reach them (Figure A3).

The sigmoid value is calculated as:

$$f(x) = \frac{1}{1 + e^{-x}}$$

where:

x = the weighted sum of the inputs.

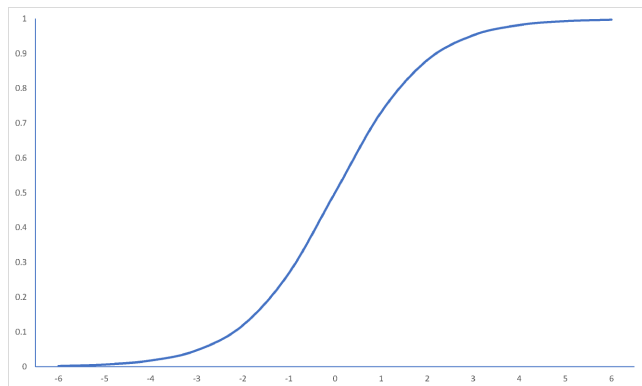


Figure A3. Output of a sigmoid function

The sigmoid function is a common choice when the model predicts the probability of an output. The function is differentiable³⁷ and provides a smooth gradient, which helps during gradient-based optimization while training.³⁸ Note that the sigmoid function has the potential to suffer from vanishing gradients, and is susceptible to output saturation.³⁹

Tanh function

The tanh function is similar to the sigmoid function, but the output value is in the range $(-1, 1)$. The larger the input value, the closer to 1 the output value is, and the lower the value the closer to -1 it is. The output of the tanh function asymptotically approaches its minimum and maximum values but does not reach them (Figure A4).

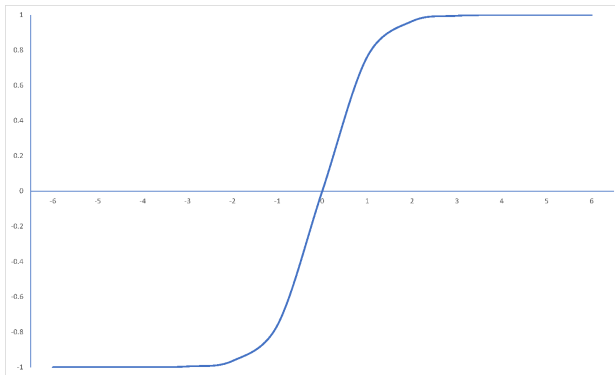


Figure A4. Output of a tanh function

Tanh is calculated by:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

where:

x = the weighted sum of the inputs.

The tanh function is used in a variety of networks, particularly recursive neural networks and long short-term memory networks (see below). The function is zero-centred, making it easy to map output as strongly negative, neutral, or strongly positive, and its steeper gradient makes it more sensitive to input changes. However, like the sigmoid function, it is susceptible to the vanishing gradient problem.

ReLU function

ReLU is widely used as an activation function, particularly in deep learning models. It is simple to implement and overcomes the vanishing gradient problem present in sigmoid and tanh functions (Van Otten, 2023a). In the ReLU function, negative values are mapped to zero and positive values are passed unchanged (Figure A5).

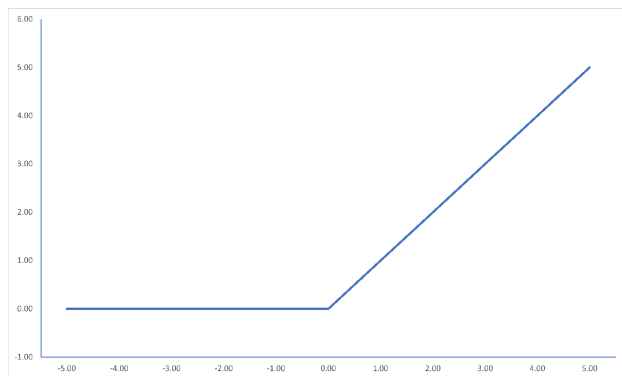


Figure A5. Output of the ReLU function

It is calculated by:

$$f(x) = \max(0, x) = \begin{cases} 0, & x < 0 \\ x, & \text{otherwise} \end{cases}$$

where:

x = the weighted sum of the inputs.

ReLU encourages sparse networks. Because some portion of the network may be inactive at any time, it avoids the vanishing gradient problem, it is computationally efficient, and it accelerates convergence in the loss function during training.⁴⁰ On the negative side, there is the potential to develop ‘dead’ neurons (neuron output is 0 regardless of the input value), and it saturates at the upper boundary, causing the neuron to lose sensitivity to large values.

An improved version, named Leaky ReLU, has been developed to address the dead-neuron problem. It is calculated by:

$$f(x) = \max(0.1 \cdot x, x)$$

where:

x = the weighted sum of the inputs.

This provides a small nonzero gradient for negative values and eliminates dead neurons.⁴¹

Softmax function

The Softmax function is an activation function commonly used in the output layer of a neural network for multi-class classification problems. It takes a vector of real numbers as input and normalizes it into a probability distribution, where the sum of the probabilities equals 1. (Van Otten, 2023a)

This function is a fundamental tool in multi-class classification, allowing the model to provide a probabilistic interpretation.

It is calculated by:

$$f(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

where:

z_i = the input value for class i

z_j = the input values for all classes.

The softmax function is differentiable and sensitive to differences between input values, and provides a probability distribution where all output values are positive and add up to 1. It is sensitive to outliers and large input values, which can cause numerical instability.

Shallow learning networks

A shallow learning network (Figure A6) typically has limited depth, with usually one hidden layer.⁴² While simple in design, it is a powerful tool that can approximate any function, provided there are a sufficient number of neurons in its hidden layer.⁴³ Shallow learning networks are the traditional ML model, and are used in many AI domains.

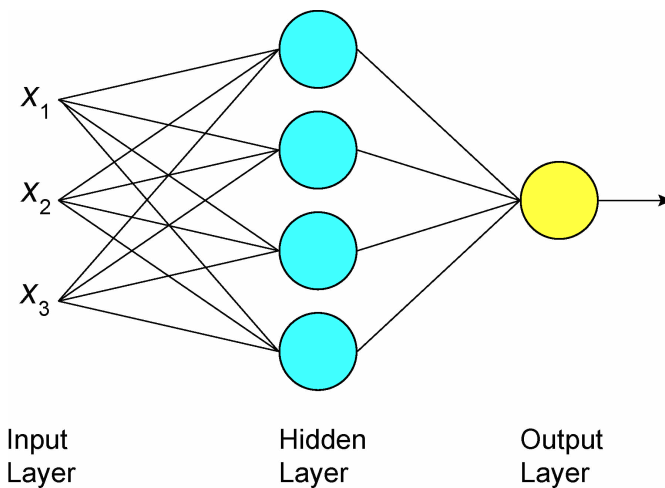


Figure A6. Schematic of a shallow, fully connected, learning network

Characteristics

- Limited depth: shallow models generally have only a single hidden layer.
- Fewer parameters: shallow models have fewer parameters, making them computationally inexpensive and easier to train.
- Feature engineering: shallow models need manual feature extraction for training.

- Efficiency: shallow models work well with smaller data sets, avoiding overfitting.⁴⁴

Advantages

- Interpretability: the simplicity of shallow learning models makes them more transparent and easier to interpret.
- Computational cost: shallow learning requires fewer resources, which makes it faster and more cost-effective to train.
- Low risk of overfitting: with fewer parameters to train, shallow models are less likely to overfit during training on small data sets.

Limitations

- Limited complexity: shallow models have difficulty learning complex patterns.
- Feature engineering: the necessity for manual feature extraction in the training data set requires domain expertise and time.
- Scalability: shallow learning does well on small-scale problems but has difficulty generalizing to larger data sets and higher-dimensional data.

Shallow learning algorithms

Common shallow learning algorithms are (Gino, 2024):

- Linear regression: a statistical model of the relationship between a dependent variable and one or more independent variables. The relationship between the input and target variable is assumed to be linear.
- Logistic regression: a linear model, used in binary or multi-class classification, that models the relationship between input features and the predicted probabilities of different classes (Spiceworks, 2022).
- Decision trees: a binary classification technique that partitions the feature space into regions and makes predictions based on the majority class or average value of the instances in each region (Spano, 2024).
- Random forests: an ensemble learning technique that aggregates multiple decision trees to improve robustness and precision (GeeksForGeeks, 2025c).
- Support vector machines (SVMs): a supervised, binary classification algorithm that performs a linear classification to find the optimal hyperplane to divide the data into different classes. Non-linear classifications can be performed by the use of kernel functions.⁴⁵

- *K*-nearest neighbors (*k*-NNs): a non-parametric, supervised learning technique used for classification and regression. Its predictions are based on the majority class or average value of the *k*-nearest neighbors in the feature space (anonymous, n.d.).
- Naive Bayes: a probabilistic classification technique, based on Bayes's Theorem, that models the distribution of inputs within a specific class or category. Despite their simplicity, they function well in such areas as text classification. See GeeksForGeeks (2025b) for an overview.

Deep learning networks

The main difference between shallow and deep models is the complexity of the network architecture (Holdsworth, 2024). A typical deep network contains dozens or hundreds of hidden layers, with many thousands of individual neurons (see Figure A7). The training of a deep network proceeds in an identical fashion to that of a shallow network, but requires exponentially more computing power and time to accomplish.

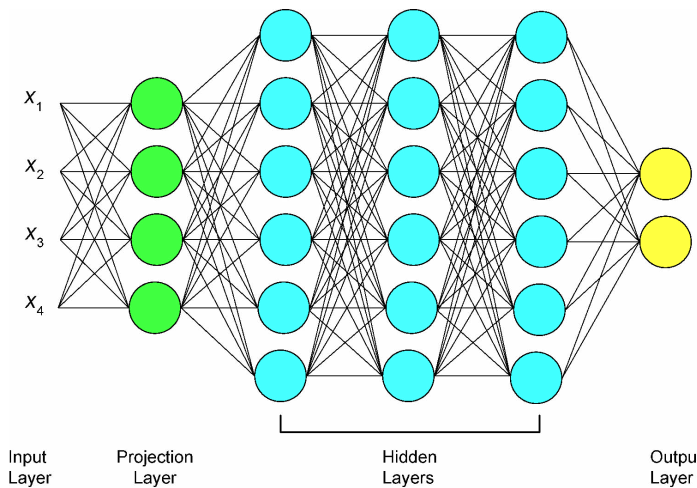


Figure A7. Schematic of a fully connected deep learning network

Characteristics

- Multiple hidden layers: deep models contain multiple hidden layers, which allows them to extract complex patterns from the data set.⁴⁶
- High model complexity: the depth of the model allows it to approximate complex, non-linear data relationships.
- End-to-end learning: the need for manual feature engineering is reduced, as the model can learn features automatically during training.
- Scalability: deep models scale well with an increase in the size of the data set, where the increase in the amount of data allows for effective model generalization.

Advantages

- Modeling of complex patterns: deep learning models can learn non-linear patterns in the data that shallow models cannot.
- Automated feature extraction: with multiple hidden layers, deep learning models are capable of automatically extracting features from the data set.
- Scalability: the output of deep learning models improves with more data.

Limitations

- Computational cost: training deep learning models is resource-intensive, generally requiring multiple graphics processing units (GPUs) in order to achieve a reasonable training time.
- Large training data set: deep learning models require large amounts of labeled data in order to avoid overfitting. The creation of this labeled training data set can take a significant amount of time and usually requires domain expertise in order to correctly label the data.
- Lack of interpretability: due to their multiple hidden layers, deep learning models are often considered to be ‘black boxes.’ This makes it difficult to interpret the model’s decision-making process.

Deep learning models

There are a number of deep learning models that have application to NLP.⁴⁷ The following provides a short overview of the most popular ones, enabling the reader to gain a basic familiarity with them. This is an area of active and intense research, with new models (and acronyms) being developed on what seems like a daily basis.⁴⁸

Recursive neural networks (RNNs)

RNNs (Poudel, 2023) recognize patterns in data sequences by using feedback loops to simultaneously process an input sequence while maintaining internal state storage of previous parts of the sequence (Figure A8). This allows them to do such things as determine the meaning of a word in a sentence by looking at the words that came before it.

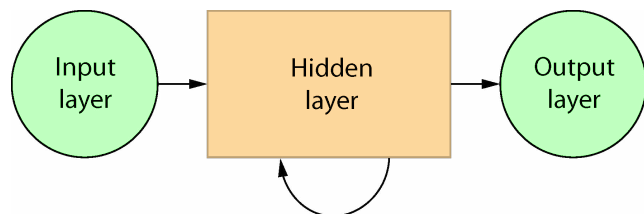


Figure A8. RNN architecture

RNNs are good at a number of different NLP tasks, such as language translation, text classification, language creation, and speech recognition. Both Siri® and Google Translate™ incorporate the use of RNNs.⁴⁹

Long short-term memory networks (LSTMs)

LSTMs (Figure A9) are a type of RNN that can learn long-term dependencies (Olah, 2015). LSTMs ‘use “memory cells” to store information and “gates” to regulate the flow of data into and out of the cells, which allows them to remember information for more extended periods than other types of RNNs’ (Van Otten, 2023c).

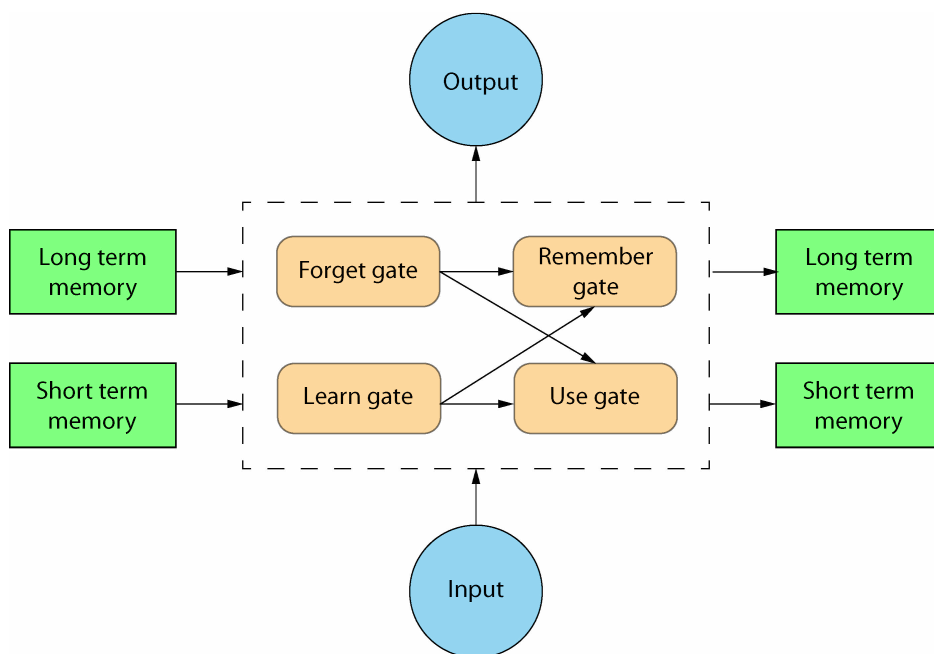


Figure A9. LSTM architecture

LSTMs are suitable for NLP tasks such as language generation, language modeling, and translation.

Transformer networks

A transformer network⁵⁰ combines an encoder–decoder architecture and a text-processing mechanism (Figure A10) in order to solve sequence-to-sequence tasks while also handling long-term dependencies (Kulshrestha, 2020a; Van Otten, 2023b). It does this by using self-attention (Kulshrestha, 2020b; Van Otten, 2023h) to process its input data.⁵¹ Because transformer networks process their data in parallel rather than sequentially, they are much faster than RNNs.

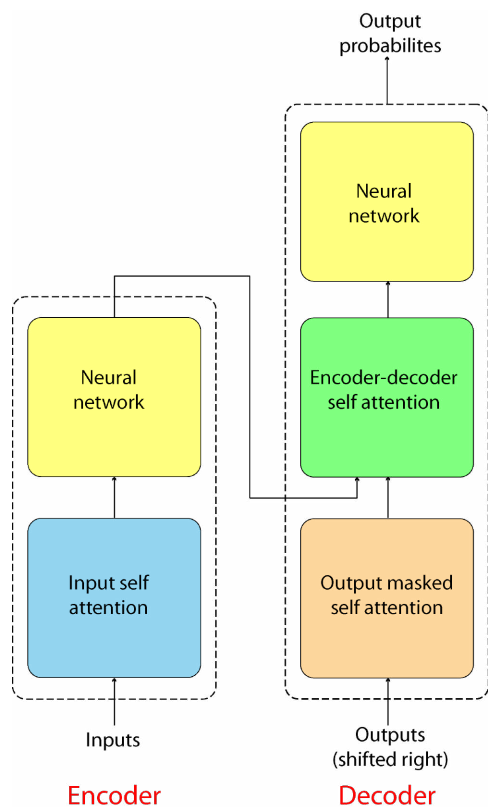


Figure A10. Transformer network architecture

Transformers are particularly well suited to NLP tasks, as they can process an entire sequence at once, rather than a single word at a time. Common NLP uses include language translation, text summarization, question answering, text generation, sentiment analysis, named-entity recognition, and text classification.

Bidirectional encoder representations from transformers (BERT)

BERT (Muller, 2022) is a type of transformer network developed in 2018 by researchers at Google™. As the authors state,

BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks. (Devlin et al., 2018)

This flexibility allows BERT models (there are a number of variants) to excel at a number of NLP tasks, such as sentiment analysis, question answering, text prediction, text generation, text summarization, and polysemy resolution.⁵²

Generative pre-training transformer (GPT)

The GPT architecture is a transformer model developed by researchers at OpenAI™ in 2018 (Radford et al., 2018).⁵³ GPT is the framework used by generative AI applications such as ChatGPT.⁵⁴ GPT models are trained to predict the next word in a sentence, based on the existing content of the sentence. This allows the model to generate coherent and grammatically accurate sentences. GPT models excel at such NLP tasks as language translation, summarization, and dialogue generation.

Notes

- 1 As an example of the use of NLP, the text of the abstract for this article was generated by the text summarization tool available from aspose.ai (<https://products.aspose.ai/words/summarize>).
- 2 Structured data is highly organized (usually quantitative) data that is easily decipherable by machine-learning algorithms; unstructured data is unorganized (often textual) data that cannot be used directly by a program (IBM Cloud Education Team, 2021; Jangid, 2023). The first task of an NLP application is to take this unstructured input and transform it into a state (usually a series of vectors) that can be processed by the application.
- 3 Topic modeling is an unsupervised learning technique that allows for the discovery of underlying topics in a collection of documents. See Murel and Kavlakoglu (2024) for an overview.
- 4 Semantic search is a data analysis method that uses NLP techniques to understand the contextual meaning and intent of a user's query in order to find matches in documents, rather than simply matching keywords from the query. See Gillis (2025) for an overview.
- 5 The earliest description of machine learning comes from a 1959 paper by Arthur Samuel, where he states, 'programming computers to learn from experience should eventually eliminate the need for much of this detailed programming effort' (Samuel, 1959: 211). This is often paraphrased as 'the field of study that gives computers the ability to learn without being explicitly programmed.'
- 6 A vector is a numerical representation of a word, sentence, or document. It is a mathematical point in an n -dimensional space, where each dimension represents a feature or characteristic of the text. Vectors allow NLP models to compare and understand the relationships between different pieces of text.
- 7 Neural networks were first proposed by Warren McCulloch and Walter Pitts in 1943 (McCulloch and Pitts, 1943). They were a major area of research until the late 1960s, when they fell out of fashion. There was a brief resurgence of interest in the early 1980s, but they did not become the dominant method of conducting many aspects of machine learning (including NLP) until the second decade of this century. This rise to dominance was fueled largely by the increased computing power of modern graphics processing chips (Hardesty, 2017). For a discussion of the different types of neural network see GeeksForGeeks (2024h).
- 8 See Van Otten (2023i) for an overview of popular text-cleaning techniques. Not all techniques are needed for every input text.
- 9 A token is usually a word, sub-word, or item of punctuation (GeeksForGeeks, 2024b). For example, in the sentence 'I can't see you.', the tokens are [I, can, 't, see, you, .].
- 10 Lemmatization is the process of grouping inflectional forms of a word into a root word (the lemma). For example, the words 'like,' 'likes,' and 'liked' all share the same lemma

(‘like’). However, the lemma of a word may be different than the word itself. For example, the lemma of ‘better’ is ‘good.’ See Eda (2023) and GeeksForGeeks (2024c) for discussions of lemmatization.

- 11 Stop words are commonly occurring words that add little value to the analysis of the text. In English, common stop words are ‘the’, ‘is’, ‘where’, ‘at’, ‘in’, and so on. Depending on the domain of the text being analysed, it may be necessary to include domain-specific stop words in preprocessing. Both SpaCy and NLTK contain extensive lists of stop words, and removal is easily accomplished. When an LLM such as BERT is used for the analysis of a document, stop word removal is optional. Stop words provide valuable information about document structure to the LLM.
- 12 The information presented in this section is derived from Dey (2023), Durna (2024), GeeksForGeeks (2025d), Van Otten (2022a, 2022h, 2024b).
- 13 In NLP, the meaning of the word ‘document’ differs from its common usage. For NLP purposes, a document is a discrete segment of text that represents a single input to the model. A document can vary in size from a single sentence to a paragraph, chapter, or the complete text of a book, depending on the analytical task being undertaken.
- 14 Comparison between documents is accomplished by determining the distance between those documents. Each document is represented by a vector, which maps to a mathematical point in n -dimensional space. The distance between documents is a measure of how far apart those points are in that n -dimensional space. The most straightforward calculation of distance uses Euclidean distance, which represents the shortest inter-document path (Henner, 2023). This is found by $d(p, q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}$. Since Euclidean distance is sensitive to magnitude, a common alternative is cosine distance, which normalizes the vectors and emphasizes relative weight, $\text{cosine distance} = 1 - (A \cdot B / \|A\| \|B\|)$. See Cardenas (2023) for a detailed example of the calculation of cosine distance. There are a number of other algorithms for calculating distance.
- 15 Since these three types of embedding each have multiple implementation models, the types are generally referred to as Word2Vec, Sent2Vec, and Doc2Vec respectively.
- 16 SpaCy is the de facto standard API used for the development of NLP applications in Python (<https://spacy.io/>).
- 17 For an overview of dependency parsing and dependency graphs, see Van Otten (2023k).
- 18 Language often contains ambiguous words (words whose meaning changes on use context). Syntactic parsing provides the framework for resolving those ambiguities.
- 19 For a detailed discussion of how to implement a relation extraction component for biomedical data, see Van Landeghem and Slocum (2023).
- 20 Rules-based paraphrasing was done using WordNet (Verma, 2021; Pramod, 2023) or via thesaurus replacement. Common machine-learning models were generative adversarial networks (de Rosa and Papa, 2022; GeeksForGeeks, 2024a) or reinforcement learning (Wang et al., 2018; Tech Robot, 2024) models.
- 21 The term ‘opinion mining’ is often used interchangeably with ‘sentiment analysis’, but Van Otten (2023d) sees it as a broader set of techniques that include sentiment analysis, but also deal with the extraction of specific emotions or opinions from the text.
- 22 Supervised learning is a machine-learning technique where a training data set is created by a person who labels words or phrases with the appropriate category (Van Otten, 2022f). In the case of sentiment analysis, the text is given a single label conveying its sentiment (positive, negative, neutral).
- 23 A declarative chatbot has scripted responses it applies to user queries. A predictive chatbot (also known as a conversational chatbot) use NLP techniques, built on transformer models,

- to learn user preferences and offer personalized responses to user queries. Siri® (Apple) and Alexa® (Amazon) are two examples of predictive chatbots.
- 24 In late January 2025, Deepseek introduced their R1 model, which has thrown the US AI industry into turmoil. Not only was training this model vastly cheaper (a reported US\$6 million instead of the hundreds of millions for the latest version of ChatGPT), but early tests also have shown that it performs on a par with the best existing chatbots. Additionally, the model can be run on a local machine (something impossible with ChatGPT) and it is free, as Deepseek has released the code as open-source (<https://github.com/deepseek-ai/DeepSeek-R1>). This development calls into question the scale-based approach (bigger data centres, larger data sets, more resource requirements) used by US chatbot developers to improve their models (Deepseek AI, 2025; Orland, 2025). Commentators are calling this a ‘Sputnik moment’ for the US AI industry.
 - 25 The Indexers Discussion Group at <https://idg.groups.io/g/main> is ‘a group discussion list for professional indexers and anyone interested in indexing books, journals, and other print and electronic media, the freelance life, and the business of indexing.’ Messages posted to the list are only visible to list members.
 - 26 See <https://www.asindexing.org/newsletter/see-also-january-2025/#all>.
 - 27 The keynote speaker, Thad McIlroy, has published a book on the use of AI in the publishing industry (McIlroy, 2025), which can be read online.
 - 28 See <https://www.txyz.com/texttract/index.htm>.
 - 29 The term ‘node’ is used interchangeably with ‘neuron’ in the literature.
 - 30 A neuron input is the numerical representation of data, often words or parts of words, that is fed into a neuron for processing. This input is typically a vector that represent the input’s features or characteristics.
 - 31 Input weights are the numerical values that determine the strength of the connections between the input layer and the first hidden layer. These weights are crucial for the network’s learning process and how it interprets the input data. When the model is initialized, each individual weight is assigned a random value. This allows each neuron in a layer to be a unique function. The weight values are adjusted during training. The best initialization method is Xavier’s Initialization (Idrees, 2024).
 - 32 A bias is a constant associated with each neuron.

Biases serve as a form of offset or threshold, allowing neurons to activate even when the weighted sum of their inputs is not sufficient on its own. They introduce a level of adaptability that ensures the network can learn and make predictions effectively ... This flexibility is crucial because real-world data is rarely perfectly aligned with specific thresholds. Biases enable neurons to activate in response to various input conditions, making neural networks more robust and capable of handling complex patterns. (GeeksForGeeks, 2024j)

Like weights, bias is adjusted during network training.

- 33 An activation function is a mathematical function that determines the output of a neuron in the network. All the neurons in a layer share the same activation function. In deep learning models, the activation function can vary for each hidden layer.
- 34 Neural networks are trained using a combination of forward propagation and backpropagation. In forward propagation each node calculates its output using its input, algorithmic weights and bias, and activation function. After forward propagation is completed, an error value is calculated by analyzing the network output compared to the actual target value. Backpropagation transmits the error value back through the network, with each node updating its weights and bias in order to minimize the loss function through the use of

- one of a number of different optimization functions (GeeksForGeeks, 2024e). See Chauhan (2020) for a discussion of common optimization functions used in neural networks.
- 35 The learning rate is the amount of change to a model's parameters made during each iteration of the training process (Van Otten, 2024a).
 - 36 The vanishing gradient problem occurs when the rate of change of the loss function becomes too small during backpropagation. The optimizer then has problems updating parameters and improving the model (Van Otten, 2023g).
 - 37 For an explanation and overview of differentiable functions, see Wikipedia (n.d.).
 - 38 Gradient-based optimization is a fundamental technique in NLP used to train neural-network models. It involves iteratively adjusting model parameters to minimize a loss function, which measures the difference between predicted and actual outputs.
 - 39 Output saturation occurs when the input is very positive or negative, which causes an output very close to the maximum or minimum value. This can slow learning, as the network is less sensitive to changes in input near its extremes.
 - 40 A loss function is a way to measure how well a model's predictions align with the actual data. The lower the loss value, the better your model is doing.
 - 41 There are a number of other ReLU functions, see Baheti (2021) and Van Otten (2023a).
 - 42 Hidden layers are the layers of neurons that transform data from the input layer into information that can be used by the output layer. They are termed hidden since they are not directly accessible to the user.
 - 43 This property is called the universal approximation theorem (GeeksForGeeks, 2024i).
 - 44 Overfitting occurs when the model can reliably classify data it was trained on, but cannot accurately classify new data.
 - 45 A kernel function maps data from its input space to a new space where SVM can perform a non-linear classification of the data (Pasupa and Sunhem, 2016: 2).
 - 46 See Yadav (2024a) for a discussion of how to select the number of neurons in a hidden layer, and the number of hidden layers required for the model.
 - 47 For readers interested in exploring publicly available machine-learning models, data sets, and applications, Hugging Face (<https://huggingface.co/>) is the place to start.
 - 48 Information for this section is from Biswal (2025), Holdsworth (2024), and Van Otten (2023c).
 - 49 Siri is a registered trademark of Apple, Inc., Google and Google Translate are trademarks of Google, LLC.
 - 50 The transformer architecture was first proposed by researchers at Google in 2017 (Vaswani et al., 2017). All transformer-based models fall in the general category of large language models (LLMs). See Vaniukov (2024) for a discussion of how LLMs are distinct from traditional NLP techniques.
 - 51 Self-attention 'is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence' (Kulshrestha, 2020a). Consider the following sentences, 'I poured water from the **bottle** into the **cup** until **it** was full' and 'I poured water from the **bottle** into the **cup** until **it** was empty.' Self-attention allows a transformer-based model to correctly associate the pronoun 'it' with 'cup' in the first sentence and 'bottle' in the second sentence.
 - 52 Polysemy resolution is the ability of the model to differentiate words that have multiple meanings. In the following sentences, 'I deposited some money in the **bank**' and 'I went to the **bank** of the river', BERT models can distinguish the two different meanings of the word 'bank.'
 - 53 OpenAI is a trademark of OpenAI in the United States and other countries.
 - 54 For an accessible overview of the ChatGPT architecture with a minimum of mathematics, see Dugas (n.d.).

References

- Agrawal, R. (2019) *Shallow neural networks: toward data science*. Available at <https://towardsdatascience.com/shallow-neural-networks-23594aa97a5>.
- Ali, A. (2023) 'Understanding the NLP pipeline: a comprehensive guide,' *Medium*. Available at https://medium.com/@asjad_ali/understanding-the-nlp-pipeline-a-comprehensive-guide-828b2b3cd4e2.
- Al-Ma'arif, Q. (2024a) 'The essential role of syntactic and semantic parsing in NLP', *Medium*. Available at <https://medium.com/@datailm/the-essential-role-of-syntactic-and-semantic-parsing-in-nlp-47b92118d9de>.
- Al-Ma'arif, Q. (2024b) 'The key to unlocking true language understanding: coreference resolution', *Medium*. Available at <https://medium.com/@datailm/the-key-to-unlocking-true-language-understanding-coreference-resolution-c01d569e2e87>.
- Anonymous (n.d.) 'What is the k-nearest neighbors (KNN) algorithm?', *Think*. Available at <https://www.ibm.com/think/topics/knn>.
- Baheti, P. (2021) 'Activation functions in neural networks [12 types & use cases]', *V7 Blog*. Available at <https://www.v7labs.com/blog/neural-networks-activation-functions>.
- Banoula, M. (2024) 'What is perceptron?', *Simplilearn*. Available at <https://www.simplilearn.com/tutorials/deep-learning-tutorial/perceptron>.
- Biswal, A. (2025) 'Top 10 deep learning algorithms you should know in 2025', *Simplilearn*. Available at <https://www.simplilearn.com/tutorials/deep-learning-tutorial/deep-learning-algorithm>.
- Box, G. (1976) 'Science and statistics', *Journal of the American Statistical Association* **71**(356), 791–9. Available at <https://doi.org/10.1080/01621459.1976.10480949>.
- Browne, G. (2024) 'Looking for examples on the use of controlled vocabulary tools, and any potential role in AI', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/106750928>.
- Byers, M. and Boley, A. (2023) *Who owns outer space? International law, astrophysics, and the sustainable development of space*, 1st edn. Cambridge: Cambridge University Press. Available at <https://doi.org/10.1017/9781108597135>.
- Cardenas, E. (2023) 'Distance metrics in vector search', *Weaviate*. Available at <https://weaviate.io/blog/distance-metrics-in-vector-search>.
- Chauhan, N. S. (2020) 'Optimization algorithms in neural networks', *KDnuggets*. Available at <https://www.kdnuggets.com/2020/12/optimization-algorithms-neural-networks.html>.
- Ching, N. (2023) 'ChatGPT, or: everything old is new again . . .?', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/99315346>.
- Connolly, D. (2024) 'AI indexing and publisher policies', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/104332943>.
- Dahl, M., Magesh, V., Suzgun, M. and Ho, D. E. (2024) 'Large legal fictions: profiling legal hallucinations in large language models', *Journal of Legal Analysis*, **16**(1), 64–93. Available at <https://doi.org/10.1093/jla/laae003>.
- de Rosa, G. H. and Papa, J. P. (2022) 'A survey on text generation using generative adversarial networks'. *arXiv*. Available at <https://arxiv.org/pdf/2212.11119>.
- Deepseek AI (2025) 'DeepSeek-R1: incentivizing reasoning capability in LLMs via reinforcement learning', *arXiv*. Available at <https://arxiv.org/abs/2501.12948>.
- Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. (2018) 'BERT: pre-training of deep bidirectional transformers for language understanding'. *arXiv*. Available at <https://doi.org/10.48550/ARXIV.1810.04805>.

- Dey, S. (2023) 'From traditional to modern: a comprehensive guide to text representation techniques in NLP', *Medium*. Available at <https://deysusovan93.medium.com/from-traditional-to-modern-a-comprehensive-guide-to-text-representation-techniques-in-nlp-369946f67497>.
- Dugas, D. (n.d.) 'The GPT-3 architecture, on a napkin'. Available at https://dugas.ch/artificial_curiosity/GPT_architecture.html.
- Durna, M. B. (2024) 'Text representation techniques', *Medium*. Available at <https://medium.com/@mervebdurna/text-representation-techniques-d40741eb0916>.
- Eda, A. (2023) 'A gentle introduction to lemmatization', *Medium*. Available at <https://medium.com/@amos.eda/nlp-basics-part-1-a-gentle-introduction-to-lemmatization-f739b7d2b0c>.
- GeeksForGeeks (2024a) 'Generative adversarial network (GAN)', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/generative-adversarial-network-gan/>.
- GeeksForGeeks (2024b) 'How tokenizing text, sentence, words works', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/nlp-how-tokenizing-text-sentence-words-works/>.
- GeeksForGeeks (2024c) 'Lemmatization with NLTK', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/python-lemmatization-with-nltk/>.
- GeeksForGeeks (2024d) 'Machine translation in AI', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/machine-translation-of-languages-in-artificial-intelligence/>.
- GeeksForGeeks (2024e) 'Neural network node', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/neural-network-node/>.
- GeeksForGeeks (2024f) 'Relationship extraction in NLP', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/relationship-extraction-in-nlp/>.
- GeeksForGeeks (2024g) 'Stochastic gradient descent (SGD)', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd/>.
- GeeksForGeeks (2024h) 'Types of neural networks', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/types-of-neural-networks/>.
- GeeksForGeeks (2024i) 'Universal approximation theorem for neural networks', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/universal-approximation-theorem-for-neural-networks/>.
- GeeksForGeeks (2024j) 'Weights and bias in neural networks', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/the-role-of-weights-and-bias-in-neural-networks>.
- GeeksForGeeks (2025a) 'Doc2Vec in NLP', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/doc2vec-in-nlp/>.
- GeeksForGeeks (2025b) 'Naive Bayes classifiers', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/naive-bayes-classifiers/>.
- GeeksForGeeks (2025c) 'Random forest algorithm in machine learning', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/>.
- GeeksForGeeks (2025d) 'Understanding TF-IDF (term frequency-inverse document frequency)', *Geeks for Geeks*. Available at <https://www.geeksforgeeks.org/understanding-tf-idf-term-frequency-inverse-document-frequency/>.
- Gillis, A. S. (2025) 'What is semantic search?', *TechTarget*. Available at <https://www.techtarget.com/searchenterpriseai/definition/semantic-search>.
- Gino, S. (2024) 'The important role of shallow-learning and statistics in modern data-science era', *Medium*. Available at <https://sgino209.medium.com/the-important-role-of-shallow-learning-and-statistics-in-modern-data-science-era-01b52b0f8e60>.
- Hardesty, L. (2017) 'Explained: neural networks', *MIT News*. Available at <https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414>.

- Hashemi-Pour, C. and Barney, N. (2024) 'What is named entity recognition (NER)?', *TechTarget*. Available at <https://www.techtarget.com/whatis/definition/named-entity-recognition-NER>.
- Henner, K. (2023) 'An intuitive introduction to text embeddings', *Stack Overflow*. Available at <https://stackoverflow.blog/2023/11/09/an-intuitive-introduction-to-text-embeddings/>.
- Holdsworth, J. (2024) 'What is deep learning?', *Think*. Available at <https://www.ibm.com/think/topics/deep-learning>.
- Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B. and Liu, T. (2024) 'A survey on hallucination in large language models: principles, taxonomy, challenges, and open questions', *ACM Transactions on Information Systems*, **1**(1). Available at <https://doi.org/10.1145/3703155>.
- Hyperskill (n.d.) 'Introduction to dialogue systems', *Hyperskill*. Available at <https://hyperskill.org/learn/step/30048>.
- IBM Cloud Education Team (2021) 'Structured vs unstructured data', *Think*. Available at <https://www.ibm.com/think/topics/structured-vs-unstructured-data>.
- Idrees, H. (2024) 'Shallow learning vs. deep learning: is bigger always better?', *Medium*. Available at <https://medium.com/@hassaanidrees7/shallow-learning-vs-deep-learning-is-bigger-always-better-51c0bd21f059>.
- Izzard, T. (2024) 'Generative artificial intelligence (AI) and its performance at indexing tasks', *The Indexer* **42**(4), 383–400. Available at <https://doi.org/10.3828/index.2024.24>.
- Jangid, A. (2023) 'Transforming unstructured text into structured insights with NLP techniques', *Medium*. Available at <https://medium.com/@jangidajay271/transforming-unstructured-text-into-structured-insights-with-nlp-techniques-f0537583f563>.
- Jha, A. (2023) 'Vectorization techniques in NLP', *Neptune Blog*. Available at <https://neptune.ai/blog/vectorization-techniques-in-nlp-guide>.
- Ji, Z., Yu, T., Xu, Y., Lee, N., Ishii, E. and Fung, P. (2023) 'Towards mitigating hallucination in large language models via self-reflection', *openreview.net*. Available at <https://openreview.net/pdf?id=up8EYzrKV>.
- Jones, K. P. (1990) 'Natural-language processing and automatic indexing: a reply', *The Indexer*, **17**(2), 114–15. Available at <https://doi.org/10.3828/indexer.1990.17.2.8>.
- Kang, H. and Liu, X.-Y. (2023) 'Deficiency of large language models in finance: an empirical examination of hallucination', *arXiv*. Available at <https://doi.org/10.48550/ARXIV.2311.15548>.
- Korycinski, C. and Newell, A. F. (1990) 'Natural-language processing and automatic indexing', *The Indexer* **17**(1), 21–9. Available at <https://doi.org/10.3828/indexer.1990.17.1.8>.
- Kulshrestha, K. (2020a) *Transformers: toward data science*. Available at <https://towardsdatascience.com/transformers-89034557de14>.
- Kulshrestha, K. (2020b) *Understanding attention in deep learning: toward data science*. Available at <https://towardsdatascience.com/attaining-attention-in-deep-learning-a712f93bdbl>.
- Lazaruk, S. (2024) 'B.C. Law Society investigates lawyer who used AI to make fake case law', *Vancouver Sun*, 27 February. Available at <https://vancouversun.com/news/local-news/fake-case-law-in-b-c-divorce-court-points-up-pitfalls-with-ai-tools-for-lawyers>.
- Li, J., Chen, J., Ren, R., Cheng, X., Zhao, W. X., Nie, J. and Wen, J. (2024) 'The dawn after the dark: an empirical study on factuality hallucination in large language models', *arXiv*. Available at <https://doi.org/10.48550/ARXIV.2401.03205>.
- McCulloch, W. S. and Pitts, W. (1943) 'A logical calculus of the ideas immanent in nervous activity', *Bulletin of Mathematical Biophysics* **5**, 115–33.
- McIlroy, T. (2025) 'The AI revolution in book publishing: a concise guide to navigating artificial intelligence for writers and publishers', *LeanPub*. Available at <https://leanpub.com/ai-revolution-in-book-publishing/read>.

- Maślankowska, M. and Mielniczuk, P. (2022) 'Intro to coreference resolution in NLP', *neurosys*. Available at <https://neurosys.com/blog/intro-to-coreference-resolution-in-nlp>.
- Mathur, V. (2023) 'PEGASUS large language model', *Medium*. Available at <https://medium.com/@varun5/pegasus-large-language-model-8c2aece1e11>.
- Modi, A. (2024) 'Adobe transforms PDF experiences with general availability of Acrobat AI Assistant', *Adobe Blog*. Available at <https://blog.adobe.com/en/publish/2024/04/15/adobe-transforms-pdf-experiences-with-general-availability-acrobat-ai-assistant>.
- Moffat, S. (2024) 'AI, freelancing and publishing', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/106879607>.
- Mudadla, S. (2023) 'What is parts of speech (POS) tagging natural language processing?', *Medium*. Available at <https://medium.com/@sujathamudadla1213/what-is-parts-of-speech-pos-tagging-natural-language-processing-in-2b8f4b07b186>.
- Muller, B. (2022) 'BERT 101 state of the art NLP model explained', *Hugging Face*. Available at <https://huggingface.co/blog/bert-101>.
- Murel, J. and Kavlakoglu, E. (2024) 'What is topic modeling?', *Think*. Available at <https://www.ibm.com/think/topics/topic-modeling>.
- Olah, C. (2015) 'Understanding LSTM networks', *Colah's Blog*. Available at <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>.
- Orland, K. (2025) 'How does DeepSeek R1 really fare against OpenAI's best reasoning models?', *Ars Technica*. Available at <https://arstechnica.com/ai/2025/01/how-does-deepseek-r1-really-fare-against-openais-best-reasoning-models/>.
- Palachy, S. (2020) 'Document embedding techniques', *Topbots*. Available at <https://www.topbots.com/document-embedding-techniques/#unsupervised-document-embedding>.
- Parnaby, L. (2023) 'NYC lawyers who used ChatGPT for research in aviation injury claim are fined \$5,000 in legal first after judge found AI chatbot cited legal cases that were FAKE and contained "gibberish"', *Daily Mail*, 23 June. Available at <https://www.dailymail.co.uk/news/article-12226823/NYC-lawyers-used-ChatGPT-fined-5-000-fake-submissions-aviation-injury-case.html>.
- Pasupa, K. and Sunhem, W. (2016) 'A comparison between shallow and deep architecture classifiers on small dataset', in *8th International Conference on Information Technology and Electrical Engineering* Yogyakarta, Indonesia. Available at <https://doi.org/10.1109/ICITEED.2016.7863293>.
- Peace, A. (2023) 'Something AI can't do', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/100876393>.
- Poudel, S. (2023) 'Recurrent neural network (RNN) architecture explained', *Medium*. Available at <https://medium.com/@poudelsushmita878/recurrent-neural-network-rnn-architecture-explained-1d69560541ef>.
- Pramod, O. (2023) 'WordNet in action: enhancing NLP capabilities through NLTK', *Medium*. Available at <https://medium.com/@ompramod9921/wordnet-in-action-enhancing-nlp-capabilities-through-nltk-2aa922a57d43>.
- Premadasa, N. (2024) 'Understanding the essence of NLP pipelines', *Nerd for Tech*. Available at <https://medium.com/nerd-for-tech/understanding-the-essence-of-nlp-pipelines-fb829907598f>.
- Proctor, J. (2024) 'B.C. lawyer reprimanded for citing fake cases invented by ChatGPT', *CBC News*, 24 February. Available at <https://www.cbc.ca/news/canada/british-columbia/lawyer-chatgpt-fake-precedent-1.7126393>.
- Radford, A., Narasimhan, K., Salimans, T. and Sutskever, I. (2018) 'Improving language understanding by generative pre-training'. Available at https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

- Reed, C. (2023) 'Author's AI index inquiry', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/99340131>.
- Rising, J. (2023) 'AI applications to publishing', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/message/49892>.
- Rooney, P. (2015) 'Indexing an unstructured manuscript', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/772043>.
- Rosenblatt, F. (1958) 'The perceptron: a probabilistic model for information storage and organization in the brain', *Psychological Review* **65**(6), 386–408.
- Saigh, R. (2023) 'Artificial intelligence', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/103420210>.
- Samuel, A. L. (1959) 'Some studies in machine learning using the game of checkers', *IBM Journal of Research and Development* **3**(3), 210–29.
- Shah, D. (2023) 'The beginner's guide to hallucinations in large language models', *Lakera*. Available at <https://www.lakera.ai/blog/guide-to-hallucinations-in-large-language-models>.
- Shah, S. V. (2024) 'Accuracy, consistency, and hallucination of large language models when analyzing unstructured clinical notes in electronic medical records', *JAMA Network Open* **7**(8). Available at <https://doi.org/10.1001/jamanetworkopen.2024.25953>.
- Shariq, A. (2023) 'Understanding natural language inference: from models to containers', *Medium*. Available at <https://medium.com/red-buffer/title-understanding-natural-language-inference-from-models-to-containers-26c1e777d72f>.
- Snyder, B., Moisescu, M. and Zafar, M. B. (2024) 'On early detection of hallucinations in factual question answering', in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. KDD '24: The 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Barcelona: ACM, 2721–32. Available at <https://doi.org/10.1145/3637528.3671796>.
- Spano, M. (2024) 'Shallow learning roots: decision trees'. Available at https://www.mitchspano.com/blog/decision_trees.
- Spiceworks (2022) 'What is logistic regression? Equation, assumptions, types, and best practices', *spiceworks*. Available at <https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/>.
- Stanford Natural Language Processing Group (n.d.) *Machine translation*. Available at <https://nlp.stanford.edu/projects/mt.shtml>.
- Stryker, C. (2024) 'What is NLP?', *Think*. Available at <https://www.ibm.com/think/topics/natural-language-processing>.
- Tech Robot (2024) 'The role of reinforcement learning in NLP', *Medium*. Available at <https://medium.com/@techrobot45/the-role-of-reinforcement-learning-in-nlp-72d9d9cd2b40>.
- Ullstrom, S. (2023) 'Hello AI, goodbye indexers?', *Indexing Insights*. Available at <https://www.stephenullstrom.com/hello-ai-goodbye-indexers/>.
- Unger, V. (2024) 'The great AI indexing experiment', *Indexer's Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/104320017>.
- van Landeghem, S. and Slocum, V. (2023) 'Implementing a custom trainable component for relation extraction', *Explosion Blog*. Available at <https://explosion.ai/blog/relation-extraction>.
- van Otten, N. (2022a) 'How to implement bag-of-words in Python', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2022/12/20/bag-of-words-python/>.
- van Otten, N. (2022b) 'How to implement named entity recognition in Python with SpaCy, BERT, NLTK & Flair', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2022/12/06/named-entity-recognition-ner/>.
- van Otten, N. (2022c) 'How to implement sentiment analysis in Python', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2022/12/16/sentiment-analysis-tools-in-python/>.

- van Otten, N. (2022d) 'How to translate text in Python', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2022/12/19/translate-text-python/>.
- van Otten, N. (2022e) 'NLP text summarization: popular machine learning and deep learning algorithms', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2022/12/01/nlp-text-summarization/>.
- van Otten, N. (2022f) 'Supervised learning explained and how to choose the right model', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2022/10/20/supervised-learning/>.
- van Otten, N. (2022g) 'Text classification: how to in Python', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2022/12/20/text-classification-python/>.
- van Otten, N. (2022h) 'TF-IDF made simple & how to get started in Python tutorial', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2022/11/28/tf-idf/>.
- van Otten, N. (2023a) 'Activation function: top 9 most popular explained & when to use them', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/06/16/activation-function/>.
- van Otten, N. (2023b) 'How to implement transformers for natural language processing (NLP)', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/01/23/transformers-natural-language-processing/>.
- van Otten, N. (2023c) 'Most popular deep learning algorithms for natural language processing (NLP)', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/01/04/deep-learning-for-natural-language-processing/>.
- van Otten, N. (2023d) 'Opinion mining: top 8 useful tools more than sentiment analysis', *Natural Language Processing (NLP) Blog*. Available at https://spotintelligence.com/2023/01/17/opinion-mining-sentiment-analysis/#Opinion_mining_vs_sentiment_analysis.
- van Otten, N. (2023e) 'Practical guide to Doc2Vec & how to tutorial in Python', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/09/06/doc2vec/>.
- van Otten, N. (2023f) 'The basics of syntactic analysis', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/10/28/syntactic-analysis-nlp/>.
- van Otten, N. (2023g) 'The vanishing gradient problem, how to detect & overcome it', *The Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/02/06/vanishing-gradient-problem/>.
- van Otten, N. (2023h) 'Top 6 most useful attention mechanism in NLP explained and when to use them', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/01/12/attention-mechanism-in-nlp/>.
- van Otten, N. (2023i) 'Top 20 essential text cleaning techniques', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/09/18/top-20-essential-text-cleaning-techniques-practical-how-to-guide-in-python/>.
- van Otten, N. (2023j) 'What is a question-answering system?', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/01/20/question-answering-qa-system-nlp/>.
- van Otten, N. (2023k) 'What is dependency parsing in NLP?', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/10/22/dependency-parsing/>.
- van Otten, N. (2023l) 'What is information extraction?', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/10/10/information-extraction/>.
- van Otten, N. (2023m) 'What is neural machine translation?', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2023/01/04/neural-machine-translation/>.
- van Otten, N. (2024a) 'Learning rate in machine learning and deep learning made simple', *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2024/02/19/learning-rate-machine-learning/>.

- van Otten, N. (2024b) ‘Text representation: a simple explanation of complex techniques’, *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2024/10/01/text-representation-a-simple-explanation-of-complex-techniques/>.
- van Otten, N. (2024c) ‘Understanding stochastic gradient descent (SGD) in machine learning’, *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2024/03/05/stochastic-gradient-descent-sgd-machine-learning/>.
- van Otten, N. (2024d) ‘What is coreference resolution in NLP?’, *Natural Language Processing (NLP) Blog*. Available at <https://spotintelligence.com/2024/01/17/coreference-resolution-nlp/>.
- Vaniukov, S. (2024) ‘NLP vs LLM: a comprehensive guide to understanding key differences’, *Medium*. Available at <https://medium.com/@vaniukov.s/nlp-vs-llm-a-comprehensive-guide-to-understanding-key-differences-0358f6571910>.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. and Polosukhin, I. (2017) ‘Attention is all you need’, in *31st Conference on Neural Information Processing Systems*, Long Beach, CA. Available at <https://arxiv.org/pdf/1706.03762>.
- Verma, Y. (2021) ‘A complete guide to using WordNET in NLP applications’, *Analytics India Magazine*. Available at <https://analyticsindiamag.com/deep-tech/a-complete-guide-to-using-wordnet-in-nlp-applications/>.
- Wang, W. Y., Li, J. and He, X. (2018) ‘Deep reinforcement learning for NLP’, in Y. Artzi and J. Eisenstein (eds), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*. Melbourne: Association for Computational Linguistics, pp. 19–21. Available at <https://doi.org/10.18653/v1/P18-5007>.
- Wikipedia (n.d.) ‘Differentiable function’. Available at https://en.wikipedia.org/wiki/Differentiable_function.
- Winston, H. (2024) ‘Indexing Assistant (Adobe)’, *Indexer’s Discussion Group (IDG)*. Available at <https://idg.groups.io/g/main/topic/106175318>.
- Yadav, A. (2024a) ‘How to create hidden layers in neural networks’, *Medium*. Available at <https://medium.com/biased-algorithms/how-to-create-hidden-layers-in-neural-networks-31772414445f>.
- Yadav, A. (2024b) ‘Word embeddings vs. sentence embeddings’, *Medium*. Available at <https://medium.com/biased-algorithms/word-embeddings-vs-sentence-embeddings-a0295be74d4b>.
- Yse, D. L. (2022) ‘Paraphrasing in natural language processing (NLP)’, *Medium*. Available at <https://lopezysse.medium.com/paraphrasing-in-natural-language-processing-nlp-857c28e68488>.
- Yu, L., Cao, M., Cheung, J. C. K. and Dong, Y. (2024) ‘Mechanistic understanding and mitigation of language model non-factual hallucinations’, *arXiv*. Available at <https://doi.org/10.48550/ARXIV.2403.18167>.
- Yücel, B. (2023) ‘Generative vs. extractive language models’, *Haystack*. Available at <https://haystack.deepset.ai/blog/generative-vs-extractive-models>.
- Zhao, X., Deng, Y., Yang, M., Wang, L., Zhang, R., Cheng, H., Lam, W., Shen, Y. and Xu, R. (2023) ‘A comprehensive survey on relation extraction: recent advances and new frontiers’, *arXiv*. Available at <https://doi.org/10.48550/ARXIV.2306.02051>.