Review

An Overview of Recent Advances in Natural Language Processing for Information Systems

Douglas O'Shaughnessy *

Posted Date: 2 January 2026

doi: 10.20944/preprints202512.2126.v1

Keywords: machine learning; information systems; natiural language processing; question-answer

 doi:10.20944/preprints202512.2126.v1

Review

An Overview of Recent Advances in Natural Language Processing for Information Systems

Douglas O'Shaughnessy

Institut National de la Recherche Scientifique; douglas.oshaughnessy@inrs.ca

Abstract

The crux of information systems is efficient storage and access to useful data by users. This paper is an overview of work that has advanced the use of such systems in recent years, primarily in machine learning, and specifically deep learning methods. Situating progress in terms of classical pattern recognition techniques for text, we review computational methods to process spoken and written data. Digital assistants such as Siri, Cortana, and Google Now exploit large language models and encoder-only transformer-based systems such as BERT. Practical tasks include Machine Translation, Information Retrieval, Text Summarization, Question-Answering, Sentiment Analysis, Natural Language Generation, Named Entity Recognition, and Relation Extraction. Issues to be covered include: post-training through alignment, parsing, and Reinforcement Learning.

Keywords: machine learning; information systems; natiural language processing; question-answer

1. Introduction

Recent years have seen considerable interest in facilitating computational access to information in databases. There is a huge demand for users to be able to access and manipulate the massive amount of data in various servers (e.g., the Cloud and the Internet of Things). Such Information can be in the form of text, images/video, audio (music and speech), and other data (e.g., from sensors). While the recent growth in such accessible data has been massive, ways to reliably and efficiently access this data have not always kept pace. Early question-and-answer systems were very simplistic and limited, with excessive constraints. Recent web search systems and digital assistants (e.g., Siri, Cortana, and Google Now) have become very efficient, but limitations remain.

Most people interact with information systems by queries, either textual (typed) or verbal (speech), using natural language to express their wishes. For efficient data access, such queries must be transformed into suitable representation forms to allow efficient interaction with stored data. Thus, the research area of natural language processing (NLP) is foremost in facilitating such access. NLP converts natural human output (speech or text) into forms that facilitate data access. Typical text includes documents, email, web pages, articles, reports, and blogs. For example, in the domain of answering a question, much useful natural language content in found textbooks, encyclopedias, guidelines, and electronic records.

The NLP field has numerous applications in business Intelligence and knowledge management; specific cases include Named Entity Recognition, Named Entity Linking, Coreference Resolution, Temporal Information Extraction, Relation Extraction, Knowledge Base Construction and reasoning, paraphrase detection, entailment recognition, discourse analysis, grounded language learning and image retrieval, computer-assisted translation, and grammatical error correction. One seeks to efficiently analyze text, speech and other data, to find relevant knowledge and structured information, and to extract salient facts about specific types of events, entities, and relationships.

This paper is an overview of recent advances in NLP to permit better access to stored information. We do not give an extensive survey of methods, as such has been well handled in recent papers [1–5]. Instead, we explain NLP methods without great technical or algorithmic detail, as this allows understanding by non-experts.

 doi:10.20944/preprints202512.2126.v1

2. Applications

There are many ways for access in information systems. In Information Retrieval, also called Information Extraction, a user seeks to obtain desired information from a computer system. One specific version is Question-Answering (QA) . Information systems can provide a list of documents or websites in response to a user query, e.g., what web-searchers Google or Bing do. NLP assists the initial steps via Natural Language Understanding (NLU), to convert inquiries to meta-forms to access the data, and then uses Natural Language Generation (NLG) to format the text or speech output to the user. We describe methods for NLU and NLG below.

Specific forms of NL interaction include: 1) Machine Translation (MT), where the input is speech/text in a user's (native) language and the output is in a different desired (target) language , 2) Automatic Text Summarization (ATS), which transforms a text into a more concise version , 3) Sentiment Analysis, where the affect or emotion present in some text or speech is estimated automatically , 4) text mining, where useful elements are retrieved from broader text , 5) automatic annotation of web pages .

As an example, NLG could access time-series data in an information system and produce a weather forecast or medical report generation . Summarization of such data could be content selection (what to say) , or surface realization (how to say it, i.e., selecting, ordering, and inflecting words) . The objective of content selection is to choose the correct information to output, for a given semantic input and communicative goal. For summarization, one seeks to reduce a given document in size, while retaining as much important and relevant information as possible, with minimal distortion . Some systems generate computer source code from natural language descriptions: specialized LLMs, such as CodeLLaMa and DeepSeek-Coder , as discussed below.

3. Methods for Question Answering

Given the text of a user question, one NLP procedure is to first attempt to understand the intent of the question. A common way is via forms of semantic parsing (shallow or deep) . A question is converted into a meaning representation, then mapped to database queries. This applies to both word sense discovery (acquisition of vocabulary) and word sense disambiguation (understanding) .

Semantic parsing maps natural language text or utterances into a suitable semantic representation, often based on some formalism or grammar. QA without semantic parsing may use SPARQL (Protocol and Query Language) queries on interlinked data represented by Resource Description Framework triples .

4. Language Models (LMs)

Many applications for information systems involve LMs, which are computational systems that can process natural language. Large LMs (LLMs) are extensive and complex models that learn very detailed statistical properties and semantics from vast amounts of training text. As with all artificial neural networks (ANNs) (section 8), LLMs are based on huge numbers of training examples, which provide the knowledge to understand and transform language.

The task of a LM is estimation of the probability of a specified sequence of text "tokens," i.e., short sequences of text, usually words, but sometimes shorter (e.g., byte-pairs) and sometimes sets of words. Simple LMs use N-gram statistics, i.e., probabilities of sequences of N successive words in ordinary text. As LMs grew in size and power (with advances in computer memory and speed), the training text used all sorts of available text on the internet and elsewhere (private companies also exploit their access to confidential data).

4.1. Basic N-Gram Models

Unigrams (N = 1) are individual words, and so a unigram probability is simply the general estimated likelihood of any given word among all possible text words (e.g., common "function" words such as a, an, the, with, for, etc. have relatively high probabilities, as they occur more often

 doi:10.20944/preprints202512.2126.v1

than "content" words such as nouns, verbs, adjectives, and adverbs). Bigrams are sequences of two successive words in text, trigrams uses 3 words, etc. One can readily see that using large values of N for N-grams uses exponentially large memory. However, it is also the case that natural language often has correlations for large values of N. For example, subject noun phrases and their corresponding verbs usually have high correlation (e.g., animals eat, fish swim, water flows). Many languages allow much correlation across many words in a sequence (e.g., in German, the verb occurs at the end of a sentence in general).

Older NLP system used tokenizers and part-of-speech taggers to identify the syntactic role of each word . More modern systems use neural networks to indirectly do such tasks.

A challenge of basic LMs is to reliably estimate probabilities of successive words that are generally conditioned on long text sequences. Use of N-grams tends to assume the validity of a Markov assumption, i.e., that probabilities can be approximated by limiting the range of conditional probabilities. Simply extending N-grams to large ranges of N not only risks exponential memory growth, but also severe under-training, despite the increasing availability of lots of training text data. There are various approaches to mitigate this problem, such as class LMs that group words into categories based on attributes such as color and size (e.g., rather than have separate individual stored likelihoods for "blue ball" and "red ball," use merged statistics for COLOR+ball). For more efficient LMs, one can cluster various word classes, or allow word sequences to skip some words .

When combining statistics of N-grams into a single global probability, one can use "back-off" principles, giving more weight to the N-grams of lower N, which are based on larger numbers of training examples (i.e., shorter contexts are observed more often, and suffer less from data sparsity). There are many variations for LMs : Good Turing estimator , deleted interpolation , Katz backoff , and Kneser-Ney smoothing .

Relations among words in a text go beyond simple statistics of word sequences such as N-grams. Relation extraction systems find closed-domain relationships. Open-domain relation extraction systems use specific phrases to define word relationships. To discover such relationships, sentences can be analyzed using techniques such as part-of-speech tagging, dependency parsers, or Named Entity Recognizers .

One method is multi-relational learning, which consists of: (1) statistical relational learning (SRL), such as Markov-logic networks, which directly encode multi-relational graphs using probabilistic models (Getoor et al, 2007); (2) path ranking methods, which explicitly explore the large relational feature space of relations with random walk ; and (3) embedding-based models, which embed multi-relational knowledge into low-dimensional representations of entities and relations via tensor/matrix factorization, Bayesian clustering framework, and neural networks .

4.2. Large Language Models

In recent years, research has developed LMs with much more complex systems called Large Language Models (LLMs) [30,31]. They can provide a very detailed understanding of text, using a limited window of focus. With advanced neural architecture, an LLM processes written instructions or questions from users and generates natural language text as output. After pre-training on large amounts of general information, LLMs have had great recent success in solving a wide range of information tasks by conditioning their models on a very small number of examples ("few-shot") or even only using instructions describing the task ("zero-shot"). Conditioning an LLM is called "prompting" (section 6), with either manual prompts or automatic ones.

LLMs use autoregressive Transformer neural architectures (section 8.2.3), where word tokens are iteratively predicted. On the other hand, alternative masked diffusion language models (MDLMs) are not constrained to generate data sequentially. In comparison to masked language models, the MDLM objective is a principled variational lower bound, and it supports NLG by ancestral sampling .

Pre-Trained Language Models (PLMs) such as BERT (Devlin et al, 2019) are commonly used in many applications due to their favourable performance. Other notable PLMs are open-source

 doi:10.20944/preprints202512.2126.v1

BLOOM , LLaMa-1/2 , OPT , and Falcon ; these have similar performance to closed "product" pre-trained LLMs such as GPT-3 and Chinchilla , ChatGPT, Google's BARD, PaLM , and Claude . The latter are heavily fine-tuned to align with human preferences, which greatly enhances their usability and safety (reliability), but incurs large cost in computation and human annotation. With (commercial) private training, data and model architecture details are generally not shared with the general public.

LLMs can do a huge amount of unsupervised pre-training on textual corpora, and then use a limited amount of supervised fine-tuning (SFT) on high-quality data from an appropriate domain, which may depend upon the application. This latter stage can be expensive, but is done to align the model with human preferences. Prime methods for this fine tuning are Reinforcement Learning (RL) from Human Feedback (RLHF; [40,41] and Direct Preference Optimization (DPO) . DPO directly optimizes using a loss function derived from the classic BT model . RLHF learns from training environments through interaction and rewards , and has three stages: SFT, reward modeling, and RL. Other methods are: syntax fine-tuning, knowledge preservation fine-tuning, and task-oriented fine-tuning .

As ANNs are almost universally based on huge numbers of examples and their huge networks are impossible to debug, there has been a recent effort toward "explainable AI," which may eventually allow ANNs that can be interpreted . As of now, however, LLMs sometimes generate incorrect information (though otherwise plausible), known as "hallucinations" .

5. Semantic Role Labeling (SRL)

Semantic hierarchies can offer ways to organize textual (NLP) knowledge.

SRL is used to identify predicates and arguments in text, and to label their semantic roles in sentences. This may involve joint classification of semantic arguments [48,49], feature engineering for SRL [50,51], or WordNet . In the ontology YAGO , categories in Wikipedia are linked onto WordNet, which is limited by the scope of Wikipedia. These operations often involve automatic discovery of *hypernym* (word with a broad meaning or class) versus *hyponym* (more specific meaning; subordinate in a class) relations, e.g., "is-a" relations ("X is a dog").

6. Prompts

NLP often uses fine tuning to raise model performance by use of instructions ("prompts"), without needing to modify existing model parameters . In "few-shot prompting," one provides a few examples to the model for a specific task. LLMs can be fed with step-by-step reasoning examples, e.g., Chain-of-Thought (CoT) prompting , Automatic Prompt Engineer , and Chain of Code . One can tune such systems by appending learnable tokens to the input of the model. CoT decomposes complex problems into smaller sub-problems, then solves them with natural language reasoning steps. This works best if the knowledge to answer an input question exists in its context or in the model's parameters (e.g., common sense reasoning, learned from much pre-training). Some NLP approaches combine reasoning and acting with LLMs to handle language reasoning and decision making tasks (e.g., ReAct).

Prompting is commonly used to improve LLMs for downstream applications. One tries to employ well-designed text prompt sequences in an LLM's input. Finding an effective prompt from a small training set for a specific task is a discrete optimization problem with a large search space, and classic gradient-based optimization methods are not always useful.

Recent advancements in prompt optimization include: (1) discrete representation-based greedy methods and (2) soft representation-based gradient methods. They leverage gradient information to accelerate prompt optimization. Hard representation methods directly use discrete optimization by using hard tokens as intermediate representations: 1) AutoPrompt uses a token search strategy based on gradients to locate useful prompts, 2) Greedy Coordinate Gradient uses a greedy strategy

 doi:10.20944/preprints202512.2126.v1

to advance optimization. Projected gradient descent methods map a soft prompt to the nearest token in the vocabulary at each step of regular gradient descent.

7. Distance Metrics for NLP

There are several ways to measure success in training for information systems. Mean Absolute Error has been used for regression tasks , whereas for classification tasks, one generally uses Precision, Recall and F1-score . For recommendation tasks, Mean Reciprocal Rank is in common usage. NLG tasks find metrics like BLEU .

The reward model (RM) is a metric that estimates the degree to which model outputs align with human preferences. RMs typically follow the classic B-T model , but one also uses regression paradigms and the "LLM as a judge" approach . The Area Under the Receiver Operating Characteristic Curve is a common threshold-independent metric . Human coding metrics include the Jaccard Score .

One can measure the performance of a topic model by the log-likelihood of a model on held-out test documents, i.e., its predictive accuracy. Another measure is perplexity, which is inversely proportional to average log-likelihood. Other common measures are topic coherence and topic diversity . For coherence, point-wise mutual information measures the proximity of words in each topic based on relative co-frequencies with each other. Topic diversity uses the percentage of unique words in a topic set.

8. Neural Networks in Information Systems

As with many computer applications in recent years, artificial neural networks (ANNs) have come to dominate the field of information systems. At the cost of significant computer resources, ANNs provide an automatic processing tool to accomplish a wide range of artificial intelligence tasks, including all those mentioned above for NLP. While termed "neural," such processing algorithms are only very loosely associated with the natural nervous system found in humans and other living beings. The elemental basis of both natural and artificial neural systems is the neuron (or called a mathematical node in ANNs), which transforms a weighted set of electrical inputs (from other neurons) to a single output in a nonlinear operation. By arranging huge numbers of such neurons/nodes in increasingly complex architectures, arbitrarily complex tasks can be accomplished.

The basic ANN feedforward architecture ("multi-layer perceptron") arranges neurons in layers, whereby the outputs of one layer feed into a successive (higher) layer, and so forth. The initial lowest layer accepts input data from sensors, and the highest layer can provide a classification of the input into a set of predetermined classes (i.e., do pattern recognition). Alternatively, the output can be a set of data that corresponds to a transformation of the input data (e.g., text-to-speech synthesis, or NLG). The nonlinear processing in ANNs allows analysis and synthesis of complex data patterns, but needs huge network models and much computation. The term deep neural network (DNN) refers to an ANN with several layers of nodes, rather than shallow ANNs such as support vector machines . Intermediate "hidden" layers in an ANN refine features of analysis (e.g., as data progresses, layer-to-layer, in a well-trained ANN, the outputs at each layer gradually move toward more relevant features for a given task). Numbers of nodes in layers vary greatly, and are often chosen empirically.

8.1. Fundamentals of ANNs

For NLP classification such as text interpretation, the training of one node in an ANN can be viewed as optimizing the placement of hyperplane boundaries for class regions in a representation space. For a neuron in biological (natural) neural systems, dendrites route electrical signals to succeeding axons, each of which then yields a binary output consisting of a brief pulse in time when the weighted sum of its inputs exceeds a threshold. The output of an ANN node follows an activation function $y = \varphi(w \cdot x + b)$, where w is a N-dimensional weight vector, b is a scalar bias, x is a N-

 doi:10.20944/preprints202512.2126.v1

dimensional set of inputs for the node, $\varphi(.)$ is a nonlinearity (e.g., sigmoid, rectifier, or tanh). For each node, its parameters specify the location of a hyperplane in a virtual space, for which the node's binary output chooses either side of the hyperplane. Huge DNNs, with many millions of parameters, are often used to attain enough precision for a desired task.

Training parameters in an ANN starts from initial estimates chosen randomly (or pre-trained, using much unlabeled data). Iterative training alters these parameters in incremental fashion, while minimizing a loss (or cost) function. Direct minimization of accuracy for any specific task is not feasible, because ANN training requires to have a differentiable loss, to allow a product chain of derivatives. This shows the direction and amount to alter parameters at each training iteration, using a typical steepest gradient descent approach. Loss functions approximate a penalty for ANN classification errors. They vary greatly across applications, but a common one is cross-entropy , which uses the log-likelihood of training data, and corresponds to least squares error (L2 norm).

8.2. Common ANN Architectures

In a basic ANN, all nodes in each layer provide their outputs to all nodes in the next layer. Each connection requires a multiply-and-add operation (hence, the popularity of graphical processing units, which specialize in such calculations).

Such a general approach is usually excessively costly (in memory and computation). The useful information in most data (including text) is distributed very non-uniformly in its dimensions (e.g., in time). Thus, most applications use versions of the more efficient ANN components discussed below.

8.2.1. Convolutional Neural Networks (CNNs)

For many tasks, relevant information in an input data sequence is largely localized (i.e., found in limited spans); e.g., in images, objects have useful edges that occupy a small percentage of the full data range. In addition, in localized regions, data may display pseudo-random variations. Such cases suggest use of filters to exploit these factors, to better accomplish practical tasks. It is also common to view data as a tensor (e.g., a matrix, if the data are conveniently viewed in two dimensions). For example, while audio data such as speech is a signal showing air pressure as a function of time, a spectral display notes energy as a function of both time and frequency; smoothing edges of a wide-band spectrogram is useful in ASR. In such cases, 2-D data are multiplied by a square weight matrix over a small range (e.g., 3x3), and results are summed (*pooled*) . One may also apply 1-dimensional convolution to vectors, in cases such as text, which do not benefit from a 2-D display. CNNs thus usually have alternating layers of "convolution" (weighted multiply-and-add filters), with pooling layers that reduce the size of the vector from the previous layer, and thus do data reduction.

8.2.2. Recurrent Neural Networks (RNNs)

To further take advantage of the uneven distribution of pertinent data across dimensions, nodes in ANNs can be more properly weighted across time. For example, CNNs usually exploit data correlations over small local ranges, but recurrence can be helpful to utilize significant correlations over long ranges. RNNs have neural models where data is fed back from earlier layers, and use distributed hidden states that store information about past input . Specialized network gates (called input, forget, and output) control the flow of data across layers. Typical RNNs are termed long short-term memory and residual network.

8.2.3. Attention

A major recent development in the field of ANNs is called *attention* - a way to improve performance by emphasizing weights for specific sections of data; this is another method to exploit the non-uniform distribution of information. The focus for attention is determined mathematically as correlation among data using matrix operations (e.g., multiplications and sums) that use *queries*

 doi:10.20944/preprints202512.2126.v1

(inputs), *keys* (features), and *values* (desired outputs). The queries come from a decoder layer earlier in an ANN, and keys and values come from encoder outputs. Attention can be readily applied across various dimensions including time, frequency, and ANN layers. While the principle of such correlation-based attention is sound, the simple mechanisms of correlation that are used are often inadequate to exploit well the complex information distribution in data.

Neural models that use attention are often called *Transformers* . In recurrent networks (RNNs), inputs are processed in sequence. Transformers do not follow sequences, instead using an embedding table with time-positional encodings. These latter use non-linear functions for monotonic temporal mappings.

A major disadvantage of Transformers is their need for more computation than other ANN approaches. Transformers can be quadratic in the length of data sequences. Transformer architecture can learn general structural information from high-dimensional data, and exploit the huge amount of unlabeled data (available generally) through self- and un-supervised learning.

8.3. Neural LLM Architectures

Many information systems that use analysis and synthesis (e.g., of text) apply an approach that involves coding and decoding data. (This is analogous to inverse operations in radio transmission.) In an ANN approach for such an *encoder/decoder* structure, the initial network layers act as an encoder to automatically learn hidden latent features for data in a compressed representation, and then ensuing layers act as decoder to form a reconstructed data form. In many audio and video coding schemes, the decoder steps often proceed in inverse fashion to the encoder steps. When this encoder-decoder is trained on unlabelled data, it is called an *autoencoder.*

Many neural systems use deep generative models try to learn complex data distributions through unsupervised training. The so-called GAN (adversarial) architecture uses two networks, generator and discriminator, in a minimum-maximum operation. The generator produces data from a low-dimensional latent representation space, often starting from a simple Gaussian noise vector. The discriminator learns to distinguish between "real" training data and "false" generator outputs. The criterion for generators is to maximize the discriminator's error rate, while discriminators try to minimize their error rate.

The encoder processes and encodes an input sentence into a "hidden" representational form, which attempts to find relationships between words and the overall textual context. For example, the common module BERT uses only an encoder (no decoder) with bi-directional attention.

In an autoencoder, the decoder utilizes the hidden space of the system to generate a target output text; it converts the abstract representation into specific contextually-relevant expressions; examples are: PLBART , CodeT5 , AlphaCode , and CoTexT .

For many applications, it is not necessary to use both an encoder and a decoder. For example, GPT uses only a decoder; indeed, most current LLMs are decoder-only with causal attention, i.e., earlier tokens in a text do not attend to ensuing tokens. Other decoder-only systems include: Google Gemini and Claude.

9. Sentence Models

Distributional semantic models (DSMs) use vectors that keep track of text contexts (e.g., co-occurring words) in which target terms appear in a large corpus as proxies for meaning representations, and apply geometric techniques to these vectors to measure the similarity in meaning of the corresponding words .

A general class of basic sentence models is that of Neural Bag-of-Words (NBoW) models . These generally consist of a projection layer that maps words, sub-word units or N-grams to high dimensional embeddings; the latter are then combined as components with an operation such as summation. Large-scale word embeddings can provide context for topic models (next section). The Bag-of-Words approach ignores the sequential ordering of words in text.

 doi:10.20944/preprints202512.2126.v1

10. Topic Models

In NLP, a topic model attempts to obtain a general idea of the topics (subjects) of an input text . Topic models use unsupervised machine learning to compress many documents into a short summary that captures the most prevalent subjects in a corpus. When applied to a set of documents, a topic model estimates a set of underlying (latent) topics, each of which describes a semantic concept. Early topic models used Latent Semantic Indexing (LSI) or PLSA [86,87]: using a word-document matrix, a singular value decomposition reduces the dimensionality. Another approach is Latent Dirichlet Allocation (LDA), which uses the Dirichlet distribution, a bag-of-words model, and the same document-term matrix as in LSI . In LDA, a topic is a distribution of words, where each word in a text comes from a mixture of multinomial distributions with Dirichlet as the prior.

Before neural approaches, most topic models were probabilistic graphical models (such as LDA) or non-negative matrix factorization (NNMF). The former model the document generation process with topics as latent variables and model parameters through Variational Inference or Monte Carlo Markov Chain methods like collapsed Gibbs sampling . In probabilistic generative models, model random variables are assumed to be come from certain prior distributions. NNMF directly finds topics via decomposition of a term-document matrix into two low-rank factor matrices.

A simple topic model is the “bag-of-words” model, which represents a document by a vector of word counts, across the vocabulary of all possible words, by the numbers of their occurrences in the text. In graph-based approaches , words are nodes and their co-occurrences use weighted edges. Large-scale word embeddings can provide context for the topic models; they represent words as continuous vectors in a low-dimensional space.

Neural Topic Models (NTMs) use DNNs to model distributions for topic models . Unlike earlier topic models, NTMs can estimate model parameters through automatic gradient back-propagation by adopting deep neural networks to model latent topics, such as Variational Autoencoder . This flexibility enables adjustments to model structures to fit diverse application scenarios. They typically use the re-parameterization trick of Gaussian distributions. In addition, NTMs can handle large-scale datasets via use of parallel computing facilities like GPUs.

One can use LLMs to improve topic modeling. Topic models are largely based on word co-occurrences to infer latent topics, but such information is scarce in short texts such as tweets (i.e., problem of data sparsity). Probabilistic topic models, such as PLSA and LDA, work well on long texts. For short text topic modeling, Biterm Topic Model (BTM) and Dirichlet Multinomial Mixture (DMM) model are common approaches. BTM directly constructs the topic distributions over unordered word-pairs (biterms), while DMM applies auxiliary pre-trained word embeddings to introduce external information from other sources; both use classic Bayesian inference. Contrastive learning is also used for topic models .

Several extensions based on BTM and DMM have also been proposed, such as Generalized Polya Urn-DMM with word embeddings and Multiterm Topic Model . Besides, Semantics-assisted Non-negative Matrix Factorization was lately proposed as an NMF topic model incorporating word-context semantic correlations solved by a block coordinate descent algorithm.

A recent method called Adversarial-neural Topic Model (ATM) uses a generator network to learn a projection function between a document-topic distribution and a document-word distribution. A discriminator network determines if an input document is real or fake and its output signal helps the generator to construct a more realistic document from a random noise drawn from a Dirichlet distribution.

11. Fine Tuning LLMs

The important step of fine-tuning (transfer learning) for LLMs has several versions: prefix, adapter, task-oriented, and parameter-efficient.

In prefix tuning, trainable tokens (soft prompts) are added to both the input and internal layers ; their parameters are shared. In these decoder-only models, instead of a causal mask, a fully visible

 doi:10.20944/preprints202512.2126.v1

mask is used for the prefix part of the input sequence, and a causal mask is used for the target sequence.

With adapter tuning, one adds small neural network modules to the original model, then fine-tunes them on specific tasks, thus fine-tuning a few external parameters instead of entire LLMs .

As for task-oriented fine-tuning , Parameter Efficient Fine-Tuning optimizes the subset of parameters fine-tuned, thereby reducing the overall computational complexity. One version of this is Low-Rank Adaptation, which freezes pre-trained model weights and inserts trainable rank decomposition matrices into each layer of the Transformer architecture, which greatly reduces the trainable parameters for downstream tasks .

Most LLMs base their training on fixed sets of data, which can limit performance on specific or dynamic topics. A recent approach called Retrieval Augmented Generation (RAG) integrates real-time knowledge retrieval with LLM generation . RAG uses a neural retriever to find the best text sections by dense similarity, which are concatenated with the query and conditioned into a NLG component. RAG integrates real-time knowledge retrieval with LLM generation, which secures outputs in specific data from current domains.

12. Meaning Representation

A major objective in information systems is to extract meaning from text. One method for this uses distributional semantics. Distributional semantic models (DSMs) employ vectors that follow context where target words in large corpora serve as proxies for meaning representations . They use geometric techniques to measure similarity in the meaning of corresponding words . This follows the Distributional Hypothesis , where words with similar linguistic contexts are likely to have similar meanings. DSMs represent lexical entries using vectors (embeddings) that demonstrate their distribution in text corpora.

Early DSMs established distributional vectors with word co-occurrence frequencies. Then, prediction DSMs learned vectors with shallow ANNs for local surrounding words. Now, contextual DSMs use deep NTMs to generate contextualized vectors for words. Contextual embeddings (e.g., ELMo and BERT) improve on global word representations (e.g., Word2Vec) [1,108].

13. Challenges and Benchmarks

To motivate uniform research in the field, several technical meetings and efforts have been organized. One of the first was the Text REtrieval Conference (TREC), of which there were several in the 1990s . A later one was the Cross Language Evaluation Forum .

The Document Understanding Conference examined query-focused multi-document summarization since 2004; focus on complex queries with specific answers . A preposition disambiguation task was focus in SemEval 2007 .

Benchmarks or datasets to assess language models 'reasoning:

- CriticBench evaluates a LM's capability to critique solutions and correct mistakes in reasoning tasks.
- MathCheck synthesizes solutions containing erroneous steps using the GSM8K dataset .
- PRM800K builds on the MATH problems .
- process reward models explicitly assess and guide reasoning at the step or trajectory level.
- The Stanford Question Answering Dataset and DROP are often used as reading comprehension benchmarks.

 doi:10.20944/preprints202512.2126.v1

- HellaSwag advances common sense NL inference via use of Adversarial Filtering, which is a data collection paradigm where discriminators iteratively select an adversarial set of machine-generated wrong answers.

- Flan : a large publicly available set of tasks and methods for instruction tuning.

Older knowledge bases include:

- DBPedia

- Freebase

- Yago2

- FrameNet

- Propbank , a large hand-annotated text corpus.

- Other possibilities include use of expert annotators, or through crowdsourcing (e.g., Amazon's Mechanical Turk).

Early software include: MetaMap to identify concepts in text.

SEMREP can detect some relations using hand-crafted rules. As for commercial systems, we have Google Now (with Google's Knowledge Graph), and Facebook Graph Search . An earlier system was the Unified Medical Language System , having 2.7 million concepts from over 160 source vocabularies. Distant Supervision has a knowledge base such as Wikipedia or Freebase that is used to automatically tag training examples from the text corpora .

In a related direction, some interactions with databases involve sentence compression, where an input text is transformed into shorter output text, while retaining the general meaning (i.e., omitting lesser detail). Examples are: Ziff-Davis corpus and the Edinburgh compression corpora .

14. Computational Issues

Major factors in all information systems are their computational requirements, i.e., storage space and mathematical operations. In general, the trend has been toward massive systems. For example, LLaMa-2 uses 13 billion parameters (e.g., multiply-and-adds). Fine-tuning a 16-bit LLaMa model with 65 billion parameters requires more than 780 gigabytes of memory . The massive recent progression in model size is readily seen in GPT: GPT-1 had 117 million parameters; GPT-2 1.5 billion, and further to GPT-3 175 billion .

The pre-trained model CodeBERT has a total of 125M parameters, resulting in a model size of 476 MB. Recently proposed models like Codex and CodeGen have over 100 billion parameters and over 100 GB in size.

Training a relatively smaller model like the PolyCoder (2.7 billion paramters) , employing eight NVIDIA RTX 8000 GPUs on a single machine, requires about 6 weeks to train.

15. Conclusions

This overview examined recent developments in information systems, in the context of classical approaches. As with many application areas of artificial intelligence, the field has become dominated by deep learning methods, as they provide rapid and reasonable solutions, at the cost of significant computer memory and computation needs.

References

1. Liu, Q.; Kusner, M.J.; Blunsom, P.; A survey on contextual embeddings. **2020,** *arXiv:2003.07278*.
2. Zhao, H.; Phung, D.; Huynh, V.; Jin, Y.; Du, L.; Buntine, W.; Topic modelling meets deep neural networks: A survey. **2021,** *arXiv:2103.00498*.

 doi:10.20944/preprints202512.2126.v1

3. Long, L.; Wang, R.; Xiao, R.; Zhao, J.; Ding, X.; Chen, G.; Wang, H.; On LLMs-driven synthetic data generation, curation, and evaluation: A survey. **2024,** *arXiv:2406.15126*.
4. Tan, Z.; Li, D.; Wang, S.; Beigi, A.; Jiang, B.; Bhattacharjee, A.; Karami, M.; Li, J.; Cheng, L.; Liu, H.; Large language models for data annotation and synthesis: A survey. **2024,** *arXiv:2402.13446*.
5. Wu, X.; Nguyen, T.; Luu, A.T.; A survey on neural topic models: methods, applications, and challenges. *Artificial Intelligence Review*, **2024,** *57*(2), p.18.
6. Peñas, A.; Magnini, B.; Forner, P.; Sutcliffe, R.; Rodrigo, A.; Giampiccolo, D.; Question answering at the cross-language evaluation forum 2003–2010. *Language resources and evaluation*, **2012,** *46*(2), 77-217.
7. Wang, H.; Wu, H.; He, Z.; Huang, L.; Church, K.W.; Progress in machine translation. *Engineering*, **2022,** *18*, 143-153.
8. El-Kassas, W.S.; Salama, C.R.; Rafea, A.A.; Mohamed, H.K.; Automatic text summarization: A comprehensive survey. *Expert systems with applications*, **2021,** *165*, 113679.
9. Wankhade, M.; Rao, A.C.S.; Kulkarni, C.; A survey on sentiment analysis methods, applications, and challenges. *Artificial Intelligence Review*, **2022,** *55*(7), 5731-5780.
10. Topaz, M.; Murga, L.; Gaddis, K.M.; McDonald, M.V.; Bar-Bachar, O.; Goldberg, Y.; Bowles, K.H.; Mining fall-related information in clinical notes: Comparison of rule-based and novel word embedding-based machine learning approaches. *Journal of biomedical informatics*, **2019,** *90*, 103103.
11. Angeli, G.; Liang, P.; Klein, D.; A simple domain-independent probabilistic approach to generation. *Conference on Empirical Methods in Natural Language Processing*, **2010,** 502-512).
12. Lee, S.H.; Natural language generation for electronic health records. *NPJ digital medicine*, **2018,** *1*(1), p.63.
13. Zhang, H.; Cai, J.; Xu, J.; Wang, J.; Pretraining-based natural language generation for text summarization. *Conference on computational natural language learning*, **2019,** 789-797.
14. Mille, S.; Belz, A.; Bohnet, B.; Graham, Y.; Pitler, E.; Wanner, L.; The first multilingual surface realisation shared task (SR'18): Overview and evaluation results. *Workshop on Multilingual Surface Realisation*, **2018,** 1-12.
15. Van Veen, D.; Van Uden, C.; Blankemeier, L.; Delbrouck, J.B.; Aali, A.; Bluethgen, C.; Pareek, A.; Polacin, M.; Reis, E.P.; Seehofnerová, A.; Rohatgi, N.; Adapted large language models can outperform medical experts in clinical text summarization. *Nature medicine*, **2024,** *30*(4), 1134-1142.
16. Roziere, B.; Gehring, J.; Gloeckle, F.; Sootla, S.; Gat, I.; Tan, X.E.; Adi, Y.; Liu, J.; Sauvestre, R.; Remez, T.; Rapin, J.; Code LLaMa: Open foundation models for code. **2023,** *arXiv:2308.12950*.
17. Guo, D.; Zhu, Q.; Yang, D.; Xie, Z.; Dong, K.; Zhang, W.; Chen, G.; Bi, X.; Wu, Y.; Li, Y.K.; Luo, F.; DeepSeek-Coder: When the Large Language Model Meets Programming--The Rise of Code Intelligence. **2024,** *arXiv:2401.14196*.
18. Berant, J.; Chou, A.; Frostig, R.; Liang, P.; Semantic parsing on freebase from question-answer pairs. C*onference on empirical methods in natural language processing*, **2013,** 1533-1544.
19. Ide, N.; Véronis, J., Introduction to the special issue on word sense disambiguation: the state of the art. *Computational linguistics*, **1998**, *24*(1), 1-40.
20. Pan, J.Z.; Resource description framework. *Handbook on ontologies* **2009,** 71-90; Berlin, Heidelberg: Springer.
21. Mielke, S.J.; Alyafeai, Z.; Salesky, E.; Raffel, C.; Dey, M.; Gallé, M.; Raja, A.; Si, C.; Lee, W.Y.; Sagot, B.; Tan, S.; Between words and characters: A brief history of open-vocabulary modeling and tokenization in MLP. **2021**, *arXiv:2112.10508*.
22. Ney, H.; Essen, U.; Kneser, R.; On structuring probabilistic dependences in stochastic language modelling. *Computer Speech & Language*, **1994,** *8*(1), 1-38.
23. Chen, S.F.; Goodman, J.; An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, **1999,** *13*(4), 359-394.

 doi:10.20944/preprints202512.2126.v1

24. Good, I.J.; The population frequencies of species and the estimation of population parameters. *Biometrika*, **1953,** 40(3-4):237–264.
25. Jelinek, F.; Mercer, R.L.; "Interpolated estimation of Markov source parameters from sparse data", in E.S. Gelsema and L.N. Kanal (eds.), Pattern Recognition ill Practice, **1980,** 381-397. North-Holland, Amsterdam.
26. Katz, S.; Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on acoustics, speech, and signal processing*, **2003,** *35*(3), 400-401.
27. Kneser, R.; Ney, H.; Improved backing-off for m-gram language modeling. *International conference on acoustics, speech, and signal processing*, **1995,** 181-184.
28. Kühnel, L. and Fluck, J., We are not ready yet: limitations of state-of-the-art disease named entity recognizers. *Journal of Biomedical Semantics*, **2022**, *13*(1), 26.
29. Zhang, X.; Zhan, K.; Hu, E.; Fu, C.; Luo, L.; Jiang, H.; Jia, Y.; Yu, F.; Dou, Z.; Cao, Z.; Chen, L.; Answer complex questions: Path ranker is all you need. *International ACM SIGIR Conference on Research and Development in Information Retrieval*, **2021,** 449-458.
30. OpenAI. GPT-4 technical report, **2023.**
31. Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; Bikel, D.; LLaMa 2: Open foundation and fine-tuned chat models. **2023,** *arXiv:2307.09288*.
32. Sahoo, S.; Arriola, M.; Schiff, Y.; Gokaslan, A.; Marroquin, E.; Chiu, J.; Rush, A.; Kuleshov, V.; Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, **2024,** *37*, 130136-130184.
33. Scao, T.L.; Fan, A.; Akiki, C.; Pavlick, E.; Ilić, S.; Hesslow, D.; Castagné, R.; Luccioni, A.S.; Yvon, F.; Gallé, M.; Bloom: A 176b-parameter open-access multilingual language model. **2022,** *arXiv:2211.05100*.
34. Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X.V.; Mihaylov, T.; OPT: Open pre-trained transformer language models. **2022,** *arXiv:2205.01068*.
35. Penedo, G.; Malartic, Q.; Hesslow, D.; Cojocaru, R.; Cappelli, A.; Alobeidli, H.; Pannier, B.; Almazrouei, E.; Launay, J.; The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only; **2023;** *arXiv:2306.01116*.
36. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Language models are few-shot learners. *Advances in neural information processing systems*, **2020,** *33*, 1877-1901.
37. Hoffmann, J.; Borgeaud, S.; Mensch, A.; Buchatskaya, E.; Cai, T.; Rutherford, E.; Casas, D.D.L.; Hendricks, L.A.; Welbl, J.; Clark, A.; Hennigan, T.; Training compute-optimal large language models. **2022,** *arXiv:2203.15556*.
38. Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H.W.; Sutton, C.; Gehrmann, S.; Schuh, P.; PaLM: Scaling language modeling with pathways. *Journal of Machine Learning Research*, **2023,** *24*(240), 1-113.
39. Anthropic, P.B.C.; Introducing Claude. **2023,** March, 14, 2023.
40. Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; Schulman, J.; Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, **2022,** *35*, 27730-27744.
41. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O.; Proximal policy optimization algorithms. **2017,** *arXiv:1707.06347*.
42. Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C.D.; Ermon, S.; Finn, C.; Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, **2023,** *36*, 53728-53741
43. Bradley, R.A.; Terry, M.E.; Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, **1952**, *39*(3/4), 324-345.

 doi:10.20944/preprints202512.2126.v1

44. Yuan, W.; Pang, R.Y.; Cho, K.; Li, X.; Sukhbaatar, S.; Xu, J.; Weston, J.E.; Self-rewarding language models. *International Conference on Machine Learning,* **2024.**
45. Dong, G.; Yuan, H.; Lu, K.; Li, C.; Xue, M.; Liu, D.; Wang, W.; Yuan, Z.; Zhou, C.; Zhou, J.; How abilities in large language models are affected by supervised fine-tuning data composition. *Annual Meeting of the Association for Computational Linguistics;* **2024,** 177-198.
46. Zhao, H.; Chen, H.; Yang, F.; Liu, N.; Deng, H.; Cai, H.; Wang, S.; Yin, D.; Du, M.; Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, **2024,** *15*(2),1-38.
47. Zhang, Y.; Li, Y.; Cui, L.; Cai, D.; Liu, L.; Fu, T.; Huang, X.; Zhao, E.; Zhang, Y.; Chen, Y.; Wang, L.; Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models**,** *Computational Linguistics*, **2025,** 1-46.
48. Toutanova, K.; Haghighi, A.; Manning, C.D.; A global joint model for semantic role labeling. *Computational Linguistics*, **2008,** *34*(2), 161-191.
49. Johansson, R.; Nugues, P.; Dependency-based syntactic–semantic analysis with PropBank and NomBank. *Conference on Computational Natural Language Learning,* **2008,** 183-187.
50. Zhou, J.; Xu, W.; End-to-end learning of semantic role labeling using recurrent neural networks. *Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing,* **2015**, 1127-1137).
51. Björkelund, A.; Hafdell, L.; Nugues, P.; Multilingual semantic role labeling. *Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task,* **2009,** 43-48.
52. Miller, G.A.; WordNet: a lexical database for English. *Communications of the ACM,* **1995,** *38*(11), 39-41.
53. Suchanek, F.M.; Kasneci, G.; Weikum, G.; Yago: A large ontology from Wikipedia and WordNet. *Journal of Web Semantics*, **2008,** *6*(3), 203-217.
54. Guo, Q.; Wang, R.; Guo, J.; Li, B.; Song, K.; Tan, X.; Liu, G.; Bian, J.; Yang, Y.; Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. **2023,** *arXiv:2309.08532*.
55. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q.V.; Zhou, D.; Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, **2022,** *35*, 24824-24837.
56. Zhou, Y.; Muresanu, A.I.; Han, Z.; Paster, K.; Pitis, S.; Chan, H.; Ba, J.; Large language models are human-level prompt engineers. *International conference on learning representations,* **2022.**
57. Li, C.; Liang, J.; Zeng, A.; Chen, X.; Hausman, K.; Sadigh, D.; Levine, S.; Fei-Fei, L.; Xia, F.; Ichter, B.; Chain of code: Reasoning with a language model-augmented code emulator. **2023,** *arXiv:2312.04474*.
58. Yao, S.; Zhao, J.; Yu, D., Du, N.; Shafran, I.; Narasimhan, K.R.; Cao, Y.; ReAct: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.
59. Shin, T.; Razeghi, Y.; Logan IV, R.L.; Wallace, E.; Singh, S.; Autoprompt: Eliciting knowledge from language models with automatically generated prompts. **2020,** *arXiv:2010.15980*.
60. Zou, A.; Wang, Z.; Carlini, N.; Nasr, M.; Kolter, J.Z.; Fredrikson, M.; Universal and transferable adversarial attacks on aligned language models. **2023,** *arXiv:2307.15043*.
61. Tian, J.; He, Z.; Dai, X.; Ma, C.Y.; Liu, Y.C.; Kira, Z.; Trainable projected gradient method for robust fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,* **2023,** 7836-7845).
62. Willmott, C.J.; Matsuura, K.; Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate research*, **2005,** *30*(1), 79-82.
63. Yacouby, R.; Axman, D.; Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. *Workshop on evaluation and comparison of NLP systems,* **2020,** 79-91.

 doi:10.20944/preprints202512.2126.v1

64. Chapelle, O.; Metlzer, D.; Zhang, Y.; Grinspan, P.; Expected reciprocal rank for graded relevance. *ACM conference on Information and knowledge management,* **2009,** 621-630.
65. Reiter, E.; A structured review of the validity of BLEU. *Computational Linguistics,* **2018,** *44*(3), 393-401.
66. Stiennon, N.; Ouyang, L.; Wu, J.; Ziegler, D.; Lowe, R.; Voss, C.; Radford, A.; Amodei, D.; Christiano, P.F.; Learning to summarize with human feedback. *Advances in neural information processing systems*, **2020,** *33*, 3008-3021.
67. Bradley, R.A.; Terry, M.E.; Rank analysis of incomplete block designs: I. the method of paired comparisons. Biometrika, **1952,** 39:324–345.
68. Wang, C.; Deng, Y.; Lyu, Z.; Zeng, L.; He, J.; Yan, S.; An, B.; Q*: Improving multi-step reasoning for LLMs with deliberative planning. **2024, a***rXiv:2406.14283*.
69. Zheng, L.; Chiang, W.L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E.; Zhang, H.; Judging LLM-as-a-Judge with MT-Bench and chatbot arena. *Advances in neural information processing systems*, **2023,** *36*, 46595-46623.
70. Hanley, J.A.; McNeil, B.J.; The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, **1982,** *143*(1), 29-36.
71. Singh, R.; Singh, S.; Text similarity measures in news articles by vector space model using NLP. *Journal of The Institution of Engineers (India): Series B*, **2021,** *102*(2), 329-338.
72. Wahba, Y.; Madhavji, N.; Steinbacher, J.; A comparison of SVM against pre-trained language models (PLMs) for text classification tasks. *International Conference on Machine Learning, Optimization, and Data Science, Springer Nature Switzerland.* **2022**, 304-313.
73. . Yin, J.; Wang, J.; A Dirichlet multinomial mixture model-based approach for short text clustering. *ACM SIGKDD international conference on Knowledge discovery and data mining,* **2014,** 233-242.
74. Yin, W.; Kann, K.; Yu, M.; Schütze, H.; Comparative study of CNN and RNN for natural language processing. **2017,** *arXiv:1702.01923*.
75. So, D.; Mańke, W.; Liu, H.; Dai, Z.; Shazeer, N.; Le, Q.V.; Searching for efficient transformers for language modeling. *Advances in neural information processing systems,* **2021,** *34*, 6010-6022.
76. Cheng, J.; Yang, Y.; Tang, X.; Xiong, N.; Zhang, Y.; Lei, F.; Generative adversarial networks: A literature review. *KSII Transactions on Internet & Information Systems*, **2020,** *14*(12).
77. Ahmad, W.; Chakraborty, S.; Ray, B.; Chang, K.W.; Unified pre-training for program understanding and generation. *Conference of the North American chapter of the association for computational linguistics: human language technologies,* **2021,** 2655-2668.
78. Wang, Y.; Wang, W.; Joty, S.; Hoi, S.C.; Codet-5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. **2021,** *arXiv:2109.00859*. Rosenfeld, R.; A maximum entropy approach to adaptive statistical language modelling. *Computer speech and language*, **1996,** *10*(3), 187.
79. Li, Y.; Choi, D.; Chung, J.; Kushman, N.; Schrittwieser, J.; Leblond, R.; Eccles, T.; Keeling, J.; Gimeno, F.; Dal Lago, A.; Hubert, T.; Competition-level code generation with alphacode. *Science*, **2022,** *378*(6624), 1092-1097.
80. Pham, C.M.; Hoyle, A.; Sun, S.; Resnik, P.; Iyyer, M.; 2024,. Topicgpt: A prompt-based topic modeling framework. *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* **2024,** 2956-2984.
81. Imran, M.; Almusharraf, N.; Google Gemini as a next generation AI educational tool: a review of emerging educational technology. *Smart Learning Environments*, **2024,** *11*(1), 22.
82. Lenci, A.; Sahlgren, M.; Jeuniaux, P.; Cuba Gyllensten, A.; Miliani, M.; A comparative evaluation and analysis of three generations of Distributional Semantic Models. *Language resources and evaluation*, **2022,** *56*(4), 1269-1313.
83. Iyyer, M.; Manjunatha, V.; Boyd-Graber, J.; Daumé III, H.; Deep unordered composition rivals syntactic methods for text classification. *Annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing,* **2015,** 1681-1691).

 doi:10.20944/preprints202512.2126.v1

84. Churchill, R.; Singh, L.; The evolution of topic modeling. *ACM Computing Surveys*, **2022,** *54*(10s), 1-35.
85. Deerwester, S.; Dumais, S.T.; Furnas, G.W.; Landauer, T.K.; Harshman, R.; Indexing by latent semantic analysis. *Journal of the American society for information science*, **1990,** *41*(6), 391-407.
86. Hofmann, T.; Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, **1999,** 50-57.
87. Blei, D.M.; Ng, A.Y.; Jordan, M.I.; Latent Dirichlet allocation. *Journal of machine Learning research*, **2003,** 3: 993-1022.
88. Blei, D.M.; Kucukelbir, A.; McAuliffe, J.D.; Variational inference: A review for statisticians. *Journal of the American statistical Association*, *112*(518), **2017,** 859-877.
89. Steyvers, M.; Griffiths, T.; Probabilistic topic models. In *Handbook of latent semantic analysis*, **2007,** 439-460, Psychology Press.
90. Cataldi, M.; Di Caro, L.; Schifanella, C.; Emerging topic detection on twitter based on temporal and social terms evaluation. *International workshop on multimedia data mining*, **2010,** 1-10.
91. Kingma, D.P.; Welling, M.; An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, **2019,** *12*(4), 307-392.
92. Yan, X.; Guo, J.; Lan, Y.; Cheng, X.; A biterm topic model for short texts. *International conference on World Wide Web*, **2013,** 1445-1456.
93. Nguyen, T.; Luu, A.T.; Contrastive learning for neural topic model. *Advances in neural information processing systems*, **2021,** *34*, 11974-11986.
94. Li, C.; Wang, H.; Zhang, Z.; Sun, A.; Ma, Z.; Topic modeling for short texts with auxiliary word embeddings. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, **2016,** 165-174.
95. Wu, X.; Li, C.; Short text topic modeling with flexible word patterns. *International Joint Conference on Neural Networks*, **2019,** 1-7.
96. Shi, T.; Kang, K.; Choo, J.; Reddy, C.K.; Short-text topic modeling via non-negative matrix factorization enriched with local word-context correlations. *World wide web conference*, **2018,** 1105-1114.
97. Wang, R.; Zhou, D.; He, Y.; ATM: Adversarial-neural topic model. *Information Processing & Management*, **2019,** *56*(6), 102098.
98. Li, X.L.; Liang, P.; Prefix-tuning: Optimizing continuous prompts for generation. **2021,** *arXiv:2101.00190*.
99. Hu, Z.; Wang, L.; Lan, Y.; Xu, W.; Lim, E.P.; Bing, L.; Xu, X.; Poria, S.; Lee, R.; Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. *Conference on empirical methods in natural language processing*, **2023,** 5254-5276.
100. Rastogi, A.; Zang, X.; Sunkara, S.; Gupta, R.; Khaitan, P.; Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *AAAI conference on artificial intelligence;* **2020,** 34:05, 8689-8696.
101. Han, Z.; Gao, C.; Liu, J.; Zhang, J.; Zhang, S.Q.; Parameter-efficient fine-tuning for large models: A comprehensive survey. **2024,** *arXiv:2403.14608*.
102. Hu, E.J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; LoRA: Low-rank adaptation of large language models. *ICLR*, **2022,** *1*(2), 3.
103. Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, H.; Wang, H.; Retrieval-augmented generation for large language models: A survey. **2023,** *arXiv:2312.10997*, *2*(1).
104. Boleda, G.; Distributional semantics and linguistic theory. *Annual Review of Linguistics*, **2020,** *6*(1), 213-234.
105. Lenci, A.; Distributional models of word meaning. *Annual review of Linguistics*, **2018,** *4*(1), pp.151-171.
106. Sahlgren, M.; The distributional hypothesis. *Italian Journal of linguistics*, **2008,** *20*, 33-53.

 doi:10.20944/preprints202512.2126.v1

107. Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; & Zettlemoyer, L.; Deep contextualized word representations. *NAACL-HLT,* **2018,** 2227–2237.
108. Voorhees, E.; Harman, D.; 1998. Overview of the sixth text retrieval conference (TREC-6). *NIST Special Publication Sp*, **1998,** 1-24.
109. Nenkova, A.; Automatic text summarization of newswire: Lessons learned from the document understanding conference, **2005.**
110. Navigli, R.; Litkowski, K.C.; Hargraves, O.; Semeval-2007 task 07: Coarse-grained English all-words task. *International Workshop on Semantic Evaluations,* **2007,** 30-35).
111. Lin, Z.; Gou, Z.; Liang, T.; Luo, R.; Liu, H.; Yang, Y.; CriticBench: Benchmarking LLMs for critique-correct reasoning. **2024,** *arXiv:2402.14809*.
112. Zhou, Z.; Liu, S.; Ning, M.; Liu, W.; Wang, J.; Wong, D.F.; Huang, X.; Wang, Q.; Huang, K.; Is your model really a good math reasoner? Evaluating mathematical reasoning with checklist. **2024,** *arXiv:2407.08733*.
113. Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; Training verifiers to solve math word problems. **2021,** *arXiv:2110.14168*.
114. Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; Cobbe, K.; Let's verify step by step. *International Conference on Learning Representations,* **2023.**
115. Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; Steinhardt, J.; Measuring mathematical problem solving with the math dataset. **2021,** *arXiv:2103.03874*.
116. Rajpurkar, P.; Jia, R.; Liang, P.; Know what you don't know: Unanswerable questions for SQuAD. **2018,** *arXiv:1806.03822*.
117. Dua, D.; Wang, Y.; Dasigi, P.; Stanovsky, G.; Singh, S.; Gardner, M.; DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. **2019,** *arXiv:1903.00161*.
118. Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; Choi, Y.; Hellaswag: Can a machine really finish your sentence? **2019,** *arXiv:1905.07830*.
119. Longpre, S.; Hou, L.; Vu, T.; Webson, A.; Chung, H.W.; Tay, Y.; Zhou, D.; Le, Q.V.; Zoph, B.; Wei, J.; Roberts, A.; The Flan collection: Designing data and methods for effective instruction tuning. *International Conference on Machine Learning.* **2023,** 22631-22648.
120. Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; Ives, Z.; Dbpedia: A nucleus for a web of open data. I*nternational semantic web conference,* **2007,** 722-735: Springer Berlin Heidelberg.
121. Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; Taylor, J.; Freebase: a collaboratively created graph database for structuring human knowledge. *ACM SIGMOD international conference on Management of data,* **2008,** 1247-1250.
122. Hoffart, J.; Suchanek, F.M.; Berberich, K.; Lewis-Kelham, E.; De Melo, G.; Weikum, G.; YAGO2: exploring and querying world knowledge in time, space, context, and many languages. *International conference companion on World wide web,* **2011,** 229-232.
123. Baker, C.F.; Fillmore, C.J.; Lowe, J.B.; The Berkeley FrameNet project. *International Conference on Computational Linguistics*. **1998.**
124. Palmer, M.; Gildea, D.; Kingsbury, P.; The proposition bank: An annotated corpus of semantic roles. *Computational linguistics,* **2005,** *31*(1), 71-106.
125. Crowston, K.; Amazon Mechanical Turk: A research tool for organizations and information systems scholars. *Shaping the Future of ICT Research. Methods and Approaches: IFIP WG 8.2, Working Conference,* **2012,** 210-221: Berlin, Heidelberg: Springer Berlin Heidelberg.
126. Aronson, A.R.; Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. *AMIA Symposium* **2001,** 17.
127. Fiszman, M.; Rindflesch, T.C.; Kilicoglu, H.; Integrating a hypernymic proposition interpreter into a semantic processor for biomedical texts. In *AMIA Annual Symposium Proceedings,* **2003,** 239.

 doi:10.20944/preprints202512.2126.v1

128. Fensel, D.; Şimşek, U.; Angele, K.; Huaman, E.; Kärle, E.; Panasiuk, O.; Toma, I.; Umbrich, J.; Wahler, A.; Introduction: what is a knowledge graph?. *Knowledge graphs: Methodology, tools and selected use cases,* **2020,** 1-10: Cham: Springer International Publishing.
129. Huang, J.T.; Sharma, A.; Sun, S.; Xia, L.; Zhang, D.; Pronin, P.; Padmanabhan, J.; Ottaviano, G.; Yang, L.; 2020. Embedding-based retrieval in facebook search. *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining,* **2020,** 2553-2561.
130. Lindberg, D.A.; Humphreys, B.L.; McCray, A.T.; The unified medical language system. *Yearbook of medical informatics*, **1993,** *2*(01), pp.41-51.
131. Mintz, M.; Bills, S.; Snow, R.; Jurafsky, D.; Distant supervision for relation extraction without labeled data. *Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP,* **2009,** 1003-1011.
132. Knight, K.; Marcu, D.; Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, **2002,** *139*(1), 91-107.
133. Clarke, J.; Lapata, M.; 2006. Models for sentence compression: A comparison across domains, training requirements and evaluation measures. *International Conference on Computational Linguistics,* **2006,** 377-384.
134. Dettmers, T.; Pagnoni, A.; Holtzman, A.; Zettlemoyer, L.; Qlora: Efficient fine-tuning of quantized LLMs. *Advances in neural information processing systems*, **2023,** *36*, 10088-10115.
135. Floridi, L.; Chiriatti, M.; GPT-3: Its nature, scope, limits, and consequences. *Minds and machines*, **2020,** *30*(4), 681-694.
136. Feng, Z.; Guo, D.; Tang, D.; Duan, N.; Feng, X.; Gong, M.; Shou, L.; Qin, B.; Liu, T.; Jiang, D.; Zhou, M.; CodeBERT: A pre-trained model for programming and natural languages. **2020,** *arXiv:2002.08155*.
137. Chen, M.; et al, Evaluating large language models trained on code. **2021,** *arXiv:2107.03374*.
138. Nijkamp, E.; Pang, B.; Hayashi, H.; Tu, L.; Wang, H.; Zhou, Y.; Savarese, S.; Xiong, C.; CodeGen: An open large language model for code with multi-turn program synthesis. **2022,** *arXiv:2203.13474*.
139. Xu, F.F.; Alon, U.; Neubig, G.; Hellendoorn, V.J.; A systematic evaluation of large language models of code. *ACM SIGPLAN international symposium on machine programming,* **2022,** 1-10.