Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 9
AIDML 2025

Application of Pre-training Model in Natural Language Processing

Zhaosicheng Chu *

Department of Computer Science, South-Central Minzu University, Wuhan, China

* Corresponding Author Email: czjude7@outlook.com

Abstract. Natural language processing is a very important research area in the field of artificial intelligence, with the goal of enabling computers to understand human language, this research area brings together information from multiple disciplines such as linguistics, computer science, machine learning, mathematics, and cognitive psychology, it has two sides: cognition and understanding of natural language and natural language processing. Of these, natural language recognition and understanding allow computers to represent meaningful and quantifiable symbols and relationships in input language and compute accordingly depending on tasks. Natural language processing activities include the construction of models that explain language abilities and language applications, the creation of computational mechanisms for implementing and improving language models, the design of usage systems from language models, and the exploration of evaluation methods for such systems. This article elaborates on the conceptual framework of pre training models and discusses three mainstream pre training language architectures. Through comparative experiments with multiple indicators, this article verifies the excellent performance of pre trained models in natural language processing, which has significant advantages compared to traditional methods. Finally, the current status and future prospects are summarized.

Keywords: pre-trained model; Natural Language Processing; BERT; Large Language Model.

1. Introduction

The pretraining model represents a class of AI architectures that perform unsupervised learning on a wide range of uncommented datasets [1, 2]. Thus, robust feature representation and parameter distribution can be obtained, and special tasks can be completed through excellent generalization ability. The paradigm shift embodied in the pretraining language model is fundamentally to avoid the traditional practice of random initialization but to implement the system's pretraining protocol in the architecture design phase [3]. By using a huge corpus containing annotations and raw text data, these models absorb the common language patterns embedded in the context framework [4]. This acquired knowledge condenses in the high-dimensional parameter space, and then undergoes the fine-tuning process of specific tasks [5]. The improvement of this methodology enables the refined language intelligence to be transferred in various downstream applications [6]. Thus, the training efficiency can be improved through adaptive knowledge extraction, and at the same time, it can be more closely aligned with the target of a specific task.

The training model refers to a model that conducts unsupervised learning on large-scale unlabeled data to learn effective feature representation and parameter distribution, can meet specific tasks, and has strong generalization ability.

The key to the pretraining language model is to give up the randomization of the original model, use a huge corpus to pretrain the model at the beginning of design, capture the general language information containing context information from a large number of labeled and unlabeled data, and then put the obtained information into a large number of parameters and fine-tune it according to specific tasks, so that this part of information can benefit various downstream tasks, not only improve training efficiency but also better meet the goal of the task. The paper elaborates on the conceptual framework of pre training models and discusses three mainstream pre training language architectures. Through comparative experiments with multiple indicators, this article verifies the excellent

performance of pre trained models in natural language processing, which has significant advantages compared to traditional methods. Finally, the current status and future prospects are summarized.

2. Introduction to Commonly Used Pre-trained Language Models

This section provides a detailed description of three classic pre training language architectures in chronological order from basics to contemporary innovation.

2.1. Word2vec

Two papers were published in 2013 by Tomas Mikolov and a few of his co-workers at Google: *Distributed Representations of Words and Phrases and their Compositionality* [7] and *Efficient Estimation of Word Representations in Vector Space* [8]. In these studies, they suggest Word2Vec, which is an extremely efficient method to learn distributed representation of words. With this technique, words can be converted to low-dimensional dense vectors which are capable of capturing the semantic and syntactic relationships between the words. It successfully avoids the shortcomings of traditional sparse hot encoding and introduces a tremendous advance in natural language processing. This method transforms words into dense and meaningful higher-dimensional mathematical representations, as opposed to the previous sparse and difficult-to-use encoding methods. This leads to a much better capture and understanding of the relationships and contexts between words in this new method. That means it can reason the association and context of words to improve language comprehension [7].

Word2Vec utilizes a corpus in unsupervised learning, and it has 3-layer neural network architecture (input layer, hidden layer, output layer) which is quite simple. It has two main architectures: Continuous Bag of Words (CBOW) and Skip Gram. CBOW predicts the target, given the context, and Skip Gram does the reverse task of predicting the context given the target word The system inputted one-hot-represented word strings into a hidden-layer of flexible dimensionality, adjusting along the way until it was capable of grouping semantically similar words into the same regions in the resulting embedding space. This positioning of elements in a 'word space' permits us to quantify linguistic relationships with vector mathematics, beautifully mitigating any chasms of meaning that may exist in the discrete, symbolic forms with which we typically represent our thoughts.

In terms of the model structure. Word2Vec It's a very light-weight three layer neural network (input layer, hidden layer, output layer) The model inputs the word and then convert it into One-Hot encodes then feeds it to the hidden layer. In the hidden layer, there are two training architectures: CBOW and Skip-Gram; with CBOW, you can predict a target word given the context, and, with Skip-gram, you can estimate the context of the target word in the reverse direction.

2.1.1. Skip-Gram

The Skip-Gram architecture, also known as the skip-gram model, works by predicting the surrounding words within a specific window around a central word [8]. In practice, this approach takes a one-hot encoded vector as input and produces probability distributions across the vocabulary to measure how likely each word is to appear in the context.

Consider the exemplar phrase "I like driving": when the central term "like" is encoded as a sparse binary vector, the model computationally optimizes the conditional probability $P(\{"I", "driving"\} \mid "like")$, thereby maximizing the joint likelihood of contextual associations.

It converts words, in contrast to being discrete, into movements of mathematics. These representations will form word vectors after passing through the hidden layer that are able to capture the semantic similarity of different words.

Fig.1 illustrates the transformation process from input to output.

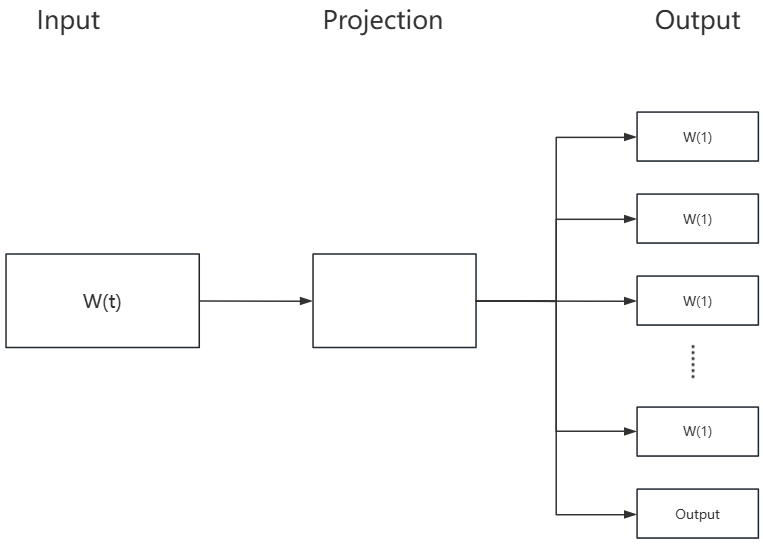

Fig. 1 The process of Skip-Gram from input to output

2.1.2. CBOW

The Continuous Bag of Words model (CBOW model for short) differs from the Skip-Gram model in that it takes surrounding words into account when generating a central word [8]. This means that a word in the center is generated based on the surrounding words, similar to filling in a blank with selected words. CBOW generates a central word in a sentence based on the relationship between contextual words and calculates the conditional probability for each word in the sentence. This process adds context to the language model, thereby enhancing its performance.

The conditional probability of generating the central word wc can be expressed by the following formula:

$$\mathrm{P}\left(w_c \mid w_{o_1}, \ldots, w_{o_{2m}} \right) = \frac{\exp\left(\frac{1}{2m} u_c{}^{\top} \left(v_{o_1} + \cdots + v_{o_{2m}}\right)\right)}{\sum_{i \in V} \exp\left(\frac{1}{2m} u_i{}^{\top} \left(v_{o_1} + \cdots + v_{o_{2m}}\right)\right)} \tag{1}$$

This formula calculates the dot-product similarity between the central word vector uc and the averaged contextual vectors Voi within a symmetric window of half-width mm (spanning $2m$ terms), amplifies this measure via the exponential function, normalizes it through summation over all candidate vectors in the vocabulary V, and ultimately outputs the conditional probability of generating target word wc under given contextual constraints, thereby encoding semantic associations and distributional patterns within a continuous vector space. The parameter mm specifies the contextual window's half-width (e.g., $m = 2$ captures two terms preceding and two following the central word), with $\frac{1}{2m}$ representing the averaging operation over summed contextual embeddings.

2.1.3. The limitations of Word2vec

While Word2Vec radically altered natural language processing by educating machines about context through neural networks and inaugurated the idea of word embeddings, it possesses perceptible deficiencies. For example, the model encounters complications distinguishing between words with many interpretations and often fails when coming across unfamiliar expressions [9]. In addition, its understanding of semantics remains relatively elementary compared to human-level nuance. Despite gatecrashing the arena and propelling the technological evolution, certain key factors persist outside the model's current reach.

2.2. BERT

Developed by Google Research in 2018, BERT represents a Transformer-based pretrained language model [1,10] that assimilates comprehensive linguistic knowledge—lexical semantics, syntactic

structures, and contextual patterns—through exposure to vast unsupervised pretraining corpora. During fine-tuning, minimal annotated data suffices to adapt its parameters for downstream tasks, achieving rapid domain specialization. This architecture marked a paradigm shift in natural language processing, dominating nearly all NLP domains until the advent of larger models. BERT introduced the standard "pretrain-then-finetune" approach for language models [1], paving the way for the era of pretrained language models (PLMs). It achieved this through a two-step training process that separates the learning of general language patterns from the fine-tuning needed for specific tasks.

Based on the Transformer architecture [10], the significant difference between BERT and its predecessor is that it only uses stacked Transformer encoder layers instead of the original encoder-decoder framework [1]. This design choice achieves deep bidirectional modeling of input text, overcoming the limitations of traditional one-way language models by capturing contextual information from various directions. Each layer of the encoder consists of two key components: a multi-head self-attention mechanism and a feedforward neural network. Among them, the self-attention mechanism is particularly crucial because it enables the model to capture the relationships between all words in the input sequence, rather than being limited to attention focused on local context. Through this approach, BERT can achieve comprehensive semantic deconstruction by synergistically fusing linguistic clues of positional distribution.

Table 1 illustrates the architectural differences between BERT's Transformer and traditional frameworks.

Table 1. differences between BERT's Transformer and traditional frameworks.

	Original Transformer	BERT
Structure	Encoder-Decoder (for generation tasks)	Encoder-only (for understanding tasks)
Attention Mechanism	Cross Attention	Pure Self-Attention (no Cross Attention)
Training Objective	Sequence-to-Sequence Translation Task	MLM + NSP (unsupervised pre-training)
Input Processing	Positional Encoding via Sine Function	Learnable Positional Embeddings
Context Modeling	Bidirectional Encoder, Unidirectional Decoder	Fully Bidirectional (no directional constraints in self-attention)

2.3. GPT

The rapid advancement of computing hardware and the deep integration of GPU technology in machine learning have driven the scale of model parameters to grow from billions to tens of billions, even reaching trillions of dimensions. GPT is a series of pretrained language models based on Transformer created by OpenAI, and its origin can be traced back to the GPT-1 architecture based on the Transformer framework. This architecture is trained on an unprecedented scale through GPU-accelerated unsupervised pretraining [2]. Subsequent iterations—GPT-2 with expanded datasets and parameter counts, GPT-3 which revolutionized both natural language processing and broader computational paradigms, and the now-pervasive GPT-4—each marked quantum leaps in capability through exponential scaling. This development has driven NLP to shift from task-specific designs to versatile, general-purpose frameworks, making AI more accessible and breaking free from the traditional limits of language processing [1].

2.3.1. GPT-1

Compared with BERT, GPT-1 uses stacked Transformer decoder layers optimized for generation tasks, which is different from BERT with encoder as the core architecture [2]. The Transformer module of this model focuses on one-way autoregressive generation, gradually synthesizing text by predicting each token based solely on left contextual clues—a design paradigm that prioritizes sequential coherence over bidirectional semantic integration [1].

Table 2 describes the aspects of the Framework that distinguish the GPT from the BERT.

Table 2. differences between BERT's Transformer and GPT‘s frameworks.

Feature	GPT's Transformer	BERT's Transformer
Attention Mechanism	Masked Self-Attention (Unidirectional)	Bidirectional Self-Attention (No Masking)
Positional Encoding	Learnable Positional Embeddings	Learnable Positional Embeddings
Residual Connection Order	Self-Attention → LayerNorm → FFN → LayerNorm	Self-Attention → LayerNorm → FFN → LayerNorm
Core Task	Next Token Generation (Autoregressive)	Contextual Understanding (Bidirectional Modeling)
Input-Output Dependency	Output Depends on Previous Tokens (Sequential)	Input-Independent, Parallelizable

2.3.2. GPT-2

The generalization ability of the GPT-1 architecture is limited and significantly surpassed by BERT's bidirectional architecture, while still relying on the traditional paradigm of unsupervised pretraining on large-scale unlabeled corpora followed by task-specific fine-tuning [11]. This method imposes strict retraining requirements on new downstream tasks and assumes the availability of labeled data even for unseen datasets [3]. When developing GPT-2, OpenAI adopted an expansion-centered strategy of expanding the training corpus and parameter capacity but did not make any architectural innovations. This has improved generalization ability to some extent but has not addressed its fundamental limitations [2]. As a transitional model of the GPT series, the main contribution of GPT-2 is to verify the hypothesis that when the input corpus is large enough and the designed parameters are sufficient, the generalization ability of the language processing model will also be correspondingly improved. This improvement relies more on large-scale text resources instead of algorithm optimization [12].

2.3.3. GPT-3

On the basis of verifying that GPT-2 achieves language generalization through extreme extensions, OpenAI developed GPT-3, which has a parameter count of 175 billion, ten times the previous Turing NLG's 1.7 billion parameter count. Through architecture optimization and unprecedented parameter expansion, GPT-3 achieved this breakthrough [13]. Unlike the simple expansion growth of GPT-2, GPT-3 introduces a sparse attention mechanism in its Transformer framework, thereby reducing the computational complexity of the attention layer and achieving efficient processing of longer input sequences [14]. The massive growth in training data, from 40GB to 45TB, has fueled the rise of multi-task reasoning and cross-domain adaptability [15]. This massive scaling effort provided clear evidence of power-law relationships between model performance and the combination of compute, data, and parameters, firmly supporting the "scaling hypothesis" [16]. The architecture’s ability to integrate context and interact in a human-like way represented a significant shift—from being just language tools to becoming collaborative partners—changing how language models fit into computational systems [17].

Table 3 demonstrates the development of parameter sizes of different versions of GPT and compares the exponential pattern of growth of parameter size from GPT-1 to GPT-4 through quantitative analysis.

Table 3. Overview of GPT Model Versions and Parameters

Model Version	Release Date	Parameters
GPT-1	June 2018	117 million
GPT-2	February 2019	1.5 billion
GPT-3	May 2020	175 billion
GPT-3.5	2022	200 billion
GPT-4	March 2023	Trillion-level

3. Applications of Pre-Trained Models in NLP

This chapter explores the typical applications of two pre trained models in the field of natural language processing, comparing and analyzing their methodologies and the advantages of pre trained models in this field from multiple dimensions.

3.1. Domain adaptation for medical text classification via pre-trained language models

This study compared and analyzed three models: the universal BERT Base Chinese model, the domain-specific BERT model (Med BERT Chinese) retrained in Chinese medical literature, and the traditional SVM classifier. All models were trained using the same Chinese medical corpus and evaluated on a dataset of 3200 annotated literature abstracts [18]. Their performance in medical text classification was comprehensively evaluated through metrics such as precision, recall, and F1 score. Experiments have shown that domain adaptation pretraining can significantly improve BERT's understanding of medical terminology, while traditional SVM methods struggle to capture complex language hierarchy patterns. The results also revealed a significant correlation between the targeting of pretrained data and the generalization ability of the model, providing valuable references for optimizing the application of pretrained models in specific NLP tasks.

When the sample size is set to 16,000, the obtained data is demonstrated in Table 4.

The F1 score is the harmonic mean of precision and recall, providing a comprehensive indicator that balances the two, which evaluates the overall performance of the model by considering both its accuracy and comprehensiveness.

Table 4. F1 scores of the models in this experiment

Metric	SVM	BERT-Base-Chinese	BERT-Re-Pretraining-Med-Chi
Precision	0.7787	0.8477	0.8502
Recall	0.7523	0.8289	0.8445
F1 Score	0.7651	0.8248	0.8420

When the sample size is set to 32,000, the obtained data is demostrated in the Table 5.

Table 5. F1 scores of the models in this experiment

Metric	SVM	BERT-Base-Chinese	BERT-Re-Pretraining-Med-Chi
Precision	0.7787	0.8477	0.8502
Recall	0.7523	0.8289	0.8445
F1 Score	0.7651	0.8248	0.8420

Experimental findings demonstrate the superior efficacy of pretrained language models over conventional approaches in Traditional Chinese Medicine (TCM) literature classification tasks. The BERT-Re-Pretraining-Med-Chi model—domain-adapted through biomedical corpus re-pretraining—achieved peak performance with mean F1-scores of 0.8420 and 0.8767 under 3,200 and 16,000 sample conditions respectively, outperforming both the generic BERT-Base-Chinese and SVM baselines. This empirical validation underscores two critical insights:

(1) pretrained architectures inherently surpass traditional models in hierarchical feature extraction for text classification, and

(2) task-specific adaptation through domain-informed pretraining substantially enhances model specialization, as evidenced by the 4.1% F1-score differential between domain-tuned and general-purpose BERT variants. The observed performance gradients mirror the hierarchical nature of linguistic representations in transformer architectures, where targeted pretraining amplifies the model's capacity to disentangle domain-specific semantic patterns.

3.2. A Method for Automatic Reliable Code Generation Based on Knowledge Graphs and GPT Models

This study establishes a structured knowledge graph where entities and relationships are algorithmically converted into interpretable variables, which are co-embedded with functional code segments into a pre-trained GPT-2 architecture [19]. Through domain-specific fine-tuning, the augmented model demonstrates code generation capabilities by inferring contextual relationships from hybrid knowledge-code inputs. Post-training validation incorporates classical reliability analysis paradigms—Monte Carlo simulations (MCS), first-order reliability approximation methods (FORM), and subset simulation (SS)—with comparative benchmarking against MATLAB Deep Learning Toolbox, CodeGen, and MATGPT frameworks.

The assement covers a series of reliability engineering case studies by quantitative comparison of syntactic correctness, behavioural completeness and computational efficiency of the generated code.

The Score of the code generated by each model for the three case studies are presented in Table 6.

Table 6. Scores of the models in this experiment

	Score for MCS	**Score for DSTM**	**Score for SS**
Model Used in This Experiment	0.98	0.60	0.80
Matlab-deep-learning	0.95	0.40	0.62
Code-gen	0.85	0.35	0.58
MatGPT	0.95	0.50	0.68

The effectiveness of the pretrained model is validated through typical reliability analysis cases. Comparative evaluation with baseline models demonstrates that the knowledge graph and GPT-based pretrained model proposed in this chapter achieves superior code generation performance across multiple metrics, exhibiting robust contextual analysis and logical reasoning capabilities.

4. Conclusion

This paper elucidates the conceptual framework of pre trained models and explores three representative pre training language architectures. Through comparative experiments of multiple indicators, the paper empirically verifies the excellent performance of pre trained models in natural language processing, which has significant advantages compared to traditional methods, and demonstrates their emerging abilities as "assistants".

Pre trained models have become the mainstream paradigm in the field of natural language processing, replacing traditional methods with their excellent efficiency, accuracy, and powerful generalization ability. At present, high-performance architectures mainly rely on the encoder or decoder components in the Transformer framework. Encoder based models perform well in context understanding and can achieve precise semantic disambiguation; The architecture centered around the decoder exhibits unparalleled performance in text generation and inference capabilities.

As one of the typical examples of the decoder paradigm, the GPT series has surpassed the application fields of natural language processing models, and with its powerful reasoning and retrieval capabilities, has become an indispensable intelligent assistant in people's daily lives. In the future, with the continuous advancement of technology, it can promote the development of more scientific training methods and ultra large scale pre training structures that integrate complete encoder decoder frameworks. This model can utilize logical reasoning ability to achieve context based text generation while retaining deep language comprehension capabilities. This new model can redefine the application and potential limits of human-machine collaboration, thereby improving social efficiency.

References

[1] Devlin J, Chang M W, Lee K, & Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics, 2019, 4171-4186.

[2] Radford A, Narasimhan K, Salimans T, & Sutskever I. Improving language understanding by generative pre-training. OpenAI Technical Report. 2018.

[3] Howard J, & Ruder S. Universal language model fine-tuning for text classification. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, 2018, 328-339.

[4] Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, ... & Liu P J. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 2020, 21(140), 1-67.

[5] Sanh V, Debut L, Chaumond J, & Wolf T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108. 2019.

[6] Hinton G, Vinyals O, & Dean J. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531. 2015.

[7] Mikolov T, Sutskever I, Chen K, Corrado G S, & Dean J. Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems, 2013a, 26, 3111-3119.

[8] Mikolov T, Chen K, Corrado G, & Dean J. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. 2013b.

[9] Pennington J, Socher R, & Manning C D. GloVe: Global vectors for word representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 1532-1543. 2014.

[10] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, ... & Polosukhin I. Attention is all you need. Advances in Neural Information Processing Systems, 2017, 30, 5998-6008.

[11] Peters M E, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, & Zettlemoyer L. Deep contextualized word representations. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics, 2018, 2227-2237.

[12] Kaplan J, McCandlish S, Henighan T, Brown T B, Chess B, Child R, ... & Amodei D. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361. 2020.

[13] Brown T B, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, ... & Amodei D. Language models are few-shot learners. Advances in Neural Information Processing Systems, 2020, 33, 1877-1901.

[14] Child R, Gray S, Radford A, & Sutskever I. Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509. 2019.

[15] Hernandez D, Kaplan J, Henighan T, & McCandlish S. Scaling laws for transfer. arXiv preprint arXiv:2102.01293. 2021.

[16] Alabdulmohsin I, Lucic M, & Cissé M. Revisiting the calibration of modern neural networks. Advances in Neural Information Processing Systems, 2023, 36.

[17] Bubeck S, Chandrasekaran V, Eldan R, Gehrke J, Horvitz E, Kamar E, ... & Zhang Y. Sparks of artificial general intelligence: Early experiments with GPT-4. arXiv preprint arXiv:2303.12712. 2023.

[18] Zhao S, Zhang Y, Liu X, & Ding N. Domain adaptation for medical text classification via pre-trained language models. Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 2022, 4567-4578.

[19] Li X, Wang H, Chen Z, & Zhang L. Knowledge graph-enhanced code generation with GPT models. Proceedings of the 2023 International Conference on Software Engineering, 2023, 1-12.