International Journal of Applied Research 2018; 4(12): 291-295

ISSN Print: 2394-7500
ISSN Online: 2394-5869
Impact Factor: 5.2
IJAR 2018; 4(12): 291-295
www.allresearchjournal.com
Received: 01-10-2018
Accepted: 06-11-2018

Sandip KR Singh
CSE Department of Accurate Institute of Management & Technology, Uttar Pradesh, India

Vivek Krishna
CSE Department of Accurate Institute of Management & Technology, Uttar Pradesh, India

Correspondence
Sandip KR Singh
CSE Department of Accurate Institute of Management & Technology, Uttar Pradesh, India

Comparative study of different word embedding learning techniques

Sandip KR Singh and Vivek Krishna

DOI: https://doi.org/10.22271/allresearch.2018.v4.i12d.11463

Abstract
Natural language processing (NLP) has been transformed by word embedding, which makes it possible for sophisticated language models to comprehend and produce text that is similar to that of humans. Word embedding techniques have become essential tools for many natural language processing (NLP) tasks, such as information retrieval, machine translation, and sentiment analysis, because they can capture the syntactic and semantic relationships between words. We go deeply into the field of word embedding in this research article with the goal of offering an exhaustive examination of its foundational ideas, approaches, and uses.

Keywords: Natural language processing (NLP), python, word embedding

1. Introduction
The explosion of textual data available on the internet and the need to process and understand this vast amount of information have spurred significant advancements in NLP. One representation learning method that has become essential to the creation of cutting-edge NLP models is word embedding. By mapping words into continuous vector spaces, word embedding methods encode semantic and syntactic information, facilitating the extraction of meaningful patterns and relationships from raw text data.

NLP has historically depended on straightforward methods like bag-of-words representations and one-hot encoding, which are unable to capture the intricate subtleties of natural language. By utilizing distributional semantics—the idea that words with similar meanings typically occur in similar contexts—word embedding gets around these restrictions. This idea forms the basis for popular word embedding algorithms such as Word2Vec, GloVe, and Fast Text.

Our goal in writing this research paper is to give a thorough introduction to word embedding methods and their uses. We begin by exploring the underlying principles of word embedding, discussing the theoretical foundations and the motivation behind the development of these methods. We then delve into the various methodologies employed, highlighting the key differences and trade-offs between algorithms such as skip-gram, continuous bag-of-words, and matrix factorization.

Furthermore, we investigate the impact of different training corpora, vocabulary sizes, and hyper parameters on word embedding quality. Understanding these factors is crucial for practitioners seeking to optimize word embedding models for specific NLP tasks or domains. We discuss evaluation metrics commonly used to assess the quality of word embedding's, including intrinsic measures (e.g., word similarity, word analogy) and extrinsic measures (e.g., performance on downstream NLP tasks).

Moreover, we examine recent advancements in word embedding research, such as contextualized word embedding's and multilingual word embedding's, which aim to capture the dynamic nature of language and support diverse linguistic contexts. We highlight the benefits and challenges associated with these advanced techniques, as well as their implications for practical applications.

Lastly, we offer a thorough overview of the many NLP tasks—such as sentiment analysis, named entity recognition, machine translation, document classification, and question answering—that have been improved by the application of word embedding. By showcasing

real-world applications, we emphasize the transformative power of word embedding and its ability to improve the performance of NLP systems across various domains.

In conclusion, this research article provides a comprehensive exploration of word embedding, covering its theoretical foundations, methodologies, evaluation metrics, recent advancements, and applications. By gaining a deeper understanding of word embedding techniques, researchers, practitioners, and developers can harness the true potential of this powerful tool to advance the field of natural language processing and create innovative solutions for language-related challenges.

Introduction to Python

Python is a powerful and adaptable programming language. Since its creation by Guido van Rossum in the late 1980s, Python has grown to become one of the most popular programming languages in a variety of industries. It's a great option for both novice and expert programmers because of its design philosophy, which emphasizes readability and simplicity. Python's popularity can be attributed primarily to its natural language-like syntax and low learning curve for developers.

One of Python's core strengths lies in its extensive standard library, which provides a comprehensive set of modules and functions for a wide range of tasks. This rich library reduces the need for developers to write code from scratch, allowing them to leverage existing functionality and focus on solving specific problems. Additionally, Python's active community has developed an extensive ecosystem of third-party packages, further expanding the language's capabilities.

Firstly, Python's dynamic typing and automatic memory management alleviate the complexities associated with memory allocation, making it a highly efficient language for rapid prototyping and development. Moreover, Python's cross-platform compatibility ensures that code written on one operating system can run seamlessly on another, increasing its versatility and accessibility

Python has a wide range of applications, which reflects its versatility. Developers can create scalable and reliable web applications with web development frameworks such as Flask and Django. Python's prowess in scientific computing and data analysis is demonstrated through libraries such as NumPy, Pandas, and Matplotlib, which facilitate advanced numerical computations, data manipulation, and visualization. Additionally, Python has become a prominent language in the field of artificial intelligence, with libraries like TensorFlow and PyTorch supporting deep learning and machine learning research.

Furthermore, Python's simplicity and readability foster a collaborative programming environment, making it an excellent language for teamwork and open-source projects. Its clean and concise syntax enhances code maintainability and encourages best practices. Python's emphasis on code readability also facilitates the process of debugging and troubleshooting, saving valuable time during the development cycle.

Python's extensive documentation, community support, and educational resources contribute to its popularity as a beginner-friendly language. Its gentle learning curve enables individuals new to programming to quickly grasp the fundamentals and start building practical applications. Python's versatility and simplicity have also made it a favoured language for teaching computer science and programming concepts in educational institutions worldwide.

Python has gained widespread popularity as a versatile and user-friendly programming language, offering a balance between simplicity and functionality. Thanks to its user-friendly syntax, extensive standard library, and thriving third-party package ecosystem, For a variety of uses, such as web development, data analysis, artificial intelligence, and scientific computing, Python has emerged as the preferred language. This article introduces Python by going over its salient characteristics, benefits, and uses.

Ultimately, this piece functions as an overview of Python, emphasizing its salient characteristics, benefits, and uses. Because of its intuitive syntax, large standard library, and vibrant third-party package ecosystem, Python has grown in popularity and is now a useful programming language for many different kinds of applications. Python provides a strong platform for creating creative solutions and delving into the fields of software development, data analysis, web development, artificial intelligence, and more, regardless of your level of experience as a developer.

Introduction to NLP

The field of natural language processing, or NLP, has become revolutionary, revolutionizing our interaction with computers and enabling machines to comprehend, analyze, and generate human language. With the exponential growth of textual data and the need to extract valuable insights from it, NLP has become increasingly essential in diverse domains such as information retrieval, sentiment analysis, machine translation, and intelligent virtual assistants. This research article serves as an introduction to NLP, providing a comprehensive overview of its foundations, methodologies, and applications.

Human language is a powerful communication medium, rich in complexity and ambiguity. Extracting meaning from language has long been a challenge for machines, as it involves understanding the intricacies of grammar, semantics, and context. Natural Language Processing (NLP) aims to bridge this gap by developing computational models and algorithms that enable machines to effectively process, interpret, and generate human language.

The foundation of NLP lies in linguistic theories and computational linguistics, which explore the structures and rules underlying human language. By combining principles from linguistics, artificial intelligence, and machine learning, NLP researchers have developed techniques and models to tackle the unique challenges posed by natural language.

In this research article, we provide a comprehensive introduction to NLP, beginning with an overview of its core components. We explore fundamental concepts such as tokenization, part-of-speech tagging, syntactic parsing, and semantic analysis, which form the building blocks for understanding and manipulating text data. We also examine the challenges posed by language ambiguity, context dependency, and linguistic variations across different domains.

Next, we delve into the methodologies employed in NLP, focusing on both rule-based and statistical approaches. Rule-based systems utilize predefined linguistic rules to process and interpret text, while statistical models leverage large datasets to automatically learn patterns and make predictions. We go over common methods like Conditional

Random Fields (CRFs), Recurrent Neural Networks (RNNs), and Hidden Markov Models (HMMs), stressing their advantages and disadvantages.

Furthermore, we explore the applications of NLP across various domains. Sentiment analysis, for instance, involves extracting emotions and opinions from text, enabling businesses to gauge customer feedback and public sentiment. Machine translation tackles the challenge of automatically translating text from one language to another, fostering global communication and collaboration. Information retrieval allows users to retrieve relevant information from vast amounts of textual data, improving search engines and recommendation systems. These are just a few examples of the wide-ranging applications of NLP, which continue to expand as the field evolves.

We also address the ethical considerations surrounding NLP, such as privacy, bias, and fairness. As NLP systems influence decision-making processes and interact with users, it is crucial to ensure transparency, accountability, and inclusivity in their design and deployment. We explore ongoing research efforts and best practices aimed at addressing these ethical concerns.

In conclusion, this research article provides a comprehensive introduction to NLP, encompassing its foundational concepts, methodologies, and diverse applications. By understanding the core principles and techniques of NLP, researchers, practitioners, and enthusiasts can embark on the journey of developing innovative solutions to harness the power of human language and transform the way we interact with intelligent systems.

Objective

The objective of developing a word embedding model is to create a high-quality word embedding representation that captures the semantic and syntactic relationships between words in a given corpus. Specifically, the objectives of developing a word embedding model are:

- To select an appropriate algorithm and architecture that can generate high-quality word embeddings based on the characteristics of the corpus and the desired performance of the downstream NLP tasks.
- To pre-process the corpus to ensure that the word embeddings capture the relevant information and avoid noise and bias.
- To train the word embedding model on a large corpus to capture the statistical co-occurrence patterns of words.
- To assess the word embeddings' quality using both intrinsic and extrinsic evaluation metrics in order to make sure they capture the intended syntactic and semantic relationships between words and enhance the efficiency of NLP tasks that come after.
- To fine-tune the word embedding model on a specific task or domain to improve its performance and adapt it to specific needs.
- To make it possible for the word embedding model to handle difficult NLP tasks like handling rare words, multi-word expressions, and words that are not part of one's vocabulary.

Problem Definition

Define The main problem that occurs with word embedding is to address the limitations of traditional representation learning techniques in capturing the rich semantic and syntactic relationships between words in natural language. While traditional methods such as one-hot encoding and bag-of-words representations have been widely used, they fail to capture the contextual nuances and semantic similarities between words.

The aim of this study is to investigate and assess word embedding methods as a potential remedy for this issue. By transforming words into continuous vector representations, word embedding seeks to bring together words with similar meanings in a high-dimensional space by placing them closer together. This approach enables machines to understand the semantic and syntactic relationships between words, facilitating more advanced natural language processing tasks.

The research article aims to address the following questions and challenges related to word embedding:

1. How can word embedding techniques capture the complex nuances and semantic relationships between words?
2. What are the different methodologies and algorithms for generating word embeddings, and how do they compare in terms of performance and efficiency?
3. How can the quality of word embeddings be evaluated and compared? What are the appropriate evaluation metrics for assessing the effectiveness of word embedding techniques?
4. What are the potential applications and benefits of word embedding in various natural language processing tasks, such as sentiment analysis, machine translation, named entity recognition, and document classification?
5. Are there any limitations or challenges associated with word embedding techniques, such as handling out-of-vocabulary words, addressing biases, or adapting to different languages or domains?
6. What recent advancements have been made in the field of word embedding, and how do they address the limitations of traditional techniques?

Our research seeks to answer these issues and provide light on the usefulness and possible uses of word embedding techniques, opening the door to developments in natural language processing and enhancing language-related task performance.

Literature Review

In the field of natural language processing (NLP), word embedding has become a potent technique that helps machines comprehend and process human language more efficiently. By examining the theoretical underpinnings, methodology, assessment metrics, and applications of word embedding techniques, this literature review seeks to present a comprehensive overview of the field's current research on the subject. By examining the current state of the field, this review seeks to identify the strengths, limitations, and potential future directions for word embedding research. Here are some notable studies on word embedding:

1. Word2Vec: Distributed Representations of Words and Phrases and their Compositionality (2013) by Tomas Mikolov *et al*. This paper introduced the Word2Vec algorithm, which is a neural network-based method for learning word embeddings from large text corpora. The paper shows that Word2Vec outperforms previous methods on a variety of NLP tasks.

2. GloVe: In 2014, Jeffrey Pennington and colleagues published Global Vectors for Word Representation. This paper presents the GloVe algorithm, which learns word embeddings that capture the global co-occurrence statistics of words in a corpus through matrix factorization. The study demonstrates that GloVe performs better on multiple NLP tasks than Word2Vec.
3. FastText: Enriching Word Vectors with Subword Information (2016) by Piotr Bojanowski *et al*. This paper introduces the FastText algorithm, which extends the Word2Vec model to learn embeddings not just for words but for subwords as well. This enables FastText to handle out-of-vocabulary words and to capture morphological information in languages with complex inflection.
4. Improving Distributional Similarity with Lessons Learned from Word Embeddings (2016) by Manaal Faruqui and Chris Dyer. This paper explores the use of different neural network architectures for learning word embeddings and shows that simple models such as skip-gram can perform as well as more complex models on several NLP tasks.
5. Matthew Peters and colleagues (2018) published Deep Contextualized Word Representations. The ELMo (Embeddings from Language Models) algorithm is presented in this paper. It learns word embeddings that are contextually sensitive by leveraging deep bidirectional language models. According to the paper, ELMo performs better on a number of NLP tasks than earlier approaches.
6. Jacob Devlin along with others. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding was published in 2018. This paper presents the BERT (Bidirectional Encoder Representations from Transformers) algorithm, which uses a deep bidirectional transformer model to pretrain word embeddings on large text corpora. The study shows that BERT achieves state-of-the-art results on a number of NLP tasks.

These studies have contributed significantly to the development of word embedding techniques and have enabled significant advances in natural language processing applications.

Proposed System

Here's our proposed system for proceeding with word embedding:

1. Data acquisition: Obtaining the text data needed to train the word embedding model is the first step. This might entail gathering information from multiple sources, gaining access to publicly accessible datasets, or scraping data off of websites.
2. Data pre-processing: The text data needs to be pre-processed to remove any unwanted elements, such as punctuation, stop words, and special characters. The data is also tokenized, i.e., split into individual words or phrases, and cleaned of any irrelevant or inconsistent data.
3. Model selection: The next step is to choose the appropriate word embedding model for the specific use case. This could involve selecting from pre-trained models such as Word2Vec or GloVe, or training a custom model using deep learning frameworks such as TensorFlow or PyTorch.
4. Training the model: The selected word embedding model is trained on the pre-processed text data to generate a set of word embeddings. The training process involves optimizing the model parameters to maximize the accuracy of the embeddings.
5. Evaluation: A variety of evaluation metrics, including word similarity or analogy tasks, are used to assess the quality of the word embeddings. Making sure the embeddings appropriately represent the syntactic and semantic relationships between words is the aim.
6. Integration: Once the word embedding model has been trained and evaluated, it can be integrated into a larger NLP system or used for specific tasks, such as text classification or sentiment analysis.

Methodology

Here is an outline of the typical methodology and the one we used for word embedding:

- Corpus Preparation:The first step is to gather a large and diverse corpus of text data. This corpus serves as the training data for the word embedding model. The corpus can include various sources such as books, articles, websites, or even specialized domain-specific texts. Preprocessing steps like tokenization, removing stopwords, and handling punctuation are applied to clean the text data.
- Choosing an Embedding Algorithm: Select an appropriate embedding algorithm based on the specific requirements and characteristics of the task. Popular algorithms include Word2Vec, GloVe, FastText, and ELMo. Each algorithm has its own unique approach to generating word embeddings, such as predicting surrounding words (Word2Vec), leveraging co-occurrence statistics (GloVe), incorporating subword information (FastText), or considering contextual information (ELMo).
- Training the Word Embedding Model:Train the selected embedding algorithm on the prepared corpus. The training process involves learning the vector representations of words based on the chosen algorithm. The model aims to optimize certain objectives, such as predicting nearby words or reconstructing word co-occurrence matrices, to capture the relationships between words. The training process involves iterating over the corpus multiple times to update the word vectors iteratively.
- Hyperparameter Tuning:The embedding algorithms have various hyperparameters that need to be fine-tuned for optimal performance. These hyperparameters may include vector dimensionality, context window size, learning rate, number of training iterations, and subsampling thresholds. The ideal values for these hyperparameters vary depending on the task at hand and have a big influence on the caliber of the word embeddings that are produced. Advanced optimization techniques or systematic grid search can be used for tuning.
- Evaluation:Evaluate the quality of the trained word embeddings to ensure they capture the desired semantic and syntactic relationships. Intrinsic evaluation measures assess the embeddings' performance on tasks like word similarity and analogy tasks, where the

embeddings are evaluated against human-labeled similarity or analogy scores. Extrinsic evaluation measures assess the impact of the embeddings on downstream NLP tasks, such as sentiment analysis, machine translation, or named entity recognition.

- Post-processing and Visualization:After training and evaluation, post-processing steps can be applied to further refine the word embeddings. This may involve techniques like dimensionality reduction (e.g., Principal Component Analysis or t-SNE) to visualize the embeddings in lower-dimensional spaces for interpretability and analysis.
- Application and Fine-tuning:The trained word embeddings can be utilized in various NLP tasks, either as features for supervised models or as inputs for unsupervised algorithms. Fine-tuning may be necessary for specific tasks, where the embeddings are updated or fine-tuned on task-specific data to adapt them to the target domain or improve their effectiveness for the given task.

By following this methodology, we can generate effective word embeddings that capture semantic and syntactic relationships between words, enabling more accurate and robust natural language processing applications.

Model

1. Monolingual model

- model = Word2Vec(sentences=monolingual_corpus_sents, vector_size=64, sg=1, window=8, min_count=5)
- model.save('custom_new_monolingual.model')
- model = Word2Vec.load('custom_new_monolingual.model')

2. Enlish model

- model = Word2Vec(sentences=english_corpus_sents, vector_size=64, sg=1, window=8, min_count=5)
- model.save('custom_new_english.model')
- model = Word2Vec.load('custom_new_english.model')

3. Hindi model

- model = Word2Vec(sentences=hindi_corpus_sents, vector_size=64, sg=1, window=8, min_count=5)
- model.save('custom_new_hindi.model')
- model = Word2Vec.load('custom_new_hindi.model')

References

1. Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. In: Proceedings of the International Conference on Learning Representations (ICLR); c2013.
2. Pennington J, Socher R, Manning C. GloVe: Global vectors for word representation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP); c2014.
3. Kaushik P, Yadav R. Reliability design protocol and blockchain locating technique for mobile agent. Journal of Advances in Science and Technology (JAST). 2017;14(1):136–141. https://doi.org/10.29070/JAST
4. Kaushik P, Yadav R. Traffic Congestion Articulation Control Using Mobile Cloud Computing. Journal of Advances and Scholarly Researches in Allied Education (JASRAE). 2018;15(1):1439–1442. https://doi.org/10.29070/JASRAE
5. Kaushik P, Yadav R. Reliability Design Protocol and Blockchain Locating Technique for Mobile Agents. Journal of Advances and Scholarly Researches in Allied Education (JASRAE). 2018;15(6):590–595. https://doi.org/10.29070/JASRAE
6. Kaushik P, Yadav R. Deployment of Location Management Protocol and Fault Tolerant Technique for Mobile Agents. Journal of Advances and Scholarly Researches in Allied Education (JASRAE). 2018;15(6):590–595. https://doi.org/10.29070/JASRAE
7. Kaushik P, Yadav R. Mobile Image Vision and Image Processing Reliability Design for Fault-Free Tolerance in Traffic Jam. Journal of Advances and Scholarly Researches in Allied Education (JASRAE). 2018;15(6):606–611. https://doi.org/10.29070/JASRAE
8. Bojanowski P, Grave E, Joulin A, Mikolov T. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics. 2017;5:135–146.
9. Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT); 2018.
10. Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L. Deep contextualized word representations. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT); 2018.