Recursive Center Embedding: An Extension of MLCE for Semantic Evaluation of Complex Sentences

ShivKishan Dubey[1], Narendra Kohli[2]
Department of Computer Science and Engineering, Harcourt Butler Technical University, Kanpur, India[1]
School of Engineering, Harcourt Butler Technical University, Kanpur, India[2]

***Abstract*—A novel method for representing hierarchical sentences, named Multi-Leveled Center Embedding (MLCE), has recently been introduced. The approach utilizes the concept of center-embedded structures to demonstrate the structural complexity of complex sentences through iterative calculations of differences between the original and modified embeddings of its hierarchy. Through an implementation of Recursive Center-Embedding (RCE), we enhance the concept of MLCE by incorporating additional leveled features from the center-word hierarchy. The features are essential for training the Word2Vec model, enabling it to generate sophisticated vectors that perform well in sentence similarity analysis. RCE produces vectors via a hierarchical arrangement of center components, illustrating sentence structure that exceeds that of traditional word vectors and the BERT-base contextual model. The aim is to assess the similarity performance of the proposed RCE strategy. Furthermore, it examines its contextual ability obtained through leveled feature vectors that successfully correlated pairs of complex sentences across multiple benchmark datasets.**

***Keywords*—Recursive Center Embedding (RCE); Multi-Level Center Embedding (MLCE); complex sentences; structural similarity**

I. Introduction

Sentence similarity challenges are pivotal in a wide range of natural language processing (NLP) applications, including retrieving similar context-based information and summarizing text. These challenges have paved the way for more advanced mechanisms [5], [6], [12]. One such mechanism is Word2Vec, a neural network-based model that has not only become a baseline for many sentence representation tasks but also one of the most widely adopted techniques for evaluating sentence similarity [7]. Word2Vec captures the semantic relationships between words by embedding them in a continuous vector space, effectively modeling word meanings based on their contextual co-occurrence in large corpora [11]. However, despite its success in word-level embeddings, Word2Vec faces a significant limitation: it requires more structural awareness, particularly when dealing with complex or compound sentence structures.

The limitation of Word2Vec arises from its treatment of sentences as mere bags of words, disregarding the sequential order and syntactic relationships that give sentences their meaning [6], [8]. This approach prevents it from distinguishing between sentences that consist of the same words but are structured differently. Take, for instance, the sentences *The dog chased the cat* and *The cat chased the dog*. Word2Vec would process these two sentences similarly, despite the fact that they convey entirely different meanings resultant from their word order. This lack of sensitivity to structure significantly impairs its effectiveness in assessing sentence similarity, especially when the intricacies of sentence construction are essential for accurate comprehension [1].

To overcome this challenge, leveraging the concept of leveled center embedding (MLCE) [1] offers a promising solution, specifically designed to enhance performance in tasks focused on structural similarity.

Recursive Center Embedding (RCE) has been proposed here as an innovative solution that integrates both word-level semantics and sentence structure into its representation. RCE is a recursive method that generates a hierarchical structure by embedding sentence components recursively around a central word. This recursive process allows RCE to capture not only the meaning of individual words but also the syntactic and structural relationships between them, producing a more comprehensive representation of the sentence. Importantly, RCE overcomes the major limitation of Word2Vec—its inability to account for word order and sentence structure, providing a convincing alternative.

A. Recursive Center Embedding (RCE): Structural Awareness in Sentences

Unlike Word2Vec, which assumes that word order is irrelevant, Recursive Center Embedding (RCE) inherently supports structural awareness, similar to Multi-Level Center Embedding (MLCE). It does this by recursively breaking down sentences into central components and their constituent parts.

The process begins by identifying the central word of the sentence and then recursively embedding the left and right contexts of that word. This approach is repeated until the entire sentence is represented hierarchically, capturing both the syntactic dependencies and the hierarchical relationships among the sentence components. The comparative structural behavior is illustrated in Fig. 1, which shows the differences between conventional word vectors and the hierarchical vectors derived through RCE. This method addresses the challenges of word order that have affected Word2Vec-based models. By taking into account the positional and syntactic relationships between words, RCE can differentiate between sentences with different structures, even if they contain the same words. This capability makes RCE particularly well-suited for tasks related to sentence similarity.

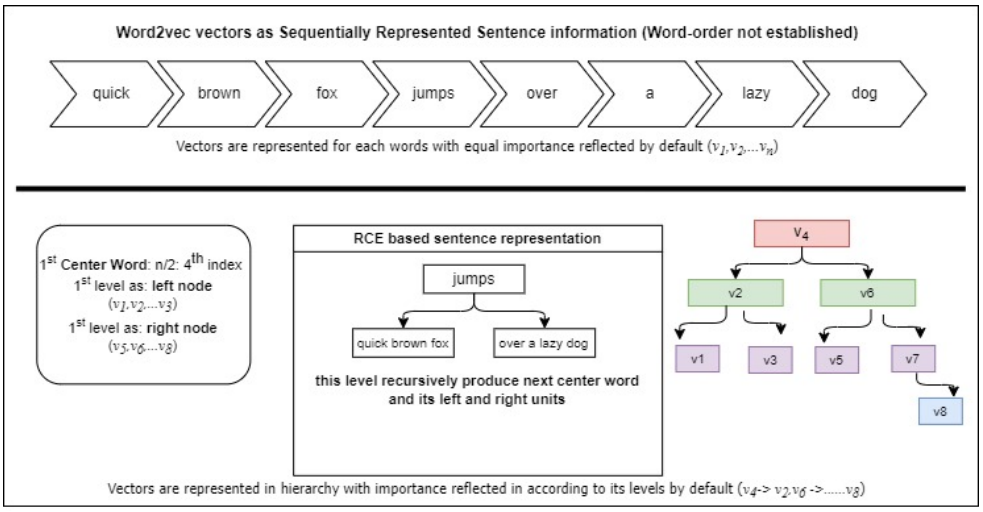


Fig. 1. Structural representation of sentence quick brown fox jumps over a lazy dog using traditional word vectors and RCE hierarchical-vectors.

B. Motivation and Need for Structural Embedding Model

Traditional models like Word2Vec, while effective for word-level tasks, fall short when applied to tasks requiring a deeper understanding of sentence composition. The structural limitations of Word2Vec have been documented in various studies, particularly in its inability to handle complex syntactic constructions or to differentiate between sentences that differ only in word order [1]. As the complexity of sentence structures increases, RCE offers a more nuanced and complete approach by integrating sentence structure directly into the embedding process. The recursive nature of RCE allows it to handle complex and nested structures, such as subordinate clauses or center-embedded sentences, that are challenging for traditional word embeddings.

C. Our Major Contributions

To show how RCE improves sentence representation by adding structural knowledge, we address Word2Vec-based embedding restrictions. We will also propose a mathematical formulation of RCE showing how to construct hierarchal vectors with their features and a theorem showing how RCE enhances by recursively accumulating contextual information. Incorporating additional contexts allows RCE to effectively assess complex sentences that extend beyond immediate context pairs.

This research compares RCE to the Word2Vec baseline and BERT-base model across benchmark datasets to determine semantic connection gains. Additionally, to evaluate its generalizability across sentence complexity levels using benchmark datasets.

D. Research Focus

The primary focus of this research is to assess the effectiveness of Recursive Center Embedding (RCE) in sentence similarity tasks across various benchmark datasets. While RCE has shown potential in addressing the structural limitations of Word2Vec, its ability to consistently outperform traditional methods in real-world datasets remains uncertain. This study specifically aims to investigate whether RCE can achieve superior performance in sentence similarity tasks by integrating both structural and semantic awareness into sentence representations. The previous version, MLCE, has the ability to handle ambiguity by considering center-embedding based contexts as grouped clusters [9].

This paper is structured as follows: Section II outlines relevant research on sentence similarity assessment, focusing on two key aspects: count-based and context-based performance factors. Section III presents the methodology, which encompasses a background of the proposed RCE, its mathematical formulation, and validation through theorem, including established primitive recursive definitions that support NLP applications. Section IV presents the results and discussion, evaluating the model's performance using two primary correlation metrics. Section V concludes the paper by discussing the implications of the findings and the limitations that inform future research directions.

II. Previous Related Work

The distributional hypothesis posits that the meaning of a word can be inferred from the contexts in which it appears [5]. This principle underpins numerous contemporary word embedding techniques, including Word2Vec, GloVe, and various vector-based semantic models. This hypothesis underpins two established primitive models commonly employed for sentence vector representation.

A. Context Play a Major Role

The hypothesis presented here assumes that context is always distributed. Since words are the fundamental units of meaning, it is crucial to determine how different combinations of words contribute to the overall meaning of a sentence. In this regard, the well-known model Word2Vec has become prominent [5], [11]. This model is built using a neural network that predicts words based on their contexts, effectively training word vectors such that similar words have similar vectors. Sentence structure plays a vital role in evaluating tasks related

to similarity. Various existing methods approach this task by treating words as the basic components of a sentence, converting them into vectors, and providing context to these vectors so that the entire sentence's information can be assessed. Two primary structural strategies have been developed: one that supports sequential structures, such as attention mechanisms, transformer models, and large language models (LLMs) [8], and another that supports non-sequential structures, including graph-based embeddings, kernel-tree-based methods, and ontology-based strategies. All of these context-based approaches have effectively excelled in sentence similarity tasks, showcasing state-of-the-art performance [7].

B. Count is Another Aspect to Support this Distributional Hypothesis

A count-based method creates word vectors using co-occurrence probabilities of words within a corpus [11]. Early approaches to semantic representation, such as Word2Vec, utilized the distributional hypothesis to generate dense word vectors that capture semantic relationships. GloVe, the first developed model [6], built upon this by incorporating co-occurrence statistics, which provided more globally optimized embeddings. This development allowed for a broader consideration of contexts beyond local contexts, while still participating in the distributional hypothesis. Later models introduced enhancements by including global features and improved the performance of sentence similarity tasks through better context understanding.

C. Findings

There is no doubt that context-based implementation has outperformed similarity tasks, especially when advanced models are constructed with deep layer involvement. The incorporation of count-based occurrences within these layers has enhanced the overall performance of the models. The authors of [1] and [9] have contributed to this field by developing the Multilevel Center Embedding (MLCE) concept, which focuses on the level representation of sentences. This approach provides insights into how to resolve word-ordering issues through its structural aspects and successfully addresses the Word Sense Disambiguation (WSD) task.

The recent introduction of Recursive Center Embedding (RCE) offers a novel perspective that extends the MLCE concept. RCE enhances traditional models like Word2Vec by incorporating leveled contexts derived from the hierarchy of center words. This utility demonstrates significant capability in assessing sentence similarity, particularly for sentences with complex structures.

III. Methodology

Center embedding is a concept introduced in our previous work that serves as a novel strategy for capturing the structural complexities of sentence representation [1], [3]. This technique was initially developed to analyze deeply nested sentence structures by embedding clauses within one another [4], thereby offering a hierarchical perspective on sentence composition.

A. Background: Center Embedding (CE) Overview

A linguistic term known as "center embedding" refers to the placement of embedded units (clauses or phrases) within a main sentence [2], [3]. Center embedding can be used in sentence similarity computations to capture hierarchical sentence structures, including nested phrases or clauses, which are typically difficult to express in flat word-vector models like Word2Vec or GloVe [5], [6]. The fundamental principle of center embedding is to take into account the syntactic structures that words form, in addition to their linear sequence. For example, consider the sentence: The quick brown fox, which was very clever, jumped over the lazy dog. Here, the clause which was very clever is embedded within the main clause The quick brown fox jumped over the lazy dog. The center embedding process recognizes this embedded structure.

The primary goal of center embedding is to recursively capture the hierarchical structure of a sentence, creating abstractions at different levels [1]. This approach breaks down sentences into manageable sub-components (i.e. words or clauses), representing their structural and contextual complexity in a leveled manner. RCE explores this idea by recursively processing each word and assessing its position within the sentence. This allows for the construction of word and clause-level embeddings in a manner similar to the previously proposed multilevel center embedding (MLCE), which is built iteratively. At its core, Center Embedding [1] calculates the difference between the original center-embedded version of a sentence and its modified counterparts, creating new levels of abstraction. This method transforms the sentence hierarchy by recursively generating embeddings for sub-sentences at various levels.

Mathematically, it is formulated in previous version by introducing a center embedding for each sentence at various levels iteratively, denoted as C_i^k where $i \in \{1,2\}$ and $k \in \{1,2,..K\}$ represents different parts of a sentence and k is the level of abstraction, with K being the maximum level. It can be expressed by splitting the sentence at the center word and gathering then sentence structure in terms of clauses or phrases on both halves as in previous work [1].

B. Recursive Center Embedding (RCE)

The Recursive Center Embedding (RCE) model formalizes the process of recursively computing sentence embeddings. By motivating iterative procedure of MLCE, following two significant steps of base-level and repeated construction guide this recursive process.

1) Base level construction (level-1): The first level embedding is constructed by averaging the word embeddings of the entire sentence S. For each word of $w_i \in S$, it's embedding x_i is calculated. The level-1 center-embedding of the sentence x_S is then given by Eq. (1):

$$x_S = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{1}$$

where n is the number of words in the sentence. This forms the base-level embedding of the sentence, represented as $x_S^1 = x_S$.

2) Recursive construction (level $k > 1$): For levels greater than 1, the sentence is recursively divided into sub-units. The process involves computing center embeddings of progressively smaller segments (words) of the sentence, refining the overall representation. The sentence S_i is divided into m sub-units (clause/sentence) of approximately equal length, where $m = 2^{k-1}$. The recursive process splits the sentence into multiple components, allowing the model to capture relationships within these smaller units.

For each sub-units $S_{i,j}$ at level $k-1$, the corresponding center embedding is calculated as Eq. (2):

$$x^{k-1}_{S[i,j]} = \phi(S_i|_{k-1}, (k-1)) \tag{2}$$

where ϕ is the recursive embedding function that applies the center embedding to the sub-units at level $k-1$. After obtaining the lower-level embeddings for each sub-unit, overall embedding for level $k-1$ is computed by averaging all sub-unit embeddings as Eq. (3):

$$x^{k-1}_S = \frac{1}{m}\sum_{i,j} x^{k-1}_{S(i,j)} \tag{3}$$

This recursive averaging helps aggregate the contextual information from the sub-units into a more coherent representation. Once the embeddings for the lower levels are computed, the final center embedding for level k is given by Eq. (4):

$$x^k_S = x^{k-1}_S + x^1_S \tag{4}$$

This step combines the information from both the base level (word-level embedding) and the recursive higher levels, allowing the final sentence embedding to incorporate both local word semantics and global sentence structure.

C. Objective Function Interpretation

RCE builds upon the traditional Word2Vec architecture by introducing a recursive mechanism to capture sentence structure. The enhanced objective function, which incorporates a penalty for differences between successive levels of abstraction, ensures that the learned embeddings reflect both the local word order and the global sentence structure, addressing a key limitation of Word2Vec.

1) Skip-gram model objective: The two popular approaches for training Word2Vec are Skip-gram and Continuous Bag-of-Words (CBOW) [5]. Due to the limited ability to capture word order information, CBOW is not used to interpreted here. The basic objective of the Skip-gram model is to predict the context words given a target word [14]. For a given (target) word w_t, the model attempts to maximize the probability of the surrounding context words $w_{t-\mathbb{K}}.....w_{t+\mathbb{K}}$, where $\mathbb{K}$ is the size of the context window. The objective function is defined as maximizing the log-likelihood of the context words given the target word. Mathematically, the objective function is Eq. (5):

$$L = \frac{1}{T}\sum_{t=1}^{T}\sum_{-\mathbb{K}\le j\le\mathbb{K}, j\neq 0} log\, P(W_{t+j}|W_t) \tag{5}$$

where T is the total number of words in the corpus.

2) Enhanced interpretation of objective function for RCE: The objective function for RCE builds on the Word2Vec objective but includes an additional term that incorporates the structural hierarchy captured by the center embeddings. The new objective becomes as Eq. (6):

$$L_{RCE} = L_{Word2Vec} + \lambda \sum_{k=2}^{K} \left\| x^k_s - x^{k-1}_s \right\|^2 \tag{6}$$

where $L_{Word2Vec}$ is the original Skip-gram loss, x^k_s, x^{k-1}_s are the multilevel center embeddings at levels k and $k-1$, and λ is a regularization parameter. This regularization term penalizes large differences between consecutive levels of center embeddings, encouraging the model to smoothly transition between different levels of abstraction. By including this term, the model is guided to focus not only on word similarity but also on the structural coherence [10] of the sentence, thus improving its handling of word order.

a) RCE Maximizes contextual information through add-on contexts: Let S be a sentence composed of n words, and x_S be its recursive embedding obtained through RCE. For each word $w_i \in S$, the RCE algorithm collects context pairs iteratively, assigning additional weight to the center word and recursively gathering surrounding words. The objective function of RCE $\mathscr{L}_{\mathscr{RECC}}$ is optimized by the cumulative collection of add-on contexts, providing a more comprehensive representation of sentence meaning. This maximization of contextual information leads to better performance in sentence similarity tasks, compared to traditional word embedding methods such as Word2Vec.

The objective function $\mathscr{L}_{\mathscr{RECC}}$ is the minimization of the distance between the RCE vector x_S and the true contextual-meaning vector $\widehat{x_s}$ as $\|x_s - \widehat{x_s}\|^2$. In RCE, for each word w_i, context pairs are collected in a recursive manner, where the center word at each level receives additional weight. Denote the word at level k of the recursive process as w^k_i, and the corresponding context pair at that level as $C^k_i = \left(w^k_{i-1}...w^k_{i+1}\right)$. Now, the total context vector from Eq. (2) as $x^k_s = \sum_{i=1}^{n} \phi(w^k_i, C^k_i)$. The RCE algorithm extends the word's context by recursively adding information from further its hierarchy in which non-terminals contains phrases/clauses. These are words outside the immediate context pair C^k_i. The add-on context for a word at level k is denoted by A^k_i. The modified context vector incorporating add-on contexts is then as $x^{k+1}_s = \sum_{i=1}^{n} \phi(w^k_i, C^k_i, A^k_i)$.

The total embedding for the sentence is the recursively compose of the context vectors (using sum) across all levels of hierarchy, from the base level to the final level k_{max}. This includes both the immediate context pairs and add-on contexts collected at each level of recursion. This recursive aggregation of context ensures that each word's representation is informed by both its local and global context within the sentence.

The recursive nature of RCE, combined with the accumulation of add-on contexts, ensures that the objective function $\mathscr{L}_{\mathscr{RECC}}$ is minimized as the collected context approaches the true sentence meaning vector. Since RCE gathers both immediate and extended context, it effectively reduces the

difference, leading to lower loss and better (multi-leveled) sentence similarity performance.

3) To validate proposed theorem using PR function definitions: The RCE method is inherently recursive, relying on recursive steps to break down sentences and build up representations from the sub-units around center words. By employing a class of primitive recursion[1], it is well known paradigm to establish the correctness of this recursion process.

At recursion level $k = 0$, RCE gathers the immediate context pair C_i^0, for each word w_i in sentence S. This is analogous to the Word2Vec approach, which captures context within a fixed window around the target word. The base case for the recursion is defined as $f(0,w_i) = C_i^0 = (w_{i-1}, w_{i+1})$. This captures the immediate neighbors of the word.

RCE recursively collects add-on contexts A_i^k from the sentence. The add-on contexts are words outside the immediate context window that contribute additional information. We can define the recursive function as $f(k+1,w_i) = g(k, f(k,w_i)) + A_i^{k+1}$, where $f(k,w_i)$ is the context pair at recursion levels, $g(k, f(k,w_i))$ computes the semantic contribution of the context at levels, and A_i^{k+1} is the set of new add-on contexts collected. Thus, the full recursive definition of RCE for a word can be expressed in the primitive recursive function under compositions as $f(k+1,w_i) = h(k, f(k,w_i), A_i^{k+1}) = g(k, C_i^k) + A_i^{k+1}$. The function $f(k,w_i)$ grows progressively more informative as levels increases, allowing RCE to capture hierarchical relationships between words.

This validation inspired through superposition and alphabatic PRPF function which are well established by the authors in their work [16], [17]. The Recursive Center Embedding (RCE) function $f(k+1,w_i)$ defined as $h(k, f(k,w_i), A_i^{k+1}) = g(k, C_i^k) + A_i^{k+1}$ for exploring add-on features can also be mapped through Superposition as $F^*(G_1(w_{i-1}, w_{i+1}), G_2(A_i^{k+1}))$, where G_1, G_2 pair immediate left and right neighbors. While the alphabetic-PRPF function H_i recursively combines previous results through its third argument $R = Q_{ai}$ recursively. With this validation, where Superposition ensures the hierarchical aggregation of word pairings and add-on contexts across levels, combining representations and Alphabetic PRPF captures extended hierarchical relationships by recursively combining previous levels' results $f(k,w_i)$ with new contexts A_i^{k+1}.

Therefore, the recursive nature of RCE aligns with the structure of primitive recursive functions, validating the effectiveness of the objective function in collecting and accumulating contexts across different levels of recursion.

D. Features Development in Word2Vec and RCE

Traditional Word2Vec model extracts features based on the immediate context of words to generate embeddings. In contrast, RCE enhances this process by incorporating hierarchical embeddings that capture deeper structural relationships among words, utilizing context pairs from both the entire sentence and its sub-units. For example, if a sentence is The quick brown fox jumps over a lazy dog, using then a context window size of 2, the context pairs for the target word fox would be as The, quick, brown, jumps, over.

Table I observes the features collections, in which the well-known Word2Vec model constructs features in immediate contexts based on window size. While RCE uses contexts and assigns according to the levels in which the center word is targeted, additional features are enhanced here by adding where target words lie either in the sentence or its left or right sub-unit levels. Target center words "fox", "dog", "The", and "over" have no add-on contexts (global information) due to these words are situated at terminals of the center-word's hierarchy, while words "jumps", "brown", "lazy", "quick", and "a" are enhancing Word2Vec contexts (local information) by add-on contexts.

E. Higher-Level Vectors Construction Based on RCE

In NLP, different sentence units (words/phrases/clauses) are considered for performing similarity evaluation where higher levels, such as phrases, clauses/sentences, can be constructed with the help of word vectors [1]. In Word2Vec, the sentence vector is typically calculated as the average of the word vectors in the sentence. For a sentence S consisting of n words, let the embedding of the i^{th} word be denoted as x_i. The sentence vector V_S or a sentence S that is obtained by averaging the word embeddings, is given by Eq. (7):

$$V_S = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{7}$$

RCE enhances the sentence vector by considering the hierarchical structure and word dependencies within the sentence. RCE works by recursively applying center embedding to progressively combine sub-sentences and word embeddings, resulting in a structured sentence representation as Eq. (8):

$$x_S^k = \frac{1}{m}\sum_{i=1}^{m} x_{S(i,j)}^{k-1} + x_S^1 \tag{8}$$

where x_S^1 represents the mean of word embeddings at the first level (sentence level), and recursively dividing the sentence and computing embeddings until the maximum level k s reached, creating a final sentence vector x_S^k that captures both word and higher-level semantics. Based on this formulation in terms of Eq. (1 – 8), we proposed an extension of MLCE (multi-level center embedding) through RCE that utilizes Word2Vec-based context pairs along with hierarchical contexts and enhanced baseline performance of Word2Vec for sentence similarity tasks. The following section will demonstrate this idea over benchmark datasets that conclude RCE effectiveness and generalizability.

IV. Result and Discussion

In this section, we have performed experiments to achieve objectives regarding RCE performance under sentence similarity evaluation and explore the generalizability of how RCE performed different sentence complexity.

[1]A primitive recursive function is a function defined using the base functions (such as the zero function, successor function, and projection function) and the operations of composition and primitive recursion [15].

TABLE I. COMPARATIVELY FEATURE-CONSTRUCTIONS OF BOTH WORD2VEC AND RCE FOR EXAMPLE SENTENCE

w_t	Word2Vec	RCE		
	Context-pairs	**Levels, wt: Center**	**Context-pairs**	**Add-on Contexts**
The	(quick, brown)	L0, jumps	(brown, fox, over, a)	Entire sentence
quick	(the, brown, fox)	L1, brown	(The, quick, fox, jumps)	Sub-unit: left
brown	(The, quick, fox, jumps)	L1, lazy	(over, a, dog)	Sub-unit: right
fox	(quick, brown, jumps, over)	L2, quick	(the, brown, fox)	Sub-unit: left
jumps	(brown, fox, over, a)	L2, fox	(quick, brown, jumps, over)	Sub-unit: []
over	(fox, jumps, a, lazy)	L2, a	(jumps, over, lazy, dog)	Sub-unit: left
a	(jumps, over, lazy, dog)	L2, dog	(a, lazy)	Sub-unit: []
lazy	(over, a, dog)	L3, The	(quick, brown)	Sub-unit: []
dog	(a, lazy)	L3, over	(fox, jumps, a, lazy)	Sub-unit: []

A. Experimental Setups

In order to evaluate the performance of the proposed Recursive Center Embedding (RCE) approach, a thorough experimental setup is designed using a variety of datasets, including both benchmark and author-constructed datasets as shown in Table II. This section outlines the datasets used, the Word2Vec training configuration, and the parameters optimized to improve the effectiveness of RCE for capturing sentence similarities through context-aware representations.

TABLE II. SUMMARY DETAILS OF SIMILARITY ASSESSMENT BENCHMARK DATASETS

Similarity Assessment Datasets		
***SICK* [12]**	***STS* [13]**	***Complex Dataset* [14]**
~10,000 pairs	~8,500 pairs	50 pairs
(1 to 5)	(0 to 5)	Final Similarity up to 5
Marelli et al., 2014	Cer et al., 2017	Chandrasekaran, D. and Mago, V., 2021
A benchmark dataset consisting of sentence pairs annotated with relatedness scores.	A widely used benchmark dataset containing pairs of sentences from diverse domains, each annotated with a score indicating the degree of semantic similarity.	A dataset containing pairs of complex sentences designed to challenge models in assessing sentence similarity for intricate structures.

1) Word2Vec training configuration: For the development of sentence vectors under the RCE framework, the Word2Vec model is trained with an optimized set of parameters to ensure the model captures meaningful context and word relationships effectively. The parameters for the Word2Vec training are as follows in Table III.

TABLE III. WORD2VEC TRAINING CONFIGURATION FOR RCE AT EPOCHS IN THE RANGE OF [50: 500]

Parameter	**Values**	**Description**
Vector Size	[100:300]	The dimensionality of the word vectors to be generated.
Window	[5:25]	The maximum distance between the current and predicted word within a sentence.
Model Type	Skip-gram	The architecture used for training the Word2Vec model, which predicts context words given a target word.
Training Data	flattens-RCE Vocab	According to the provided hierarchy of center-words, the dataset constructs the vocabulary and then flattens it in structure.
Lambda	0.8	A hyper-parameter to control the influence of the regularization term in the objective function.

The combination of these optimized parameters ensures that the trained Word2Vec model can develop RCE supported features through flat vocabs[2] meaningful word vectors that support the RCE's objective of accurately capturing hierarchical and structural sentence relationships.

2) Objective function of RCE: The Recursive Center Embedding (RCE) framework operates by recursively constructing center embeddings from sentences to capture both local word-level dependencies and global sentence structure. The objective function, as Eq. (6) discussed in the methodology section for RCE, is designed to balance the contribution of the recursive context embedding and word-level features. The parameter λ is set to 0.8, indicating that more weight is given to the recursive context structure compared to the individual word embeddings. The recursive process ensures that each word in the sentence is treated in the context of its neighboring words, creating context-pairs. These pairs are used to compute sentence-level vectors that reflect both semantic meaning and structural composition.

B. Significance of Normalize Representation of Sentence Vectors

Normalization of sentence vectors plays a crucial role in ensuring that vector magnitudes are comparable and meaningful for similarity tasks [10]. In Word2Vec, sentence vectors are typically computed as the mean of the word vectors within a sentence. On the other hand, Recursive Center Embedding (RCE) recursively computes a hierarchical embedding that captures the syntactic relationships between words. Normalization brings all vectors to a uniform scale, making it easier to compare vectors from different sentences. It is particularly important for RCE, where the recursive structure introduces multiple levels of abstraction. In mean-based Word2Vec, normalization primarily ensures length-invariance and improves the quality of similarity comparisons. We here performed Euclidean norms for both approaches as Eq. (9).

$$\left.\begin{aligned} \widehat{x_s} &= \frac{x_s}{||x_s||} \\ \widehat{x_{s,k}} &= \frac{x_{s,k}}{||x_{s,k}||} \end{aligned}\right\} \tag{9}$$

where $x_{s,k}$ is recursively obtained by averaging the word embeddings of sub-sentences and combining them as Eq. (9) for the RCE approach while the mean-based sentence vector

[2]Vocab is developed through unique sentences from available benchmark datasets.

x_s utilized for the Word2Vec. Obtained results of norm vectors are shown in both scenarios as in Table IV and Fig. 2, these normalization statistics across the Complex Sentence, SICK, and STS-B datasets suggest important insights about the performance of Recursive Center Embedding (RCE) compared to Word2Vec-based mean sentence vectors.

TABLE IV. NORM-VECTORS STATISTICS OBTAINED DURING EXPERIMENT WHICH IS SHOWN IN FIG. 2A, 2B, 2C ALONG WITH REMARKS ON WHICH DATASETS, SCHEME IS BETTER

Dataset	Word2Vec-Norms	RCE-Norms	Description
Complex-Sentence	Mean:1.1379 Std:0.1582	Mean:11.9599 Std:3.6996	RCE reflects greater captures sentence depth better. Mean-based vectors are uniform due to show a small degree of variation as indicated by standard deviations.
SICK	Mean:1.2048 Std:0.1412	Mean:9.7899 Std:2.6211	
STS-B	Mean:1.3550 Std:0.2977	Mean:9.7747 Std:3.7465	

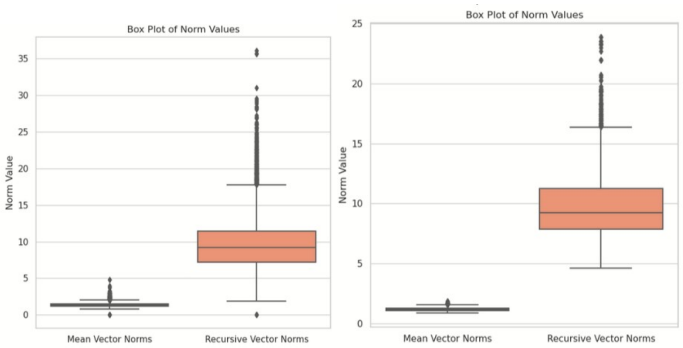
Complex Sentence Dataset is the most suitable choice for showcasing the effectiveness of Recursive Center Embedding (RCE). The significantly higher RCE norms (mean of 11.9599 vs. 1.1379 for Word2Vec) show that RCE captures the intricate structure of complex sentences much better than Word2Vec, while others demonstrate the advantages of RCE over Word2Vec, but the effect is less pronounced.

C. Assessment of Sentence Similarity Task

In normalizing sentence vectors, obtaining mean and deviation results shows a considerable significant variation. These results conclude that RCE can be a good alternative of traditional Word2Vec/BERT-base models if a primary task surrounds the structural and semantics evaluation of complex sentences. With this observation, we evaluated the similarity task in sentence pairs which are in different sizes as shown in Table II. Under the established configurations mentioned in Table III for constructing vectors, which are obtained after applying optimization for getting effective correlations through a grid search approach with various combinations of these hyper-parameters as V: vector_sizes = [100, 200, 300], W: window_size = [5, 10, 15, 20] E: epochs_list = [50, 100, 200, 500, 700].

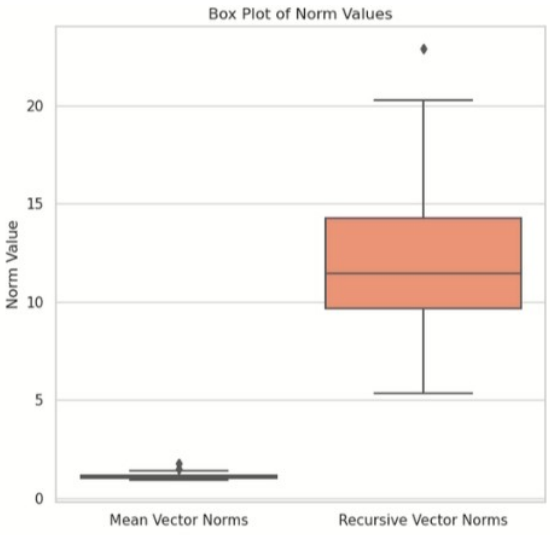
Correlation Results mentioned in Table V are reflected here that the chosen optimized parameters produce vectors that restrict them to over-fitting situations [6, 9]. Based on the observations, we can conclude that the traditional Word2Vec approach has limitations in exploring similarity tasks when sentences have structural complexities.

Another observation is found during assessment in the perspective of obtaining less over-fitting criteria and captured semantics within minimum context window size. Recently claimed complex dataset (having 50 sentence pairs) wins with a window size of 8; under this parameter, RCE effectively achieved its validation criteria. Meanwhile, Word2Vec only validated its over-fitting in terms of Spearman correlation (while Pearson metric still suffer from over-fitting) due to a lower validation score achieved in comparison to the training score. This analysis clearly illustrates the value of utilizing RCE-based hierarchical vectors for effectively determining sentence similarity tasks.

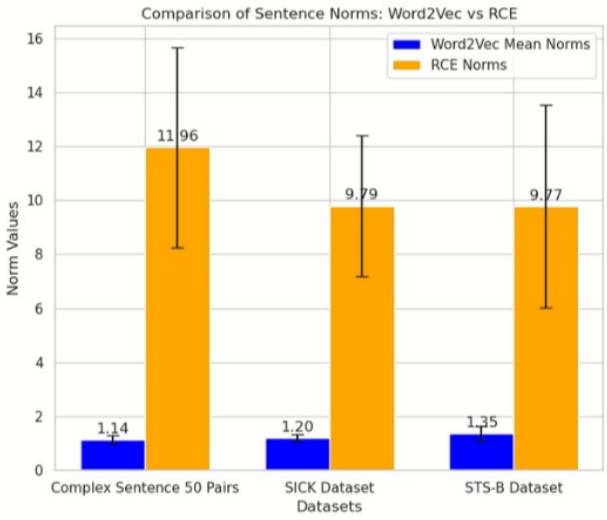


(a) Norms analysis of STS-b

(b) Norms analysis of SICK



(c) Norms analysis of complex dataset



(d) Combined norm-vectors analysis in all datasets

Fig. 2. Results for analyzing which schemes (Word2Vec and REC) effectively capture in deals with hierarchies through comparison of developed Norm-Vectors of sentences in: a) SICK dataset, b) STS-B dataset, c) Complex Sentence dataset, d) Combined comparative results among all datasets.

TABLE V. PEARSON (P_t,P_v) AND SPEARMAN(S_t,S_v) CORRELATION RESULTS OVER TRAINING AND VALIDATION SETS

Datasets	Training Parameters	**Word2Vec**				**RCE**			
		P_t	S_t	P_v	S_v	P_t	S_t	P_v	S_v
SICK	W=20, V=100, E=250	0.58	0.49	0.60	0.54	0.699	0.590	0.748	0.766
STS	W=20, V=100, E=500	0.21	0.23	0.25	0.26	0.518	0.516	0.527	0.538
Complex dataset	W=8, V=100, E=80	0.39	0.42	0.10	0.45	0.506	0.515	0.572	0.587

For comparing the proposed idea of RCE, we have chosen here Google pretrained Word2Vec model and contextual BERT-base pre-trained model on [14], applying mean of word vectors to get sentence information over benchmark datasets. As demonstrated by Table VI's results, the proposed RCE concept performed similarity to the pre-trained word vector model, with a minor improvement in SICK as well in BERT-base variant (particularly related similarities regarding Spearman correlations). Due to having less structural complexity in sentences, RCE poorly performed in the STS-B dataset.

D. Discussion

The Recursive Center Embedding (RCE) approach demonstrated significant advancements in sentence similarity tasks by integrating hierarchical context through recursive decomposition. Unlike traditional Word2Vec methods, which treat sentences as flat bags of words, RCE's recursive mechanism accounted for structural dependencies and nested relationships.

The comparative analysis revealed that RCE outperformed Word2Vec in datasets with intricate sentence structures, such as the Complex Sentence dataset, as evidenced by the significantly higher norm values and correlation scores. Specifically, RCE's ability to capture hierarchical context allowed for a nuanced understanding of sentence meaning, particularly in sentences containing nested clauses and intricate dependencies. For example, RCE achieved a higher Pearson correlation (0.567) on the Complex Sentence dataset, indicating its capability to generalize structural awareness. However, RCE exhibited limited performance on simpler datasets like STS-B, where the added hierarchical complexity was not essential. This suggests that RCE's recursive embedding mechanism may not be the most efficient approach for tasks dominated by surface-level word relationships.

An intriguing observation emerges regarding the proposed concept of RCE, which centers on structured abstraction and demonstrates efficacy over a complex dataset (comprising 50 pairs of sentences, which the authors [14] assert contains heavily complex sentences than other benchmark datasets such as SICK and STS-B), yielding substantial enhancements relative to conventional Word2Vec and the contextual pre-trained BERT-base model.

While RCE significantly improved sentence similarity tasks involving complex structures, its performance on STS-B dataset indicates that further optimization is needed. One promising direction is the incorporation of an adaptive recursion mechanism, which dynamically adjusts the depth of recursion based on sentence complexity. This dataset is still more difficult to analyze for sentence similarity than RCE, which is considered to be less effective, although earlier pre-trained models have done relatively better.

TABLE VI. COMPARISON OF RCE PERFORMANCE TO EXISTING PRE-TRAINED CONTEXTUAL MODELS [WORD2VEC, BERT-BASE VARIANTS (PARTIALLY SUPERVISED)] AT VALIDATED TRAINING PARAMETERS

Datasets	Pre-trained Result(*)		BERT Result(*)	RCE (Lambda = 0.8)	
	P	S	S	P	S
SICK	0.726	0.621	0.728	0.728	**0.736**
STS	0.626	0.587	0.769	0.515	0.521
Complex dataset	0.483	0.490	0.477	**0.567**	**0.551**

* Reported Google News pre-trained results are declared in the work as [14].

V. CONCLUSION AND FUTURE DIRECTION

The Recursive Center Embedding (RCE) approach offers a novel and effective method for sentence representation, particularly for complex-sentence structures. It introduces a recursive mechanism that captures sentence hierarchies in terms of their center-words, embedding them along with Word2Vec according to leveled-contexts gathered through syntactic structure, enabling the model to better handle intricate sentence patterns that are often difficult for traditional word embedding models like Word2Vec, BERT-base variant to capture. In this work, we explored its effectiveness for constructing sentence representations, comparing it with traditional Word2Vec-based mean sentence vectors across three datasets: Complex dataset (Sentence 50 Pairs), SICK, and STS-B.

The results demonstrated the strength of RCE, particularly in handling complex sentence structures, as shown in the significant difference in norm values between the two approaches. RCE significantly outperformed with a much higher norm value, demonstrating RCE's ability to capture intricate sentence structures, especially when dealing with complex sentences containing nested clauses or dependencies. Moreover, in terms of correlation results, highlighting its effectiveness for complex sentence evaluations.

A. Limitations

As observed in the STS-B dataset, RCE under-performed compared to pre-trained Word2Vec when dealing with simple sentence structures. The added complexity of recursively embedding center words may not be necessary for simpler, more straightforward sentence pairs, which rely more on surface-level word similarity. Sentences of these two datasets (specially in STS-B) have only up to limited levels of hierarchy. In all, This suggests that RCE's advantage lies in its ability to handle intricate syntactic structures, limiting its generalization to all types of sentence tasks.

B. Future Directions

Idea of adaptive depth mechanism can improve proposed RCE. Instead of applying the recursive center embedding approach uniformly to all sentences, the depth of recursion could be determined based on the sentence's syntactic complexity. Example: A highly complex sentence like "The student who was reading the book that was recommended by the professor passed the exam," could involve multiple levels of recursion, as it contains nested clauses. On the other hand, a sentence like "The student passed the exam," would only require minimal recursion. The recursive depth could be controlled based on the number of subordinate or relative clauses.

Possible solution for adaptive depth mechanism as to use a sentence parsing technique to identify the number of clauses, and accordingly determine the recursion depth. For sentences with higher clause density, increase the depth of RCE. Conversely, for simpler sentences, limit the recursion to one or two levels. This dynamic depth control can improve efficiency, particularly for large datasets.

REFERENCES

[1] S. Dubey, & N. Kohli, *A Multilevel center embedding approach for sentence similarity having complex structures.* in world conference on communication & computing (WCONF) (pp. 1-8). IEEE, (2023, July).

[2] N. Chomsky, *Logical structure in language*. journal of the American Society for information science, 8(4), 284, (1957).

[3] J.D Thomas, *Center-embedding and self-embedding in human language processing* (Doctoral dissertation, Massachusetts Institute of Technology), (1995).

[4] K.H. Cheon, Y. Kim, H.D. Yoon, K.C.Nam, S.Y. Lee, & H.A. Jeon, *Syntactic comprehension of relative clauses and center embedding using pseudowords.* Brain sciences, 10(4), 202, (2020).

[5] T. Mikolov, I. Sutskever, K. Chen, K., G. S. Corrado, & J. Dean, *Distributed representations of words and phrases and their compositionality.* Advances in neural information processing systems, 26, (2013).

[6] J. Pennington, R. Socher, & C.D. Manning, *Glove: Global vectors for word representation.* In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543), (2014).

[7] H. Yanaka, K. Mineshima, D. Bekki, & K. Inui, *Do neural models learn systematicity of monotonicity inference in natural language?*. In Proceedings of the 58th annual meeting of the association for computational linguistics (pp. 6105-6117), (2020).

[8] T. Ranasinghe, C. Orasan, & R. Mitkov, R. *Semantic textual similarity with siamese neural networks.* In proceedings of the international conference on recent advances in natural language processing (RANLP 2019) (pp. 1004-1011), (2019).

[9] S. Dubey, & N. Kohli, *Clustering for clarity: improving word sense disambiguation through multilevel analysis.* Computer Science, 25(2), 1-24, (2024).

[10] W. Ford, *Numerical linear algebra with applications: Using MATLAB.* Academic press (2014).

[11] D. Yogatama,& N. Smith, *Making the most of bag of words: sentence regularization with alternating direction method of multipliers.* In international conference on machine learning (pp. 656-664). PMLR, (2014).

[12] M. Marelli, L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, & R. Zamparelli, *Semeval-2014 task 1: evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment.* In proceedings of the 8th international workshop on semantic evaluation (SemEval 2014) (pp. 1-8) (2014)

[13] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, & L. Specia, *Semeval-2017 task 1: semantic textual similarity-multilingual and cross-lingual focused evaluation.* arXiv preprint arXiv:1708.00055 (2017).

[14] D. Chandrasekaran, & V. Mago, *Comparative analysis of word embeddings in assessing semantic similarity of complex sentences*. IEEE Access, 9, 166395-166408. (2021).

[15] I. D. Zaslavsky, *On some generalizations of the primitive recursive arithmetic*. theoretical computer science, 322(1), 221-230, (2004).

[16] S. Dubey, & N. Kohli, *Enhancing symbolic manipulation through pairing primitive recursive string functions and interplay with generalized pairing PRSF*. Mathematical Problems of Computer Science, 60, 27-34, (2023).

[17] M. H. Khachatryan, *On generalized primitive recursive string functions.* Mathematical Problems of Computer Science, 43, 42-46, (2015).