Statistically Optimized SGNS Model: Enhancing Word Vector Representation with Global Semantic Weight †

Yulin Liu[1*], Feng Xiong[1], Wanwei Liu[2], Minghui Wu[1]
[1]College of Meteorology and Oceanography,National University of Defense Technology,Hunan,China
[2]College of Computer Science and Technology,National University of Defense Technology Hunan, China
liuyulin@alumni.nudt.edu.cn
{xiongfeng._, wwliu,wuminghui2003}@nudt.edu.cn

Abstract

Addressing the limitations of the Skip-gram with Negative Sampling (SGNS) model related to negative sampling, subsampling, and its fixed context window mechanism, this paper first presents an in-depth statistical analysis of the optimal solution for SGNS matrix factorization, deriving the theoretically optimal distribution for negative sampling. Building upon this analysis, we propose the concept of Global Semantic Weight (GSW), derived from Pointwise Mutual Information (PMI). We integrate GSW with word frequency information to improve the effectiveness of both negative sampling and subsampling. Furthermore, we design dynamic adjustment mechanisms for the context window size and the number of negative samples based on GSW, enabling the model to adaptively capture contextual information commensurate with the semantic importance of the center word. Notably, our optimized model maintains the same time complexity as the original SGNS implementation. Experimental results demonstrate that our proposed model achieves competitive performance aganist state-of-the-art word embedding models including SGNS, CBOW, and GloVe, across multiple benchmark tasks.Compared with the current mainstream dynamic word vector models, this work emphasizes achieving a balance between efficiency and performance within a static embedding framework, and provides potential supplementation and support for complex models such as LLMs.

Keywords: Negative Sampling , Word2vec , Word Embedding , Subsampling , Pointwise Mutual Information

1 Introduction

Word embedding techniques have become a cornerstone of modern Natural Language Processing (NLP), mapping words into low-dimensional, dense vector spaces to effectively capture semantic and syntactic features. The Word2vec framework (Mikolov et al., 2013) marked a significant advance; its efficient shallow neural network architecture achieved breakthroughs in computational efficiency and semantic representation, laying a crucial foundation for subsequent word embedding research. Word2vec comprises two primary training models—Continuous Bag-of-Words (CBOW) and Skip-gram—along with optimization techniques such as subsampling, Hierarchical Softmax, and Negative Sampling. Among these, the Skip-gram model with Negative Sampling **(SGNS)** has emerged as one of the most widely adopted static word embedding models due to its excellent training efficiency and representation capabilities (Levy and Goldberg, 2014; Wang et al., 2020; Qin et al., 2016).

The SGNS model draws inspiration from Noise Contrastive Estimation (NCE), recasting the challenging probability normalization problem into a binary classification task: distinguishing true context words (positive samples) from randomly sampled noise words (negative samples). Its objective is to maximize the vector similarity between center words and their contextual words (positive samples) while minimizing vector similarity with negative samples. Despite its widespread adoption and significant advantages,

†This work is supported by the National Key R & D Program of China No. 2022YFA1005101.

the SGNS model exhibits several key limitations particularly regarding its negative sampling mechanism and associated strategies:

- Insufficient Utilization of Corpus Information: Current negative sampling and subsampling strategies rely solely on global word frequency distributions. However, frequency alone is insufficient for capturing the full informational content of the corpus, potentially leading to the oversampling of frequent but semantically less informative negative samples.

- Static Parameterization: The SGNS model employs fixed sizes for the context window and the number of negative samples. This static configuration cannot adapt to variations in the information content associated with different words or contexts, lacking dynamic adjustment capabilities.

To address these limitations, this paper proposes several key improvements integrated into the SGNS framework:

- Derivation and Application of Global Semantic Weight (GSW): Building upon the theoretically derived optimal negative sampling distribution (detailed in Section 3.2), we introduce Pointwise Mutual Information (PMI) to formulate a Global Semantic Weight (GSW) metric. GSW aims to quantify the global semantic significance of a word within the corpus more effectively than frequency alone.

- GSW-Guided Sampling Strategies: We develop novel negative sampling and subsampling methods that are directly informed by the calculated GSW, prioritizing more informative samples.

- Adaptive Training Parameters: We implement dynamic adjustment mechanisms for both the context window size and the number of negative samples, guided by GSW values. For words with higher GSW values (indicating greater semantic density), the context window is expanded, and the number of negative samples is increased, facilitating a more nuanced training process.

It is true that in recent years, dynamic word embedding models based on the Transformer architecture, notably BERT and the subsequent GPT series, have achieved significant breakthroughs in numerous complex natural language processing tasks. They can flexibly generate contextualized semantic representations. However, the training and deployment of such models are often associated with prohibitive computational requirements, substantial resource consumption, and high inference latency. This has constrained their widespread adoption in resource-constrained scenarios, such as embedded systems, mobile devices, and low-power terminals. Meanwhile, classic static word embedding models, represented by SGNS (Skip-Gram with Negative Sampling), continue to play an indispensable role due to their inherent advantages, including structural simplicity, computational efficiency, and lightweight deployment. They remain crucial for applications prioritizing computational efficiency, for serving low-resource domains, and for providing high-quality initialization parameters for more complex downstream models, including Large Language Models (LLMs). Therefore, continuously optimizing these foundational models is not only vital for deepening theoretical understanding but also holds significant practical importance.

In light of this, the focus of this paper is not to directly compete with state-of-the-art dynamic models in terms of end-performance, but rather to concentrate on the optimization of static word embeddings. Our objective is to provide a word embedding generation method that combines both theoretical superiority and practical efficiency, tailored for high-performance, low-resource applications or as an initialization scheme for more sophisticated models. Drawing upon a deep theoretical analysis of the optimal solution for the SGNS model, we innovatively propose the Global Semantic Weight (GSW). The core idea of GSW is to leverage prior knowledge, distilled from global corpus statistics (in this study, Pointwise Mutual Information or PMI), to guide the model's sampling behavior within local training windows. This specifically encompasses negative sampling, subsampling, and the dynamic adjustment of the context window size. Experimental results demonstrate that our method significantly enhances the quality of the resulting word embeddings enabling them to outperform classic models, including the original SGNS

Proceedings of the 24th China National Conference on Computational Linguistics, pages 972-984, Jinan, China, August 11-14, 2025.

and GloVe, on multiple benchmark tasks?without introducing significant additional computational complexity. More importantly, we contend that this design paradigm of "leveraging global statistics to guide local dynamic training" offers a valuable perspective for optimizing a broader range of machine learning models, including LLMs, a point we will elaborate on in the discussion section.

2 Related works

(Mikolov et al., 2013b) observed that word vectors capture syntactic and semantic regularities, such as linear offset relationships, leading them to propose the Word2Vec model. This model is grounded on the hypothesis: semantically similar words tend to occur in similar contexts (Harris, 1954). Word2Vec employs the Continuous Bag-of-Words (CBOW) and Skip-gram architectures to learn these vector representations. (Pennington et al., 2014) developed the GloVe model, which combines the statistical strengths of global matrix factorization with the computational efficiency of local context window approaches. GloVe generates word vectors through a weighted factorization of the global word-word co-occurrence matrix, also achieving excellent results.

In recent years, considerable research has focused on optimizing Word2Vec models, including both CBOW and SGNS variants. These improvements include incorporating Part-of-Speech (POS) information (Pan et al., 2018), leveraging syntactic dependencies (Deng et al., 2020), enhancing training for low-frequency words (Li et al., 2017), designing adaptive sampling strategies (Chen et al., 2018), and fine-tuning model hyperparameters (Yildiz et al., 2021). Notably, the SGNS model, utilizing the negative sampling technique, has become a foundational model in the field due to its high training efficiency and strong performance, prompting significant efforts to explore its theoretical underpinnings.

A seminal contribution by (Levy and Goldberg, 2014) demonstrated that SGNS implicitly factorizes a matrix related to Pointwise Mutual Information (PMI). Specifically, they showed that the dot product of a center word vector w and a context vector c approximates the shifted PMI: $w \cdot c \approx PMI(w, c) - log_k$. This finding revealed a fundamental connection between neural embedding methods and traditional statistical techniques. Building upon this, (Wang et al., 2020) provided an in-depth statistical analysis of SGNS within a unified Word-Context Classification (WCC) framework. They argued that the implicit PMI matrix factorization represents a special case occurring when the negative sampling distribution $\widetilde{Q}$ is independent of the center word, and they further investigated improving negative sampling using Generative Adversarial Networks (GANs). These theoretical investigations clarified the relationship between SGNS and matrix factorization, influencing the design of subsequent models such as GloVe and symmetric SVD. Despite these developments, SGNS continues to remain a valuable research focus due to its simplicity, efficiency, and sustained competitiveness on various NLP benchmark tasks.

Inspired by prior research connecting SGNS and PMI, this study leverages global PMI statistics to refine and optimize the SGNS model. Our approach specifically targets inherent limitations in negative sample selection, the subsampling strategy, context window sizing.

3 Model

This chapter details the construction and optimization of our proposed word vector representation model. We begin by reviewing the standard Skip-Gram with Negative Sampling (SGNS) model and its operational principles (Section 3.1). Subsequently, we delve into its theoretical underpinnings (Section 3.2), deriving the optimal objective function and analyzing its dependence on the negative sampling distribution, thereby revealing the theoretical motivation for optimizing negative sampling and related training strategies. Following this, we discuss the design principles for an ideal negative sampling distribution and the associated challenges (Section 3.3). We then present the core improvement strategies proposed in this paper in detail: (1) dynamically balanced negative sampling based on Global Semantic Weight (GSW) (Section 3.4); (2) an enhanced subsampling strategy incorporating GSW (Section 3.5); and (3) adaptive context window sizing and negative sample allocation based on GSW (Section 3.6); An overview of our approach is illustrated in Figure 1.

Proceedings of the 24th China National Conference on Computational Linguistics, pages 972-984, Jinan, China, August 11-14, 2025.

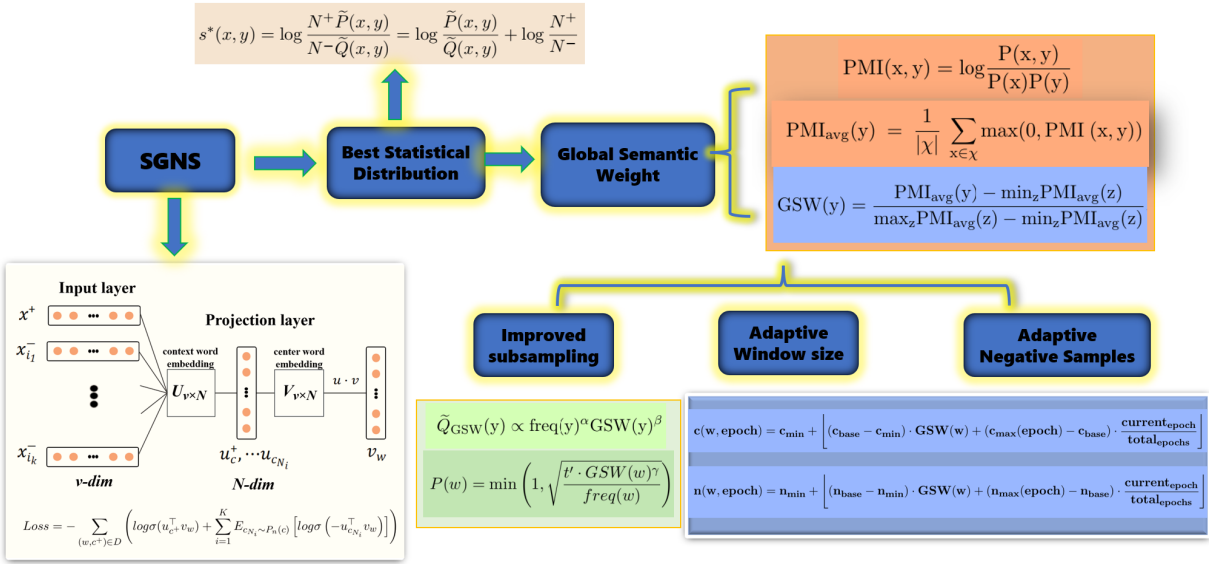

Figure 1: out work

3.1 SGNS Model

The Skip-Gram (SG) model aims to learn word vector representations by predicting the context words surrounding a central word. At its core is a shallow neural network comprising an input layer, a projection layer (hidden layer), and an output layer, and is associated with two key embedding matrices: the central word (input) embedding matrix $V \in R^{Voc \times d}$ and the context word (output) embedding matrix $U \in R^{Voc \times d}$, where Voc is the vocabulary size and d is the word embedding dimension.

For a given center word w (whose vector is $v_w \in V$) and context word c (whose vector is $u_c \in U$), the original SG model uses the Softmax function to define the probability of c appearing in the context of w:

$$P(c|w) = \frac{exp(u_c^T v_w)}{\sum_{d \in Vocab} exp(u_d^T v_w)} \tag{1}$$

The training objective is to maximize the log-likelihood of all observed (w,c) pairs in the corpus: $\Sigma_{(w,c)\in D} \log P(c|w)$. However, the normalization term of Softmax requires traversing the entire vocabulary, which is extremely computationally expensive.

To solve this problem, Mikolov et al. proposed a Negative Sampling optimization strategy, which transforms the problem into a series of binary classification tasks. For each observed positive sample pair (w, c^+), the model simultaneously samples K noise words (negative samples) $\{c_{N_1}, \cdots, c_{N_k}\}$.

These negative samples are typically sampled according to a distribution of word frequencies raised to the power of 3/4: $P_n(c) \propto freq(c)^{3/4}$ .The objective of SGNS is to maximize the similarity (score) of positive sample pairs while minimizing the similarity between the center word and negative samples. Its loss function (minimization target) is typically expressed as:

$$Loss = -\sum_{(w,c^+)\in D} \left(log\sigma(u_{c^+}^\top v_w) + \sum_{i=1}^{K} E_{c_{N_i} \sim P_n(c)} \left[log\sigma\left(-u_{c_{N_i}}^\top v_w\right)\right]\right) \tag{2}$$

3.2 Discussion of Optimal Solutions Based on Statistical Distribution

(Levy and Goldberg, 2014) analyzed SGNS from a statistical perspective, defining the scoring function $s(x, y) = u_y^T v_x$, where x is the center word and y is the context word. When the SGNS model converges, the scoring function $s(x, y)$ it learns approximates the shifted Pointwise Mutual Information (Shifted PMI): $s(x, y) \approx PMI(x, y) - logK$.

We can further derive the optimal scoring function from the perspective of optimization objectives. Let D^+ be the set of positive sample pairs and D^- be the (implicitly defined) set of negative sample pairs,

and the corresponding empirical distributions are: $\widetilde{P}(x,y) = \frac{\#_{D^+}(x,y)}{N^+}$, $\widetilde{Q}(x,y) = \frac{\#_{D^-}(x,y)}{N^-}$, where N^+ and N^- are the total effective number of observations of positive and negative samples respectively ($N^- \approx KN^+$), where $\#_{D^+}(x,y)$ represents the number of times the word-context pair (x,y) appears in the positive sample set D^+, and $\#_{D^-}(x,y)$ represents the number of times the word-context pair (x,y) appears in the negative sample set D^- . The objective function of SGNS can be written as:

$$\ell = -\sum_{x,y}\left[N^+\widetilde{P}(x,y)\log\sigma(s(x,y)) + N^-\widetilde{Q}(x,y)\log\left(1-\sigma(s(x,y))\right)\right] \tag{3}$$

Taking the partial derivative of s(x,y) and setting it to 0:

$$\frac{\partial\ell}{\partial s(x,y)} = -N^+\widetilde{P}(x,y)\cdot\frac{\sigma'(s)}{\sigma(s)} - N^-\widetilde{Q}(x,y)\cdot\frac{-\sigma'(s)}{1-\sigma(s)} = 0 \tag{4}$$

It can be solved that when $Supp(\widetilde{P}) \in Supp(\widetilde{Q})$, the optimal score $s^*(x,y)$ satisfies:

$$\mathrm{e}^{\mathrm{s}^*(\mathrm{x},\mathrm{y})} = \frac{\sigma(\mathrm{s}^*)}{1-\sigma(\mathrm{s}^*)} = \frac{\mathrm{N}^+\widetilde{\mathrm{P}}(\mathrm{x},\mathrm{y})}{\mathrm{N}^-\widetilde{\mathrm{Q}}(\mathrm{x},\mathrm{y})} \tag{5}$$

$$s^*(x,y) = \log\frac{N^+\widetilde{P}(x,y)}{N^-\widetilde{Q}(x,y)} = \log\frac{\widetilde{P}(x,y)}{\widetilde{Q}(x,y)} + \log\frac{N^+}{N^-} \tag{6}$$

This result clearly shows that the optimal scoring function directly depends on the ratio of the positive sample distribution $\widetilde{P}$ and the negative sample distribution $\widetilde{Q}$. Therefore, designing a better negative sampling distribution $\widetilde{Q}$ is crucial for learning a scoring function closer to the ideal and thus obtaining higher quality word vectors.

3.3 Ideal Negative Sampling Distribution Based on Pointwise Mutual Information

From the analysis in Section 3.2, it is known that the construction of the optimal scoring function $s^*(x,y)$ depends on the positive sample distribution $\widetilde{P}(x,y)$ and the negative sample distribution $\widetilde{Q}(x,y)$. In Word2Vec, the word pairs in the negative samples appear independently, and the selection of the context y of the negative samples does not depend on the specific central word x, i.e., $\widetilde{Q}(x,y) = \widetilde{P}(x)\cdot\widetilde{Q}(y)$, where $\widetilde{P}(x) = \sum_{\mathrm{y}}\widetilde{P}(y)$is the marginal distribution of the central word x in the positive samples, and $\widetilde{Q}(y)$ is also the marginal distribution of the context word y in the negative samples. At this time, $Q(y)$ can be designed as a distribution similar to $\widetilde{P}(x)$ (such as $freq(x)^{3/4}$ approximation). This sampling is simple and efficient, but due to the lack of utilization of the central word information, it is easy to introduce a large number of "pure negative samples" that are not related to the semantic space of the positive samples, which leads to the introduction of too much noise in $s(x,y)$, affecting the overall convergence of the model In order to make the distinction between positive samples and negative samples more obvious and the training more efficient, we need to avoid collecting "pure negative samples" that do not overlap with the positive sample distribution as much as possible, that is, we expect the negative sample distribution $Q(y|x) \approx P(y|x)$. From an intuitive point of view, we want to choose some "difficult" negative samples, that is, those context words that are semantically related to the central word but do not belong to the positive samples, in order to maximize the effect of distinguishing between positive and negative samples, instead of simply shifting the vector towards the direction of "over-distinguishing negative samples". Therefore, in order to more effectively distinguish between $\widetilde{P}$ and $\widetilde{Q}$, given the sampling budget, we consider designing a weighted negative sampling model based on Pointwise Mutual Information (PMI). According to the PMI definition:

$$\mathrm{PMI}(\mathrm{x},\mathrm{y}) = \log\frac{\mathrm{P}(\mathrm{x},\mathrm{y})}{\mathrm{P}(\mathrm{x})\mathrm{P}(\mathrm{y})} \tag{7}$$

where $P(x,y) = \frac{\#(x,y)}{N}$, $P(x) = \frac{\#(x)}{N}$, $P(y) = \frac{\#(y)}{\mathrm{N}}$, $\#(x,y)$ represents the number of co-occurrences of th central word x and the context word y in the window in the corpus, $\#(x)$ is the total number of occurrences of the central word x, $\#(y)$ is the total number of occurrences of the context word y, and N represents the total number of words in the corpus.

3.4 PMI Mean-Weighted Dynamic Balanced Negative Sampling Strategy

In order to integrate richer semantic information while ensuring global sampling efficiency, we propose a negative sampling strategy based on Global Semantic Weight (GSW). First, calculate the average Pointwise Mutual Information (Average PMI) of each word y to measure its average association strength with the entire central word set χ :

$$\mathrm{PMI_{avg}(y)} = \frac{1}{|\chi|} \sum_{\mathrm{x}\in\chi} \max(0, \mathrm{PMI\ (x, y)}) \tag{8}$$

If a word y often has a high PMI value with multiple central words, then its corresponding $\mathrm{PMI_{avg}}(y)$ is alsc large, which indicates that the word has a strong semantic signal in multiple contexts and is more likely to become an information-rich negative sample. It should be noted that $\max(0, \mathrm{PMI(x, y)})$ is used here to filter out negative PMI values, because negative PMI value between two words is usually not representative. Then we normalize $\mathrm{PMI_{avg}}(y)$ to obtain the **global semantic weight** : **GSW**(y) $\in [0, 1]$:

$$\mathrm{GSW(y)} = \frac{\mathrm{PMI_{avg}(y)} - \mathrm{min_z PMI_{avg}(z)}}{\mathrm{max_z PMI_{avg}(z)} - \mathrm{min_z PMI_{avg}(z)}} \tag{9}$$

$\mathrm{GSW(y)}$ reflects the relative importance or amount of information of word y in the global semantic space We incorporate GSW into the calculation of the negative sampling probability, and propose a new GSW - weighted negative sampling distribution $\widetilde{Q}_{\mathrm{GSW}}(y)$:

$$\widetilde{Q}_{\mathrm{GSW}}(\mathrm{y}) \propto \mathrm{freq(y)}^{\alpha} \mathrm{GSW(y)}^{\beta} \tag{10}$$

where: $freq(y)$ is the frequency of word y in the corpus, $\alpha, \beta \in [0, 1]$ are smoothing factors, used to adjust the balance between word frequency and global semantic weight, and can be dynamically adjusted according to specific corpus scenarios or downstream task requirements.

We will now analyze the overall time complexity of computing GSW. For the convenience of discussion, let N denote the total number of words in the corpus Corpus Size, V denotes the vocabulary size, C represents the context window size, k is the number of negative samples per positive sample, D stands for the dimension of the word vectors, and E indicates the number of training epochs.

The computation of GSW is a one-time preprocessing step executed before the training begins. Its complexity analysis is as follows:

- PMI Computation Preparation: This step involves calculating the co-occurrence counts for all word pairs by making a single pass over the corpus, resulting in a complexity of $O(N \times C)$.
- PMI_{avg} Computation: This step constitutes the primary computational overhead for GSW. For each word y in the vocabulary V, it is necessary to iterate through the entire vocabulary to compute its average PMI with all possible center words x. Consequently, the complexity of this step is $O(V^2)$, where V is the vocabulary size.
- Normalization to obtain GSW: This involves normalizing all PMI_{avg} values, which has a complexity of $O(V)$.

Therefore, the total complexity of the preprocessing stage is $O(N \times C + V^2) \approx O(V^2)$, as V^2 is typically the dominant term in this phase.

The training complexity of the original SGNS model is analyzed as follows:

The model iterates through every word in the corpus N. For each center word, it processes approximately $2 \times C$ context words as positive samples. For each positive sample (i.e., a center-context word pair), the model performs an update for the positive pair and for k negative samples. The update for the positive sample (involving forward propagation, loss computation, and backpropagation) has a computational cost proportional to the vector dimension D, i.e., $O(D)$. The updates for the k negative

Proceedings of the 24th China National Conference on Computational Linguistics, pages 972-984, Jinan, China, August 11-14, 2025.

samples have a combined cost of $O(k \times D)$. Therefore, the complexity of processing a single center-context word pair is $O((1 + k) \times D)$. Consequently, the complexity for a single training epoch is $O(N \times 2C \times (1+k) \times D)$. Over E epochs, the total training complexity becomes $O(E \times N \times C \times k \times D)$.

In the vast majority of NLP tasks, the corpus size N is significantly larger than the vocabulary size V. For instance, a medium-sized corpus might contain one billion words ($N = 10^9$), while the vocabulary size typically ranges from 10^5 to 5×10^5. Under these circumstances, the order of magnitude of V^2 is approximately $(10^5)^2 = 10^{10}$. In contrast, with N at 10^9, the training complexity (e.g., $E = 5, C = 5, k = 15, D = 300$) can easily reach an order of magnitude of $5 \times (10^9) \times 5 \times 15 \times 300 \approx 10^{14}$. This value, 10^{14}, is several orders of magnitude greater than 10^{10}.

The training complexity is thus several orders of magnitude higher than the GSW pre-computation complexity. Therefore, the overall time complexity of our proposed model remains on the same order of magnitude as the original SGNS implementation, and it is still dominated by the training phase, $O(ENCkD)$. This demonstrates that our proposed method enhances model performance while maintaining the original training efficiency.

3.5 Improved Subsampling Strategy

The subsampling algorithm reduces the retention probability of high-frequency words in the training data through a biased sampling strategy based on word frequency, while accelerating the training process. In the canonical word2vec implementation, the probability of retaining a word instance during training is calculated according to the following formulation:

$$P(w) = \begin{cases} 1 & if\ f(w) < t \\ \sqrt{\frac{t}{freq(w)}} & otherwise \end{cases} \tag{11}$$

where $freq(w)$ represents the frequency of word w in the corpus, and t denotes a predefined frequency threshold, typically set to approximately 10^{-5} in the original word2vec implementation (Mikolov et al., 2013c).

However, this sampling strategy based only on global word frequency has at least two main defects:

- Semantic Information Loss: Certain high-frequency words (e.g., domain-specific terms in specialized corpora or polysemous words with multiple distinct meanings) may encode crucial semantic information. Aggressively downsampling these terms based solely on their frequency can inadvertently eliminate important semantic signals, thereby degrading the model's ability to capture nuanced semantic relationships (Bullinaria and Levy, 2012)
- Context-Insensitivity: By relying exclusively on global frequency statistics, the standard approach disregards the contextual importance of words across different semantic environments, potentially leading to suboptimal representations for words that exhibit significant semantic variation across contexts

To address these limitations and better balance semantic significance against frequency distribution, we propose incorporating the Global Semantic Weight (GSW) metric (as defined in Section 3.4) into the subsampling probability calculation. This leads to our enhanced sampling formulation:

$$P(w) = \min\left(1, \sqrt{\frac{t' \cdot GSW(w)^{\gamma}}{freq(w)}}\right). \tag{12}$$

Where the hyperparameter is used to adjust the influence intensity of GSW on the sampling probability. Generally speaking, the larger the amount of text, the smaller the GSW(w) will be when the window size remains unchanged, the smaller γ needs to be, and t' needs to be appropriately increased, such as setting it to 1 + γ times the original value.

This semantically-informed subsampling approach enables more nuanced word representation learning by preserving semantically significant terms while still effectively managing the computational challenges posed by highly skewed word frequency distributions.

3.6 Adaptive Window Size and Number of Negative Samples

Traditional Skip-Gram models usually adopt a fixed-size context window and a fixed number of negative samples. This static setting fails to fully adapt to the heterogeneous needs of different words for the scope of the context, and fails to effectively utilize the dynamic information in the training process. In order to solve this limitation, we propose an adaptive context window mechanism, whose window radius c(w,epoch) can be dynamically adjusted according to the semantic importance of the central word w and the current stage of training

$$c(w, epoch) = c_{min} + \left\lfloor (c_{base} - c_{min}) \cdot GSW(w) + (c_{max}(epoch) - c_{base}) \cdot \frac{current_{epoch}}{total_{epochs}} \right\rfloor \quad (13)$$

where $c_{\min}$ is the minimum window radius, c_{base} is a basic window radius ($c_{\text{base}} > c_{\min}$) , and GSW($w$) makes words with higher semantic weights tend to have a larger basic window. $c_{\max}(epoch)$ is the maximum window radius that may dynamically increase with the number of training rounds epoch. The overall window size $c(w, epoch)$ is between $c_{\min}$ and $c_{\max}(epoch)$ and is modulated by GSW. This design allows the model to focus on the core local context (smaller window) in the early stage of training. As the training deepens and the word vectors gradually stabilize, the window range is gradually expanded to capture longer-distance dependencies, while allowing words with richer semantic information (high GSW) to use the expanded context information faster or more fully. Similarly, we also design the number of negative samples required for each central word w as an adaptive form:

$$n(w, epoch) = n_{min} + \left\lfloor (n_{base} - n_{min}) \cdot GSW(w) + (n_{max}(epoch) - n_{base}) \cdot \frac{current_{epoch}}{total_{epochs}} \right\rfloor \quad (14)$$

Where n_{min}, n_{base}, n_{max} represent the minimum, base, and maximum number of negative samples that change with the number of rounds, respectively. This mechanism allows the model to allocate more computational resources for negative sampling towards words deemed to have higher semantic importance (high GSW) or those requiring more refined distinction, particularly in the later stages of training, thereby aiming to improve overall training efficiency and effectiveness.

In the initial stages of training, when word vector representations are typically poorly initialized or unconverged, the model benefits from employing a narrower context window. This allows it to prioritize the most core, immediate contextual information to learn the fundamental semantic properties of words. As training progresses, the context window size dynamically expands, driven by the term $(c_{\max}(epoch) - c_{\text{base}}) \cdot \frac{\text{current\_epoch}}{\text{total\_epochs}}$. The rationale is that as vector representations stabilize and converge with further training, moderately expanding the window facilitates the capture of longer-range word dependencies, thereby enabling further refinement of the vector representation quality.

During the gradient update process, the update to the center word vector v_w is influenced jointly by the positive and negative samples. The gradient of the loss with respect to v_w is given by:

$$\nabla_{v_w}\text{Loss} = (\sigma(u_{c^+} \cdot v_w) - 1)u_{c^+} + \sum_{i=1}^{K} \sigma(u_{c_{N_i}} \cdot v_w)u_{c_{N_i}} \quad (15)$$

This gradient dictates the update direction for v_w (typically $v_w \leftarrow v_w - \eta\nabla_{v_w}\text{Loss}$). Consequently, the positive sample term, with its coefficient $(\sigma(u_{c^+} \cdot v_w) - 1) < 0$, tends to pull v_w towards u_{c^+}. Conversely, each negative sample term, with its coefficient $\sigma(u_{c_{N_i}} \cdot v_w) > 0$, tends to push v_w away from $u_{c_{N_i}}$.

In the initial phase of training: Word vectors are typically initialized randomly, resulting in dispersed vector orientations. For a positive pair, the dot product $u_{c^+} \cdot v_w \approx 0$, leading to a relatively large magnitude factor for its update contribution, $|\sigma(u_{c^+} \cdot v_w) - 1| \approx 0.5$. Similarly, for negative samples, $u_{c_{N_i}} \cdot v_w \approx 0$, and their contribution magnitude factor $\sigma(u_{c_{N_i}} \cdot v_w) \approx 0.5$. Thus, even a small number of negative samples (K) can provide sufficient repulsive force, which, combined with the attractive force from the positive sample, guides the vector towards a generally correct orientation.

Proceedings of the 24th China National Conference on Computational Linguistics, pages 972-984, Jinan, China, August 11-14, 2025.

As training progresses, the word vectors tend to converge. According to (Mu et al., 2018), word vectors tend to share a significant common vector component during the convergence process. The model becomes more accurate in predicting positive samples, causing the positive dot product $u_{c^+} \cdot v_w$ to increase, such that $\sigma(u_{c^+} \cdot v_w) \approx 1$. This diminishes the magnitude factor of the positive sample's contribution to the update, $(1 - \sigma(u_{c^+} \cdot v_w)) \approx 0$. Consequently, the "learning signal" from positive samples weakens.

Concurrently, the model may become adept at distinguishing "easy" negative samples, for which the dot product $u_{c_{N_i}} \cdot v_w$ is nearly negative. For these samples, $\sigma(u_{c_{N_i}} \cdot v_w) \approx 0$, and their gradient contribution magnitude also becomes negligible. To continue optimizing the model, perform fine-tuning, and effectively discriminate against "hard" negative samples – those still exhibiting similarity to v_w (i.e., $u_{c_{N_i}} \cdot v_w$ is close to zero or even positive, resulting in $\sigma(u_{c_{N_i}} \cdot v_w)$ being non-negligible) – it becomes beneficial to increase the number of negative samples, K. Increasing K enhances the probability of sampling these "hard" negatives, thereby increasing the expected magnitude of the total negative gradient contribution. This ensures that sufficient optimization force remains, even when the positive signal has attenuated, allowing the model to refine vector positions and achieve finer-grained distinctions.

4 Experiments

4.1 Experimental Setup

To validate model performance under different data scales, the experiment used the text8[1] common benchmark dataset, which is part of the text from Wikipedia and contains about 100 million tokens (the processed size is about 95.3MB). Words with a frequency of less than 20 are excluded. For fair performance comparison, we developed SGNS model code using PyTorch based on the original C language toolkit.[2] We also evaluated different combined configurations of the improvement strategies proposed in this paper, including:

- SGNS + GSW: SGNS + GSW negative subsampling
- SGNS + GSW + AW : SGNS + GSW negative subsampling + adaptive window
- SGNS + GSW + AWN : SGNS + GSW negative subsampling + adaptive window + adaptive negative samples

4.2 Hyperparameter Settings

Our GSW-SGNS model has some parameters the same as the SGNS model, including: initial learning rate $\alpha_{init} = 0.025$, word embedding dimension $d = 200$, training epochs $Epochs = 20$, and random initialization within the range $\left[-\frac{0.5}{d}, \frac{0.5}{d}\right]$. The difference lies in that for the SGNS model, the fixed window size is set to 5, wheDreas in our model it is configured as: $c_{\min} = 1, c_{\text{base}} = 7, c_{\max} = 9$. Furthermore, the baseline model employs a fixed number of negative samples $K = 10$, while our model implements: $n_{\min} = 5, n_{\text{base}} = 10, n_{\max} = 20$. Both approaches share the subsampling threshold $t = 10^{-5}$, but our model introduces an additional parameter $\gamma = 0.75$. In the adjustment of our negative sampling strategy, the parameters $\alpha = 0.75$ and $\beta = 0.5$.

4.3 Word Vector Evaluation Criteria

Word embeddings can be evaluated on intrinsic and extrinsic tasks. Extrinsic evaluation primarily assesses the improvement in word vectors' performance in real-world tasks, such as sentiment analysis and text classification, using metrics like accuracy, precision, and recall to judge the training quality of the word vectors. Intrinsic evaluation, on the other hand, starts from the semantics of the words themselves, verifying whether the semantic relationships captured by the word vectors align with human cognition. Typical tasks for intrinsic evaluation include word similarity tasks, word analogy reasoning tasks, and semantic clustering (Levy and Goldberg, 2014b). Our experiment employs intrinsic evaluation tasks.

[1] https://dataset.bj.bcebos.com/word2vec/text8.txt
[2] https://code.google.com/archive/p/word2vec/

4.3.1 Word Similarity Task

The goal of this task is to measure the consistency between the geometric distance in the word vector space (typically cosine similarity) and subjective human judgments of semantic similarity between corresponding words. The core assumption is that words with similar meanings should also be located close to each other in the vector space. The experiment utilizes several well-established benchmark datasets:

- WordSim-353 (Finkelstein et al., 2001): A widely used dataset containing 353 word pairs and human-annotated similarity scores.
- MEN Test Set (Bruni et al., 2014): Contains 3000 word pairs, labeled with their relatedness.
- SimLex-999 (Hill et al., 2015): Focuses more on evaluating pure "similarity" rather than "relatedness" or "association," containing 999 word pairs.
- SimVerb-3500 (Gerz et al., 2016): Contains 3500 verb word pairs and their similarity scores.

To quantify the correlation between the similarity predicted by the model (cosine similarity of word vectors) and the human-annotated similarity scores, we employ the Spearman Rank Correlation Coefficient. The Spearman Rank Correlation Coefficient builds upon the Pearson Correlation Coefficient and assesses whether the relationship between two variables can be described by a monotonic function. The calculation method involves first converting the original data (x_i, y_i) into ranks (R_{x_i}, R_{y_i}) and then calculating the Pearson correlation coefficient of these ranks.

$$r_s = \frac{cov(R_X, R_Y)}{\sigma_{R_X}\sigma_{R_Y}} \tag{16}$$

Compared to the Pearson coefficient, the Spearman coefficient is less sensitive to the distribution and outliers in the original data and is better at capturing non-linear monotonic relationships, making it particularly suitable for evaluating word similarity tasks. The higher the absolute values of the two Spearman Rank Correlation Coefficient, the better the model performance.

Table 1: Word Similarity Task Results

Model \ dataset	Wordsim353	simverb3500	SimLex-999	MEN-3k-1	MEN-3k-2
Glove	0.524	0.104	0.115	0.521	0.429
CBOW	0.624	0.113	0.218	0.652	0.649
SGNS	0.712	0.155	0.279	0.672	0.699
SGNS+GSW	0.716	**0.182**	0.307	0.697	0.720
SGNS+GSW+AW	**0.748**	0.181	0.287	**0.701**	0.725
SGNS+GSW+AWN	0.743	0.171	**0.308**	0.695	**0.727**

4.3.2 Word Analogy Task

This task aims to evaluate the model's ability to capture the similarity of relationships between words, which typically reflects underlying linguistic patterns (e.g., semantic relations like "country-capital" or grammatical relations like "singular-plural"). The task objective is to simulate the ability of humans to perform analogical reasoning through vector operations. We employ the widely used Google Analogy dataset, which is a classic evaluation tool for testing the performance of word embedding models in analogy reasoning tasks. This dataset contains 19,544 analogy questions, divided into 14 relationship categories, including 9 syntactic morphological categories (denoted as gram1 gram9) and 5 semantic categories (denoted as sem1 sem5).

For a given triple (A, B, C) , the task is to find word D such that the vector relationship $v_B - v_A \approx v_D - v_C$ holds. In practice, we typically seek the word D whose vector v_D has the highest cosine similarity to the target vector $v_B - v_A + v_C$ (excluding A, B, and C themselves). Evaluation Metric:

Performance is measured by Top-1 accuracy, which is the percentage of analogy questions for which the model's top-ranked prediction is the correct answer.

Table 2: Word Analogy Task Results (syntactic)

Model \ dataset	G1	G2	G3	G4	G5	G6	G7	G8	G9
Glove	02.12	13.08	35.24	12.68	27.62	80.04	15.64	46.58	27.70
CBOW	10.12	10.37	33.21	16.34	28.32	78.69	28.38	51.34	30.95
SGNS	10.57	08.11	34.76	13.66	**31.86**	72.06	19.85	51.98	21.85
SGNS+GSW	12.30	**17.97**	45.67	**28.26**	25.30	81.68	29.61	54.44	28.00
SGNS+GSW+AW	11.51	15.36	**49.05**	26.09	27.32	82.13	**34.83**	52.22	26.15
SGNS+GSW+AWN	**16.40**	15.03	47.46	23.73	30.95	**86.58**	32.28	**58.17**	**28.92**

Table 3: Word Analogy Task Results (semantic)

Model \ dataset	S1	S2	S3	S4	S5	Average(all)
Glove	50.60	21.06	03.10	30.94	64.37	32.88
CBOW	81.13	70.24	09.48	**49.34**	51.13	38.34
SGNS	75.60	61.49	13.03	40.60	50.52	40.01
SGNS+GSW	**88.74**	**71.56**	19.40	48.95	62.42	47.20
SGNS+GSW+AW	86.76	67.69	17.91	43.35	**65.36**	47.29
SGNS+GSW+AWN	85.38	69.03	**22.40**	44.94	62.09	**48.48**

Table 2 and Table 3 present the Top-1 accuracy of different models on the word analogy task. G1∼G9 represent 9 syntactic morphological categories, and S1∼S5 represent 5 semantic categories. The Average column is the average accuracy across all categories. The table shows the performance of different models in each category, as well as the overall average accuracy. Again, SGNS is the base model, and GSW, AW, and AWN are different improvement strategies.

5 Conclusions and Future works

In this paper, we proposed a series of optimization approaches based on Global Semantic Weight (GSW), with particular emphasis on improving subsampling and negative sampling techniques. The experimental results from word similarity and word analogy tasks demonstrate that models incorporating the GSW optimization significantly outperform the original SGNS baseline model. In word similarity tasks, the SGNS+GSW+AWN model achieved an optimal score of 0.748 on the Wordsim353 dataset, representing a 5.1% improvement over the original SGNS score of 0.712. For word analogy tasks, the SGNS+GSW+AWN model attained an average accuracy of 48.48%, surpassing all other models and showing notable improvement compared to the baseline SGNS model's 40.01%.

Due to resource constraints, our evaluation was limited to the text8 dataset. Future work could involve training the model on larger corpora such as Common Crawl or the complete Wikipedia corpus to further validate the universality and scalability of our proposed strategies. We anticipate that as data scale increases, the advantages of our model will become even more pronounced. Additionally, the GSW and AWN strategies introduced in this paper bring new hyperparameters, such as the parameters in subsampling and the window size adjustment functions in AWN. Systematically exploring optimal combinations of these parameters through methods like grid search or Bayesian optimization may further enhance model performance. The design philosophy embodied by GSW can also offer valuable insights for the optimization of Large Language Models (LLMs). The pre-training of LLMs relies on vast amounts of text data, yet not all data is of equal value. In real-world corpora with highly skewed data distributions, the ability to efficiently select samples and contexts with higher information density and

stronger representative power directly impacts a model's downstream performance and generalization capabilities. Current data curation strategies, however, often focus on syntactic or superficial statistical features.

First, in terms of pre-training data processing, GSW can provide an efficient and theoretically grounded metric for tasks such as data selection, hard-negative mining, and sample re-weighting for LLMs. By introducing global semantic statistical weights, it allows for the identification of high-value data. For instance, sentences containing multiple words with high GSW values typically carry richer semantic associations. During the data loading phase of pre-training, these information-dense samples can be prioritized for sampling or assigned higher weights, thereby achieving more efficient model training within the same computational budget.

Second, at the level of training strategy, the global importance represented by GSW can be used to guide the negative sampling process in LLMs. By selecting more informative and challenging negative samples, it can effectively enhance the model's discriminative power and reduce inefficient gradient computations. This approach holds potential value for contrastive learning, the construction of self-supervised losses, and the development of efficient negative sampling strategies during large-scale pre-training.

In summary, the method proposed in this paper focuses on enhancing the efficiency and semantic precision of static word embedding models. At the same time, its core philosophy maintains excellent compatibility with and offers broad extension possibilities for advanced model paradigms. Future work could explore the deep integration of the GSW philosophy with the dynamic training techniques of LLMs, aiming to further unlock the potential of different model paradigms in their respective advantageous scenarios and to achieve an organic unification of efficiency, low-resource consumption, and high performance.

References

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. *Efficient Estimation of Word Representations in Vector Space*. arXiv preprint arXiv:1301.3781.

Omer Levy and Yoav Goldberg. 2014. *Neural Word Embedding as Implicit Matrix Factorization*. *Journal of Machine Learning Research*, volume 15, number 1, pages 2177–2235.

Li Wang, Wei Zhang, and Ming Liu. 2020. *Skip-gram Based Word Embedding Enhancement*. *Computational Linguistics and Chinese Language Processing*, volume 25, number 1, pages 1–20.

Tao Qin, Wenyuan Li, Sheng Chen, and Tie-Yan Liu. 2016. *A Novel Approach for Word Embedding with Semantic and Syntactic Constraints*. arXiv preprint arXiv:1603.00892.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. *Linguistic Regularities in Continuous Space Word Representations*. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751. Association for Computational Linguistics.

Zellig S. Harris. 1954. *Distributional Structure*. *Word*, volume 10, number 2-3, pages 146–162.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. *GloVe: Global Vectors for Word Representation*. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Shuming Pan, Qiang Liu, and Maosong Sun. 2018. *An Improved Word Embedding Model with Syntactic and Semantic Regularization*. arXiv preprint arXiv:1805.02288.

Haotian Deng, Peng Li, Yuchen Zhang, and Haifeng Wang. 2020. *Improving Word Embeddings by Exploiting Syntactic and Semantic Information*. *Computational Linguistics*, volume 46, number 2, pages 357–385.

Jiwei Li, Dan Jurafsky, and Christopher D. Manning. 2017. *Improving Word Embeddings via Global Context and Multiple Word Prototypes*. arXiv preprint arXiv:1704.01946.

Qingyun Chen, Qiang Liu, and Maosong Sun. 2018. *Improving Word Embeddings with Subword Information*. *Transactions of the Association for Computational Linguistics*, volume 6, pages 257–270.

Omer Yildiz, Firat Can, and Aysen Coltekin. 2021. *Improving Word Embeddings for Code-Switching Speech via Language-Specific Subword Units*. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 268–278. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. *Distributed Representations of Words and Phrases and their Compositionality*. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

John A. Bullinaria and Roger Levy. 2012. *Extracting Semantic Representations from Word Co-occurrence Statistics: Something Old, Something New*. *PLoS ONE*, volume 7, number 6, pages e39412. Public Library of Science.

Omer Levy and Yoav Goldberg. 2014. *Dependency-based word embeddings*. arXiv preprint arXiv:1404.2188.

Lori Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zohar Solan, Guy Wolfman, and Eytan Yakir. 2001. *Placing Search in Context: The Concept Locus*. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 406–413. ACM.

Jiaqi Mu, Suma Bhat, and Pramod Viswanath. 2018. *All-but-the-Top: Simple and Effective Postprocessing for Word Representations*. arXiv preprint arXiv:1702.01417. Available at https://arxiv.org/abs/1702.01417.

Elia Bruni, My T. Tran, and Marco Baroni. 2014. *A Multimodal Vector Space Model for Words and Images*. *Journal of Artificial Intelligence Research*, volume 51, pages 141–173.

Fiona Hill, Rainer Reuber, and Anna Korhonen. 2015. *SimLex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation*. *Computational Linguistics*, volume 41, number 4, pages 665–695.

Diana Gerz, Iryna Gurevych, and Manfred Pinkal. 2016. *SimVerb-3500: A Corpus and a Lexicon for Verb Similarity*. arXiv preprint arXiv:1603.00892.