

# GloVe

*Global Vectors for Word Representation*

NLP Students

## Contents

<b>1</b>	<b>Introduction: The Landscape of Word Embeddings</b>	<b>2</b>
1.1	The Two Paradigms . . . . .	2
1.2	The GloVe Synthesis . . . . .	2
<b>2</b>	<b>The Foundation: The Co-occurrence Matrix</b>	<b>2</b>
2.1	Constructing the Matrix . . . . .	2
2.2	Example Matrix . . . . .	3
<b>3</b>	<b>The Key Insight: Ratios of Probabilities</b>	<b>3</b>
3.1	The "Ice" and "Steam" Example . . . . .	3
3.2	Interpreting the Ratios . . . . .	4
<b>4</b>	<b>Mathematical Formulation: Deriving the Objective</b>	<b>4</b>
4.1	Step 1: The Initial Postulate . . . . .	4
4.2	Step 2: Enforcing Linear Structure (Vector Differences) . . . . .	4
4.3	Step 3: Capturing Similarity (Dot Product) . . . . .	4
4.4	Step 4: The Homomorphism Requirement (The Exponential Solution) . . . . .	5
4.5	Step 5: The Log-Bilinear Model . . . . .	5
4.6	Step 6: Handling Symmetry and Bias . . . . .	5
<b>5</b>	<b>The GloVe Objective: Weighted Least Squares</b>	<b>5</b>
5.1	The Least Squares Approach . . . . .	6
5.2	The Weighting Function $f(X_{ij})$ . . . . .	6
5.3	The Final GloVe Objective Function . . . . .	6
<b>6</b>	<b>Training and Implementation Details</b>	<b>7</b>
6.1	Optimization . . . . .	7
6.2	Computational Complexity . . . . .	7
6.3	The Two Sets of Vectors . . . . .	7
<b>7</b>	<b>Summary: GloVe vs. Word2Vec</b>	<b>7</b>
<b>8</b>	<b>Numerical Example: A Single Training Step</b>	<b>8</b>
<b>9</b>	<b>Conclusion</b>	<b>9</b>

# 1 Introduction: The Landscape of Word Embeddings

The goal of word embeddings is to transform language into geometry, encoding semantic meaning as positions and directions in a high-dimensional vector space. Before GloVe (Pennington et al., 2014), the field was dominated by two distinct approaches, each with significant drawbacks.

## 1.1 The Two Paradigms

### 1. Global Matrix Factorization Methods (e.g., Latent Semantic Analysis - LSA)

These methods utilize the global statistics of the entire corpus. They typically construct a large matrix (e.g., word-document or word-word co-occurrence) and then use techniques like Singular Value Decomposition (SVD) to reduce the dimensionality.

- **Pros:** Efficiently leverages global statistical information.
- **Cons:** The resulting vectors often performed poorly on analogy tasks (like the famous  $King - Man + Woman = Queen$ ), indicating issues with the linear structure of the vector space.

### 2. Local Context Window Methods (e.g., Word2Vec: Skip-Gram and CBOW)

These methods slide a window across the corpus, learning embeddings by predicting words based on their local neighbors.

- **Pros:** Demonstrated superior performance on analogy tasks, capturing fine-grained semantic and syntactic relationships.
- **Cons:** They fail to utilize global statistics effectively. The model analyzes local windows independently, ignoring the vast amount of information contained in the overall counts of how often words appear together globally.

## 1.2 The GloVe Synthesis

GloVe (Global Vectors) was designed to combine the best of both worlds. It aims to achieve the linear substructures that made Word2Vec successful, but does so by training directly on the global co-occurrence statistics of the corpus.

### The Intuition: Building the Map Efficiently

Word2Vec learns the map of language by taking billions of tiny, local glances (windows) at the terrain. GloVe starts by first compiling a comprehensive statistical summary of the entire terrain (the co-occurrence matrix) and then draws the map based on those statistics. This allows GloVe to be much more efficient during training while capturing global relationships.

# 2 The Foundation: The Co-occurrence Matrix

The starting point for GloVe is the global word-word co-occurrence matrix, denoted by  $X$ .

## 2.1 Constructing the Matrix

Let  $V$  be the size of the vocabulary. The matrix  $X$  is a  $V \times V$  matrix.

- $X_{ij}$  is the count of how often word  $j$  appears in the context of word  $i$  across the entire corpus.
- "Context" is defined by a window size (e.g., 5 words before and 5 words after).

## Statistical Foundation: Co-occurrence Probabilities

We can derive probabilities from these counts.  $P_{ij}$  (or  $P(j|i)$ ) is the probability that word  $j$  appears in the context of word  $i$ .

$$P_{ij} = P(j|i) = \frac{X_{ij}}{X_i}$$

Where  $X_i = \sum_k X_{ik}$  is the total count of all words appearing in the context of word  $i$ .

## 2.2 Example Matrix

Consider the corpus: "I like cats. I like dogs. I love dogs." (Assuming a window size of 1 and ignoring punctuation).

	I	like	love	cats	dogs
I	0	2	1	0	0
like	2	0	0	1	1
love	1	0	0	0	1
cats	0	1	0	0	0
dogs	0	1	1	0	0

For example,  $X_{I,like} = 2$ . The probability  $P(like|I) = 2/3 \approx 0.66$ .

### Important Note: Symmetry and Weighting

In practice, the co-occurrence matrix is often constructed symmetrically ( $X_{ij} = X_{ji}$ ). Furthermore, GloVe uses a weighting scheme when building the matrix: words that are closer together within the window contribute more to the count than words further apart (e.g., using harmonic weighting  $1/d$ , where  $d$  is the distance).

## 3 The Key Insight: Ratios of Probabilities

If we want our word vectors to capture meaning, what aspect of the co-occurrence matrix is the most informative signal?

One might assume the raw probabilities ( $P_{ij}$ ) are the best signal. However, the core innovation of the GloVe model is the realization that **ratios** of co-occurrence probabilities are far more effective at encoding meaning and establishing the linear structures required for analogy tasks.

### 3.1 The "Ice" and "Steam" Example

This canonical example from the GloVe paper illustrates the concept perfectly. We want to understand the relationship between  $i = \text{ice}$  and  $j = \text{steam}$ .

We introduce a "probe word"  $k$  and examine the ratio  $\frac{P(k|\text{ice})}{P(k|\text{steam})}$ .

Probe Word ( $k$ )	$P(k \text{ice})$	$P(k \text{steam})$	Ratio
$k = \text{solid}$	High (e.g., 0.001)	Low (e.g., 0.00001)	Very High (100)
$k = \text{gas}$	Low (e.g., 0.00001)	High (e.g., 0.001)	Very Low (0.01)
$k = \text{water}$	High (e.g., 0.005)	High (e.g., 0.005)	Near 1 (1.0)
$k = \text{fashion}$	Low (e.g., 0.00001)	Low (e.g., 0.00001)	Near 1 (1.0)

## 3.2 Interpreting the Ratios

- If the ratio is very high ( $\gg 1$ ), the probe word  $k$  (solid) is strongly related to the numerator (ice) and weakly related to the denominator (steam).
- If the ratio is very low ( $\ll 1$ ), the probe word  $k$  (gas) is weakly related to the numerator (ice) and strongly related to the denominator (steam).
- If the ratio is near 1, the probe word is either related to both (water) or unrelated to both (fashion). It doesn't help distinguish between them.

### The Intuition: Meaning Through Comparison

The raw probabilities alone are noisy. Both "ice" and "steam" are related to "water". But it is the *comparison* (the ratio) against other words that defines their specific concepts. The ratio effectively filters out the noise (like the common association with "water") and isolates the distinguishing features (solid vs. gas).

This observation forms the starting point for the GloVe model: the relationship between two words ( $i$  and  $j$ ) can be derived by examining the ratios of their co-occurrence probabilities with various probe words  $k$ .

## 4 Mathematical Formulation: Deriving the Objective

We now seek a mathematical model that can generate these probability ratios based on word vectors. Let  $v_i$  and  $v_j$  be the vectors for the primary words, and  $u_k$  be the vector for the probe (context) word.

### 4.1 Step 1: The Initial Postulate

We want a function  $F$  that takes the vectors as input and outputs the ratio:

$$F(v_i, v_j, u_k) = \frac{P_{ik}}{P_{jk}} \quad (1)$$

### 4.2 Step 2: Enforcing Linear Structure (Vector Differences)

We want to encode meaning in the *differences* between vectors (like  $v_{King} - v_{Man}$ ). Therefore, the function  $F$  should operate on the difference between  $v_i$  and  $v_j$ :

$$F(v_i - v_j, u_k) = \frac{P_{ik}}{P_{jk}} \quad (2)$$

### 4.3 Step 3: Capturing Similarity (Dot Product)

The most natural way for vectors to interact in this space is the dot product, which measures similarity. We reformulate  $F$  to take the dot product of the difference and the probe vector:

$$F((v_i - v_j)^T u_k) = \frac{P_{ik}}{P_{jk}} \quad (3)$$

## 4.4 Step 4: The Homomorphism Requirement (The Exponential Solution)

This is the most abstract step. We need to maintain symmetry between the roles of "center word" and "context word" (since the co-occurrence matrix  $X$  is symmetric). The distinction between  $v$  (main vectors) and  $u$  (context vectors) is arbitrary.

If we examine the structure of Eq. 3:

$$F(v_i^T u_k - v_j^T u_k) = \frac{P_{ik}}{P_{jk}}$$

We are looking for a function  $F$  that transforms subtraction on the left into division on the right, i.e.,  $F(A - B) = F(A)/F(B)$ . This is known as a homomorphism between the additive group of real numbers and the multiplicative group of positive real numbers. The solution to this functional equation is the exponential function,  $F(x) = e^x$ .

### The Math: The Exponential Breakthrough

Substituting  $F = \exp$  into the equation:

$$\exp(v_i^T u_k - v_j^T u_k) = \frac{\exp(v_i^T u_k)}{\exp(v_j^T u_k)}$$

And aligning with the goal:

$$\frac{\exp(v_i^T u_k)}{\exp(v_j^T u_k)} = \frac{P_{ik}}{P_{jk}}$$

## 4.5 Step 5: The Log-Bilinear Model

This implies a direct relationship for each individual term:

$$\exp(v_i^T u_k) = P_{ik} = \frac{X_{ik}}{X_i}$$

Taking the logarithm of both sides:

$$v_i^T u_k = \log(P_{ik}) = \log(X_{ik}) - \log(X_i)$$

## 4.6 Step 6: Handling Symmetry and Bias

The term  $\log(X_i)$  depends only on  $i$  and is independent of the context word  $k$ . We can absorb this term into a bias term  $b_i$ . We also add a corresponding bias  $\tilde{b}_k$  for the context word  $u_k$  to restore full symmetry (as the equation  $v_i^T u_k + \log(X_i) = \log(X_{ik})$  is not symmetric if we swap  $i$  and  $k$ ).

### The Math: The Log-Bilinear Relationship

$$v_i^T u_k + b_i + \tilde{b}_k = \log(X_{ik})$$

This is the core relationship GloVe aims to learn. The dot product of two vectors (plus biases) should equal the logarithm of their co-occurrence count.

## 5 The GloVe Objective: Weighted Least Squares

We now have a target relationship (the Log-Bilinear Relationship). How do we train the vectors  $v_i, u_k$  and biases  $b_i, \tilde{b}_k$  to satisfy this?

## 5.1 The Least Squares Approach

GloVe formulates this as a regression problem. We want to minimize the squared difference between the model's prediction ( $v_i^T u_j + b_i + \tilde{b}_j$ ) and the ground truth ( $\log(X_{ij})$ ).

$$J_{unweighted} = \sum_{i,j=1}^V (v_i^T u_j + b_i + \tilde{b}_j - \log(X_{ij}))^2$$

This looks like a standard Mean Squared Error (MSE) loss. However, this unweighted approach has major flaws:

1. **The Zero Problem:** If  $X_{ij} = 0$  (which is extremely common, as the matrix is sparse),  $\log(X_{ij})$  is undefined ( $\log(0) = -\infty$ ).
2. **Equal Weighting:** It treats all co-occurrences equally. A rare pair (e.g., "quantum", "entanglement") occurring once is treated the same as a very frequent pair (e.g., "the", "and") occurring 10,000 times.

## 5.2 The Weighting Function $f(X_{ij})$

To address these issues, GloVe introduces a weighting function  $f(X_{ij})$  into the loss function. This function must have specific properties:

1.  $f(0) = 0$ . This solves the  $\log(0)$  problem. If the weight is 0, the entire term is ignored.
2.  $f(x)$  should be non-decreasing. More frequent co-occurrences should generally carry more weight.
3.  $f(x)$  should saturate (level off) for large values of  $x$ . This prevents extremely frequent stop words from dominating the training process.

### The Math: The GloVe Weighting Function

The function proposed by the authors is:

$$f(x) = \begin{cases} (x/x_{max})^\alpha & \text{if } x < x_{max} \\ 1 & \text{otherwise} \end{cases}$$

Typical parameters are  $x_{max} = 100$  and  $\alpha = 0.75$ .

The  $\alpha = 0.75$  (similar to the smoothing used in Word2Vec's negative sampling distribution) helps to increase the relative weight of rarer co-occurrences.

## 5.3 The Final GloVe Objective Function

Incorporating the weighting function gives us the final objective function that GloVe minimizes:

### The Math: The GloVe Loss Function (Weighted Least Squares)

$$J(\theta) = \sum_{i,j=1}^V f(X_{ij}) (v_i^T u_j + b_i + \tilde{b}_j - \log(X_{ij}))^2$$

## 6 Training and Implementation Details

### 6.1 Optimization

The objective function  $J(\theta)$  is minimized using stochastic gradient descent (SGD) or variants like AdaGrad. Although GloVe is often described as a "count-based" or "matrix factorization" model, the optimization process (learning the parameters  $v, u, b, \tilde{b}$ ) is iterative and very similar to training a neural network.

### 6.2 Computational Complexity

The computational complexity of GloVe is determined by the number of non-zero elements in the matrix  $X$ , denoted  $|X_{nnz}|$ . This is because  $f(0) = 0$ . This is significantly smaller than the total number of tokens in the corpus, which is what Word2Vec iterates over. This makes GloVe training substantially faster for large corpora.

#### Important Note: Iterating over Counts, Not Tokens

If ("cat", "meow") occurs 50 times in the corpus:

- Word2Vec processes this pair 50 separate times during training iterations.
- GloVe processes this pair only once in the summation, but with a weight  $f(50)$  in the loss function.

### 6.3 The Two Sets of Vectors

Like Word2Vec, GloVe learns two sets of vectors: the main vectors  $v$  (from matrix  $W$ ) and the context vectors  $u$  (from matrix  $\tilde{W}$ ).

Since the roles are symmetric, the initializations are random, and the optimization process is the same,  $W$  and  $\tilde{W}$  are theoretically equivalent. However, due to random initialization and the stochastic nature of optimization, they differ in the final result.

The GloVe authors found that summing the two vectors provides a more robust final embedding:

$$v_{final} = v_i + u_i$$

## 7 Summary: GloVe vs. Word2Vec

Feature	GloVe	Word2Vec (e.g., Skip-Gram)
Type	Primarily Count-based / Log-bilinear Regression	Predictive / Neural Network
Input Data	Global Co-occurrence Matrix ( $X$ )	Raw Corpus (Local Windows)
Objective	Weighted Least Squares Regression (Minimize error against $\log(X_{ij})$ )	Binary Classification (Negative Sampling) or Softmax
Utilization of Stats	Explicitly uses global statistics	Implicitly learns from local contexts
Training Speed	Generally faster (Iterates over $ X_{nnz} $ )	Slower (Iterates over corpus size)
Performance	Often excels at analogy tasks and global semantics	Strong at capturing local syntax and complex patterns

## 8 Numerical Example: A Single Training Step

Let's walk through a simplified example of updating vectors using the GloVe objective and gradient descent.

### Numerical Walkthrough: Minimizing the Co-occurrence Error

#### Scenario:

- We focus on a single pair:  $i = \text{cat}, j = \text{milk}$ .
- Embedding Dimension (D): 2.
- Learning Rate ( $\eta$ ): 0.05.
- Simplified Weighting: Assume  $f(X_{ij}) = 0.8$  for this pair.
- Ground Truth: Assume  $\log(X_{ij}) = 3.0$ .

#### 1. Initial Vectors and Biases (Randomized):

- $v_{\text{cat}} = [0.5, 0.2]$
- $u_{\text{milk}} = [0.1, 0.9]$
- $b_{\text{cat}} = 0.1$
- $\tilde{b}_{\text{milk}} = 0.3$

**2. Forward Pass (Prediction):** Calculate the model's current estimate for the log co-occurrence.

$$\text{Pred} = v_{\text{cat}}^T u_{\text{milk}} + b_{\text{cat}} + \tilde{b}_{\text{milk}}$$

$$\text{Dot Product} = (0.5 \times 0.1) + (0.2 \times 0.9) = 0.05 + 0.18 = 0.23$$

$$\text{Pred} = 0.23 + 0.1 + 0.3 = \mathbf{0.63}$$

**3. Calculate the Error (Residual):** The difference between the prediction and the ground truth.

$$\text{Error} = \text{Pred} - \log(X_{ij}) = 0.63 - 3.0 = \mathbf{-2.37}$$

The prediction is much lower than the actual count, so the vectors need to be pulled closer together.

#### 4. Calculate the Loss Contribution (for this pair):

$$J_{ij} = f(X_{ij}) \cdot (\text{Error})^2$$

$$J_{ij} = 0.8 \times (-2.37)^2 = 0.8 \times 5.6169 \approx \mathbf{4.49}$$

**5. Gradient Descent: Calculating the Update for  $v_{\text{cat}}$ :** We need the gradient of  $J_{ij}$  with respect to  $v_{\text{cat}}$ . We use the chain rule.

$$\frac{\partial J_{ij}}{\partial v_{\text{cat}}} = \frac{\partial J_{ij}}{\partial \text{Pred}} \cdot \frac{\partial \text{Pred}}{\partial v_{\text{cat}}}$$

$$\frac{\partial J_{ij}}{\partial \text{Pred}} = 2 \cdot f(X_{ij}) \cdot (\text{Error})$$

$$\frac{\partial \text{Pred}}{\partial v_{\text{cat}}} = u_{\text{milk}}$$

$$\nabla v_{cat} = 2 \cdot (0.8) \cdot (-2.37) \cdot u_{milk}$$

$$\nabla v_{cat} = -3.792 \cdot [0.1, 0.9] = [-0.3792, -3.4128]$$

**6. Update  $v_{cat}$  (Gradient Descent):** We update the vector by moving opposite to the gradient.

$$v_{cat}^{new} = v_{cat}^{old} - \eta \cdot \nabla v_{cat}$$

$$v_{cat}^{new} = [0.5, 0.2] - 0.05 \cdot [-0.3792, -3.4128]$$

$$v_{cat}^{new} = [0.5, 0.2] - [-0.01896, -0.17064]$$

$$v_{cat}^{new} = [0.51896, 0.37064]$$

**7. Update  $u_{milk}$  (Symmetric):** The gradient calculation is symmetric, replacing  $u_{milk}$  with  $v_{cat}$  in the derivative.

$$\nabla u_{milk} = 2 \cdot f(X_{ij}) \cdot (\text{Error}) \cdot v_{cat}$$

$$\nabla u_{milk} = -3.792 \cdot v_{cat} = -3.792 \cdot [0.5, 0.2] = [-1.896, -0.7584]$$

$$u_{milk}^{new} = [0.1, 0.9] - 0.05 \cdot [-1.896, -0.7584]$$

$$u_{milk}^{new} = [0.1948, 0.93792]$$

**8. Update Biases (Similar process):** The derivative of the prediction w.r.t a bias is 1.

$$\nabla b_{cat} = 2 \cdot f(X_{ij}) \cdot (\text{Error}) \cdot 1 = -3.792$$

$$b_{cat}^{new} = 0.1 - 0.05 \cdot (-3.792) = 0.2896$$

(And similarly for  $\tilde{b}_{milk}$ ).

**Result Analysis:** The vectors  $v_{cat}$  and  $u_{milk}$  have moved significantly closer together (their dot product will now be higher), and the biases have increased. The next time this pair is processed, the prediction will be closer to the target  $\log(X_{ij}) = 3.0$ .

## 9 Conclusion

GloVe provides a powerful and elegant approach to learning word embeddings by focusing directly on global corpus statistics. Its key innovation lies in recognizing that ratios of co-occurrence probabilities, rather than raw counts or local predictions, are the optimal signal for encoding semantic relationships. By formulating the problem as a weighted least squares objective, GloVe efficiently generates high-quality vector spaces that exhibit the linear substructures necessary for advanced NLP tasks, effectively bridging the gap between statistical count-based methods and predictive neural network approaches.