Week 2: Word2Vec Deep Dive

Understanding CBOW, Skip-gram, and Negative Sampling
Pre-Lab Exercise (No Programming Required)
INSTRUCTOR VERSION WITH ANSWER KEY

NLP Course 2025

Time: 30-40 minutes
Objective: Master the core mechanisms of Word2Vec by understanding what goes in (context), how it's processed (method), and what comes out (prediction).

Part 1: Understanding Context Windows (10 minutes)

Context Window Discovery

Consider this sentence: **"The quick brown fox jumps over the lazy dog"**
With a context window size of 2 (two words on each side), let's analyze the word "fox":
Questions:

1. Circle the context words for "fox" (window size = 2):

   `The` *`quick`* *`brown`* `fox` *`jumps`* *`over`* `the` `lazy` `dog`

2. What are the context words? *quick, brown, jumps, over*

3. For the word "jumps", what would be the context words (window size = 2)?

   Context words: *brown, fox, over, the*

4. **Key Question:** In a context window approach:

   - What is the **CONTEXT**? *The surrounding words within the window*
   - What is the **METHOD**? *Sliding window across the text*
   - What is the **PREDICTION**? *Word relationships based on co-occurrence*

Part 2: CBOW - Continuous Bag of Words (10 minutes)

CBOW Principle

CBOW predicts a target word given its surrounding context words. Think of it as a "fill in the blank" task.

CBOW in Action

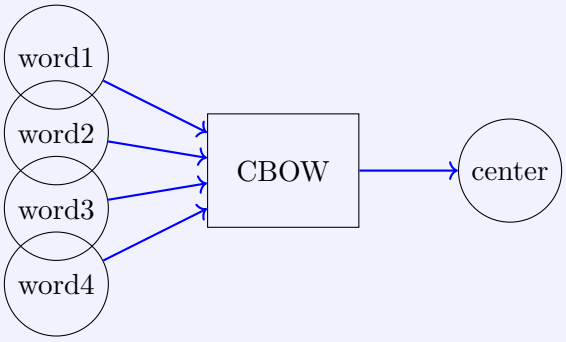
Given the sentence: "The cat sat on the _____ mat"
Task 1: Manual CBOW

1. Context words given: ["cat", "sat", "on", "the", "mat"]
   What word would you predict for the blank? *floor, ground, soft, old*
2. Now consider: "I love to _____ pizza for dinner"
   Context words: ["I", "love", "to", "pizza"]
   Your prediction: *eat, have, order*

Task 2: CBOW Analysis Complete the following table for CBOW:

Aspect	**Your Answer**
INPUT (Context)	*Multiple context words (surrounding words)*
METHOD (How does it work?)	*Average or sum the context word vectors, then predict the center word*
OUTPUT/PREDICTION	*Single center/target word*

Visual Representation: Draw arrows showing the flow:



Many-to-One prediction

Part 3: Skip-gram - Predicting Context (10 minutes)

Skip-gram Principle

Skip-gram does the opposite of CBOW: given a center word, predict the surrounding context words.

Skip-gram in Action

Given the center word "**coffee**" in: "I drink coffee every morning"
Task 1: Manual Skip-gram

1. What context words would you expect around "coffee"? List 4 likely words:
   - *drink*
   - *morning*
   - *cup*
   - *hot*
2. Given center word "**king**", what context words might appear?
   - *queen, throne, crown, royal, palace*
   - *england, arthur, kingdom*

Task 2: Skip-gram Analysis Complete the following table for Skip-gram:

Aspect	**Your Answer**
INPUT (Context)	*Single center/target word*
METHOD (How does it work?)	*Use center word vector to predict each context word independently*
OUTPUT/PREDICTION	*Multiple context words (one at a time)*

Task 3: CBOW vs Skip-gram

1. Which approach has MORE training examples from the same sentence?
   Skip-gram
2. Why? *Skip-gram creates multiple training pairs (center, context1), (center, context2), etc., while CBOW creates one pair (all-context, center)*

Part 4: Negative Sampling - Making Training Efficient (10 minutes)

Why Negative Sampling?

Training Word2Vec on large vocabularies (e.g., 50,000 words) is computationally expensive. Negative sampling speeds this up by only updating a small subset of weights.

Understanding Negative Sampling

Task 1: Positive vs Negative Pairs
Given the sentence: "The dog barked loudly"
With center word "dog" and context word "barked":

1. This is a *positive* pair because *they actually appear together in context*
2. Now consider "dog" and "elephant". This is a *negative* pair because *they don't appear together in this context*
3. Create 3 negative pairs for center word "dog":
   - (dog, *computer*)
   - (dog, *galaxy*)
   - (dog, *mathematics*)

Task 2: The Sampling Process
Instead of updating all 50,000 word weights, negative sampling:

1. Updates the positive pair: (dog, barked) → predict "1" (real pair)
2. Randomly samples k negative words (e.g., k=5)
3. Updates these negative pairs: (dog, random_word) → predict "0" (fake pair)

Complete this table for Negative Sampling:

Aspect	**Your Answer**
INPUT (Context)	*Word pairs: 1 positive (real) pair and k negative (random) pairs*
METHOD (How does it work?)	*Binary classification: predict 1 for positive pairs, 0 for negative pairs. Update only k+1 weights*
OUTPUT/PREDICTION	*Probability that a pair is real (0 or 1)*

Task 3: Efficiency Analysis

1. Without negative sampling, how many weights update for each training example?
   All vocabulary weights (e.g., 50,000)
2. With negative sampling (k=5), how many weights update?
   Only 6 weights (1 positive + 5 negative)
3. What is the speedup factor for a 50,000-word vocabulary?
   50,000 / 6 8,333x faster

Hint

Negative sampling converts the problem from multi-class classification (50,000 classes) to binary classification (real vs fake pairs).

Part 5: Putting It All Together (5 minutes)

Synthesis

Complete this comparison table:

Aspect	CBOW	Skip-gram	Negative Sampling
Input	Context words	*Center word*	*Word pairs*
Method	Average/combine context	*Predict each context word*	Binary classification
Output	*Center word*	Context words	*Real/fake (0/1)*
Best for	Frequent words	*Rare words*	*Efficiency*
Training Speed	Faster	*Slower but better quality*	Makes both faster

Reflection Questions:

1. Why might Skip-gram work better for rare words than CBOW?

   Skip-gram generates more training examples per word occurrence. For a rare word appearing once with 4 context words, Skip-gram creates 4 training pairs while CBOW creates just 1. This gives Skip-gram more opportunities to learn about rare words.

2. How does negative sampling change what the model is learning to predict?

   Without negative sampling: predict which specific word (out of 50,000) comes next. With negative sampling: predict whether a word pair is real (1) or fake (0) - binary classification instead of 50,000-way classification.

3. If you wanted to find similar words to "doctor", which approach (CBOW or Skip-gram) would likely give better results? Why?

   Skip-gram would likely give better results because it directly learns what context words appear around "doctor", capturing more nuanced relationships. CBOW averages context, potentially losing fine-grained distinctions that make "doctor" unique from related words like "nurse" or "physician".

Instructor Notes

Key Teaching Points

1. **Context-Method-Output Framework**: Consistently emphasize this trilogy:
   - CBOW: Multiple context → Average → Single word
   - Skip-gram: Single word → Separate predictions → Multiple context
   - Negative Sampling: Word pairs → Binary classification → Real/fake
2. **Intuitive Understanding**:
   - CBOW = Fill in the blank (convergent thinking)
   - Skip-gram = Word association (divergent thinking)

- Negative Sampling = Real or fake game

3. **Common Misconceptions**:
   - Students often confuse input/output for CBOW vs Skip-gram
   - Emphasize that negative sampling is an optimization technique, not a separate model
   - Clarify that "bag of words" in CBOW means order doesn't matter
4. **Computational Insights**:
   - Without negative sampling: O(V) operations per training example
   - With negative sampling: O(k) operations, where k ¡¡ V
   - This makes training on billions of words feasible

Extension Questions

For advanced students:

1. How would you handle words that appear in multiple senses (e.g., "bank")?
2. Why do we sample negative words based on frequency raised to 0.75 power?
3. How does subsampling frequent words (like "the", "a") improve embeddings?

Follow-up Programming Lab

After this conceptual introduction, students will be ready to:

- Implement a simple CBOW model in Python
- Visualize Skip-gram training pairs from real text
- Compare embeddings with and without negative sampling
- Explore pre-trained Word2Vec models

End of Instructor Version