NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

Word Sense Disambiguation

In natural language, many words are **ambiguous**, meaning they have multiple possible meanings (senses). Humans easily resolve this ambiguity using context, but computers need explicit methods. **Word Sense Disambiguation (WSD)** is the computational task of: Selecting the **correct sense of a word** given its surrounding context.

Why WSD is important?

WSD plays a crucial role in:

- Machine Translation → correct translation depends on sense
- Information Retrieval → improves search relevance
- Question Answering → ensures correct interpretation
- Text Understanding → avoids semantic confusion

Mathematical Representation of WSD

Let:

w=Ambiguous word

s_i=Possible Senses

c=context

The goal is: $\hat{s}=\arg\max_{s_i} P(s_i|c)$

We choose the sense s_i that has the highest probability given the context

This is the core idea behind almost all the WSD methods

Supervised WSD

If we have data which has been hand-labeled with correct word senses, we can use a **supervised learning** approach to the problem of sense disambiguation. Extracting features from the text that are helpful in predicting particular senses, and then training a classifier to assign the correct sense given these features. The output of training is thus a classifier system capable of assigning sense labels to unlabeled words in context. For **lexical sample** tasks, there are various labeled corpora for individual words, consisting of context sentences labeled with the correct sense for the target word.

NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

Extracting Feature Vectors for WSD

To extract useful features from a text window, a minimal amount of processing is first performed on the sentence containing the window. This processing varies from approach to approach but typically includes part-of-speech tagging, lemmatization or stemming, and in some cases syntactic parsing to reveal information such as head words and dependency relations. Context features relevant to the target word can then be extracted from this enriched input. A **feature vector** consisting of numeric or nominal values is used to encode this linguistic information as an input to most machine learning algorithms.

Two classes of features are generally extracted from these neighboring contexts: collocational features and bag-of-words features. A **collocation** is a word or phrase in a position-specific relationship to a target word (i.e., exactly one word to the right, or exactly 4 words to the left, and so on). Thus **collocational features** encode information about *specific* positions located to the left or right of the target word. Typical features extracted for these context words include the word itself, the root form of the word, and the word's part-of-speech. Such features are effective at encoding local lexical and grammatical information that can often accurately isolate a given sense.

Example:

An electric guitar and **bass** player stand off to one side, not really part of the scene, just as a sort of nod to gringo expectations perhaps.

A collocational feature-vector, extracted from a window of two words to the right and left of the target word, made up of the words themselves and their respective parts-of speech, i.e.,

[w_{i-2},POS$_{i-2}$,w_{i-1},POS$_{i-1}$,w_{i+1}, POS$_{i+1}$,w_{i+2},POS$_{i+2}$]

would yield the following vector:

[guitar, NN, and, CC, player, NN, stand, VB]

For example, selectional restriction violations (like inedible arguments of eat) often occur in well-formed sentences, for example because they are negated or because selectional restrictions are overstated.

Examples:

- But it fell apart in 1931, perhaps because people realized you can't eat gold for lunch if you're hungry.
- In his two championship trials, Mr. Kulkarni ate glass on an empty stomach, accompanied only by water and tea.

NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

The second type of feature consists of **bag-of-words** information about neighboring words. A **bag-of-words** means an unordered set of words, ignoring their exact position. The simplest bag-of-words approach represents the context of a target word by a vector of features, each binary feature indicating whether a vocabulary word *w* does or doesn't occur in the context. This vocabulary is typically preselected as some useful subset of words in a training corpus.
In most WSD applications, the context region surrounding the target word is generally a small symmetric fixed size window with the target word at the center. Bag-of-word features are effective at capturing the general topic of the discourse in which the target word has occurred. This, in turn, tends to identify senses of a word that are specific to certain domains.

Example:

For example a bag-of-words vector consisting of the 12 most frequent content words from a collection of *bass* sentences drawn from the WSJ corpus would have the following ordered word feature set:

[*fishing, big, sound, player, fly, rod, pound, double, runs, playing, guitar, band*]

Using these word features with a window size of 10, would be represented by the following binary vector:

[0,0,0,1,0,0,0,0,0,0,1,0]

Naive Bayes for Word Sense Disambiguation

Core Idea

In WSD, we want to determine the **correct sense of a word given its context**.

Naive Bayes treats this as a **probabilistic classification problem**:

Given context features → predict the most likely sense.

NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

Problem Formulation

Let:

- w = ambiguous word
- $s \in \{s_1, s_2, ..., s_n\}$ = possible senses
- c = context (surrounding words/features)

Goal:

$$\hat{s} = \arg\max_s P(s \mid c)$$

Applying Bayes Theorem

Using Bayes rule:

$$P(s \mid c) = \frac{P(c \mid s)\,P(s)}{P(c)}$$

Since $P(c)$ is same for all senses:

$$\hat{s} = \arg\max_s P(c \mid s)\,P(s)$$

Naive Independence Assumption

Context consists of features:

$$c = \{f_1, f_2, ..., f_k\}$$

Naive Bayes assumes:

> Features are conditionally independent given the sense.

$$P(c \mid s) = \prod_k P(f_k \mid s)$$

NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

Final Naive Bayes Equation

$$\hat{s} = \arg\max_{s} P(s) \prod_{k} P(f_k \mid s)$$

Explanation of Terms

- P(s) → prior probability of sense
- P((fk|s) → likelihood of feature given sense
- Product → combines evidence from all context features

Step-by-Step Algorithm

1. Collect labeled training data
2. Compute:
    - P(s)(sense frequency)
    - P(fk|s)(feature likelihoods)
3. For a new sentence:
    - extract features
    - compute probability for each sense
4. Choose sense with highest probability

Example:

Sentence:

He deposited money in the bank.

Target word:

bank

Possible senses:

- s1: financial institution

- s2: river edge

Training Data

To compute probabilities, we need a **labeled corpus**.

Assume we collected the following training sentences:

Financial Sense (s_1)

1. I deposited money in the bank
2. She withdrew cash from the bank
3. The bank approved the loan
4. He saved money in the bank
5. The bank offers credit cards

Total sentences for s1 = **5**

River Sense (s_2)

1. He sat on the river bank
2. The boat reached the bank
3. Children played near the bank
4. The river bank was muddy
5. They walked along the bank

Total sentences for s2 = **5**

Step 1: Compute Prior Probabilities

$$P(s) = \frac{\text{Number of occurrences of sense}}{\text{Total occurrences}}$$

Total sentences = 10

$$P(s_1) = \frac{5}{10} = 0.5$$

$$P(s_2) = \frac{5}{10} = 0.5$$

Step 2: Extract Features

From test sentence:

deposit, money

We now compute:

P(deposit|s),P(money|s)

Step 3: Count Word Frequencies

All words (simplified count):

Word	Count
deposit	1
money	2
bank	5
cash	1
loan	1
credit	1

Total words ≈ **11**

For River Sense (s_2)

Word	Count
river	2
bank	5
boat	1
muddy	1
played	1

Total words ≈ **10**

Step 4: Compute Likelihoods

Formula

$$P(word|s) = \frac{\text{Count(word in sense s)}}{\text{Total words in sense s}}$$

For Financial Sense

$$P(deposit|s_1) = \frac{1}{11} \approx 0.09$$

$$P(money|s_1) = \frac{2}{11} \approx 0.18$$

For River Sense

$$P(deposit|s_2) = \frac{0}{10} = 0$$

$$P(money|s_2) = \frac{0}{10} = 0$$

Step 5: Apply Laplace Smoothing

To avoid zero probability:

$$P(word|s) = \frac{\text{count} + 1}{\text{total words} + V}$$

Where:

- V = vocabulary size (assume $V = 20$)

Updated Probabilities

Financial

$$P(deposit|s_1) = \frac{1+1}{11+20} = \frac{2}{31} \approx 0.064$$

$$P(money|s_1) = \frac{2+1}{11+20} = \frac{3}{31} \approx 0.097$$

River

$$P(deposit|s_2) = \frac{0+1}{10+20} = \frac{1}{30} \approx 0.033$$

$$P(money|s_2) = \frac{0+1}{10+20} = \frac{1}{30} \approx 0.033$$

Step 6: Compute Final Scores

Formula

$$Score(s) = P(s) \times \prod P(feature|s)$$

For Financial Sense

$$Score(s_1) = 0.5 \times 0.064 \times 0.097$$

$$= 0.5 \times 0.0062 \approx 0.0031$$

For River Sense

$$Score(s_2) = 0.5 \times 0.033 \times 0.033$$

$$= 0.5 \times 0.0011 \approx 0.00055$$

Step 7: Compare Results

0.0031>0.00055

Final Answer: bank = financial institution

Decision List Classifier (WSD)

Definition

A **Decision List Classifier** is a rule-based supervised method that:

Learns a list of **if–then rules (features → senses)** ranked by their **strength (log-likelihood score)**.

During testing:

- The **highest-ranked matching rule** decides the sense.

Core Idea

Instead of combining probabilities (like Naive Bayes), we:

1. Extract **features (context patterns)**
2. Compute how strongly each feature predicts a sense
3. Rank rules
4. Apply the **best matching rule**

Training Data (Same Example)

We reuse the dataset:

Financial Sense (s_1)

1. deposit money in the bank
2. withdraw cash from the bank
3. bank approved loan
4. save money in bank
5. credit bank account

River Sense (s_2)

1. sat on river bank
2. boat reached bank
3. near river bank
4. river bank muddy
5. walked along bank

Step 1: Extract Features

We define features like:

Feature	Meaning
deposit	word appears near bank
money	context word
river	context word
boat	context word

Step 2: Count Feature Occurrences

We count how often each feature appears with each sense.

Example Counts

Feature	Count(s_1)	Count(s_2)
deposit	1	0
money	2	0
river	0	2
boat	0	1

Step 3: Compute Probabilities

Formula

$$P(s \mid f) = \frac{\text{Count of feature with sense}}{\text{Total count of feature}}$$

Feature: deposit

$$P(s_1|deposit) = \frac{1}{1} = 1$$

$$P(s_2|deposit) = \frac{0}{1} = 0$$

Feature: money

$$P(s_1|money) = \frac{2}{2} = 1$$

$$P(s_2|money) = 0$$

Feature: river

$$P(s_2|river) = \frac{2}{2} = 1$$

Feature: boat

$$P(s_2|boat) = \frac{1}{1} = 1$$

Step 4: Compute Log-Likelihood Score

Formula

$$Score(f) = \log_2 \frac{P(s_{best}|f)}{P(s_{other}|f)}$$

Handling Zero (Smoothing)

If probability = 0 → use small value (e.g., 0.01)

Feature: deposit

$$Score = \log_2 \frac{1}{0.01} = \log_2(100) \approx 6.64$$

Feature: money

$$Score = \log_2 \frac{1}{0.01} \approx 6.64$$

Feature: river

$$Score = \log_2 \frac{1}{0.01} \approx 6.64$$

Feature: boat

$$Score = \log_2 \frac{1}{0.01} \approx 6.64$$

Step 5: Build Decision List

We sort rules by score:

Decision List

1. IF "deposit" → financial (score 6.64)
2. IF "money" → financial (score 6.64)
3. IF "river" → river sense (score 6.64)
4. IF "boat" → river sense (score 6.64)

Step 6: Testing

Sentence

He deposited money in the bank

Features:

deposit, money

Apply Decision List

- First matching rule = **deposit**
- Highest-ranked rule applies

Prediction: bank = financial institution

Dictionary-Based WSD (Lesk Algorithm)

Definition

The **Dictionary-Based WSD method** (Lesk Algorithm) selects the correct sense of a word by:

Choosing the sense whose **dictionary definition (gloss)** has the **maximum overlap with the context words**.

Core Idea

- Each sense has a **gloss (definition)**
- Compare gloss words with **context words**
- Count how many words overlap
- Select sense with **maximum overlap**

Mathematical Representation

Basic Formula

Basic Formula

$$\hat{s} = \arg\max_{s_i} |\, gloss(s_i) \cap context \,|$$

Explanation

- $gloss(s_i)$ → words in dictionary definition
- $context$ → words in sentence
- $|\cdot|$ → number of overlapping words

Step-by-Step Example

Sentence

He sat on the bank of the river

Target word:

bank

Step 1: Extract Context Words

Remove stopwords (he, on, the, of):

sat, bank, river

Step 2: Get Dictionary Glosses

Sense 1: Financial Bank

bank: financial institution where money is deposited, withdrawn, loan services

Gloss words:

financial, institution, money, deposit, withdraw, loan

Sense 2: River Bank

bank: land along the side of a river or water body

Gloss words:

land, side, river, water

Step 3: Compute Overlap

For Financial Sense

Context:

sat, river

Gloss:

financial, institution, money, deposit

Overlap:

(no common words)

Overlap=0

For River Sense

Context:

sat, river

Gloss:

land, river, water

Overlap:

river

Overlap=1

Step 4: Compare

Sense	**Overlap Score**
Financial	0
River	1

Final Answer: bank = river edge

Bootstrapping Method (Semi-Supervised WSD)

Definition

The **Bootstrapping method** (often associated with **Yarowsky algorithm**) is a **semi-supervised approach** that:

Starts with a **small set of labeled examples (seeds)** and automatically learns patterns to label more data iteratively.

Core Idea

Instead of requiring a fully labeled dataset:

1. Start with a few **seed examples**
2. Learn **strong contextual features**
3. Use them to label new data
4. Expand training data
5. Repeat until stable

Key Principles

1. One Sense per Collocation

A word tends to have the **same sense in a given context pattern**

Example:

deposit money → financial sense

2. One Sense per Discourse

A word usually has the **same sense within a document**

Mathematical Representation

Classification Rule

$$\hat{s} = \arg\max_{s} P(s \mid features)$$

Iterative Update

$$D^{(t+1)} = D^{(t)} \cup \text{new labeled examples}$$

Feature Strength (similar to Decision List)

$$Score(f) = \log \frac{P(s_1 \mid f)}{P(s_2 \mid f)}$$

Step-by-Step Example

Target Word

bank

Senses:

- s1: financial
- s2: river

Step 1: Initial Seed Examples

We manually label a few examples:

Financial Seeds

deposit money in bank

loan from bank

River Seeds

river bank

sat near bank

Step 2: Extract Features

From seeds:

Financial Features

deposit, money, loan

River Features

river, sat, near

Step 3: Count Feature Occurrences

Feature	Count(s_1)	Count(s_2)
deposit	1	0
money	1	0
loan	1	0
river	0	1
sat	0	1

Step 4: Compute Probabilities

Formula

Ask ChatGPT

$$) = \frac{\text{Count of feature with sense}}{\text{Total count of feature}}$$

Example Calculations

Feature: deposit

$$P(s_1|deposit) = \frac{1}{1} = 1$$

$$P(s_2|deposit) = 0$$

Feature: river

$$P(s_2|river) = \frac{1}{1} = 1$$

Step 5: Compute Feature Strength

Using log ratio:

deposit

$$Score = \log \frac{1}{0.01} \approx 4.6$$

river

$$Score = \log \frac{1}{0.01} \approx 4.6$$

Step 6: Apply to Unlabeled Data

New Sentences

1. He deposited cash in the bank
2. The river bank was muddy
3. She took a loan from the bank

4. Children sat on the bank

Labeling

Sentence 1

deposit → financial

Assign financial

Sentence 2

river → river sense

Assign river

Sentence 3

loan → financial

Assign financial

Sentence 4

sat → river

Assign river

Step 7: Update Dataset

Now we add newly labeled data:

$$D^{(t+1)} = D^{(t)} + \text{new examples}$$

Step 8: Recompute Probabilities

Counts increase → better estimates

Example:

Feature	Count(s_1)	Count(s_2)
deposit	2	0
loan	2	0
river	2	0
sat	2	0

Step 9: Repeat Iteration

- Extract new features
- recompute probabilities
- label more data

Continue until no new changes

Final Decision

For any new sentence:

He deposited money in the bank

Feature:

deposit

Choose financial sense

NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

Thesaurus-Based WSD

Definition

The **Thesaurus-Based Method** uses a **lexical resource (like WordNet)** to:

Select the correct sense of a word by measuring **semantic similarity between the candidate senses and the context words**.

Core Idea

- Each sense is represented in a **semantic network (hierarchy)**
- Context words are also mapped to meanings
- Compute **semantic similarity**
- Choose sense with **highest similarity score**

Mathematical Representation

Basic Formula

$$\hat{s} = \arg\max_{s_i} \sum_{w \in context} sim(s_i, w)$$

Explanation

- s_i → candidate sense
- w → context word
- $sim(s_i, w)$ → semantic similarity
- Sum → total compatibility with context

NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

Types of Similarity Measures

4.1 Path-Based Similarity

$$sim = \frac{1}{1 + \text{distance}}$$

- distance = number of edges in hierarchy
- smaller distance → higher similarity

4.2 Depth-Based Similarity

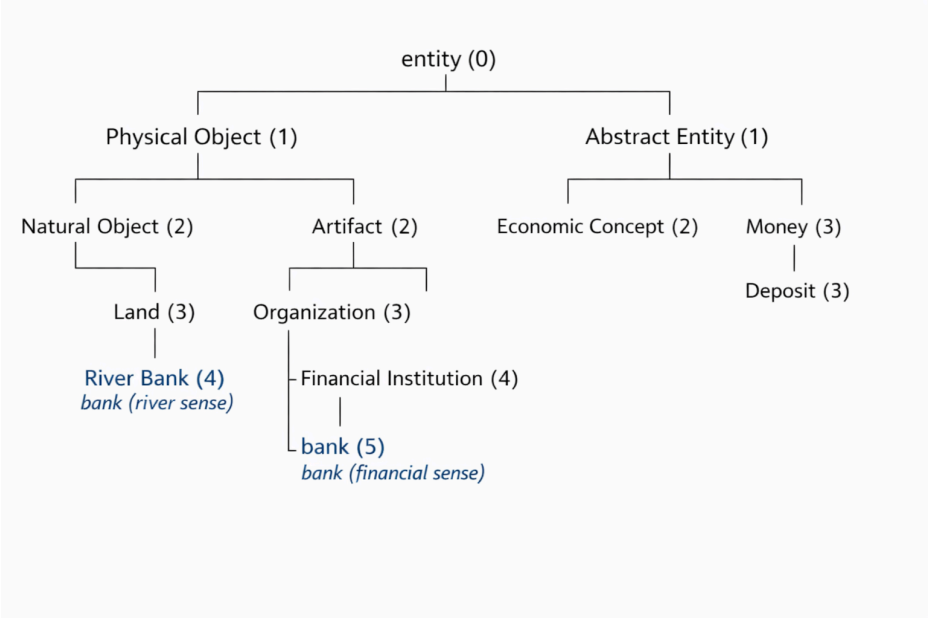
$$sim = \frac{2 \times depth(LCS)}{depth(s_1) + depth(s_2)}$$

- LCS = Least Common Subsumer (common ancestor)

4.3 Information Content (IC) (Advanced)

$$sim = IC(LCS)$$

Step By Step Example



Example: He deposited money in the bank

We compute similarity between:

- **bank (financial sense)**
- **bank (river sense)**

  with context words:

- **money**
- **deposit**

Reference from Your Diagram (Depth Values)

From your tree:

Concept	Depth
entity	0
physical object	1
abstract entity	1
natural object	2
artifact	2
economic concept	2
land	3
organization	3
money	3
deposit	3
river bank	4
financial institution	4
bank (financial)	5

NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

PATH-BASED SIMILARITY

Formula

$$sim(c_1, c_2) = \frac{1}{1 + distance(c_1, c_2)}$$

---

bank (financial) vs money

Step 1: Path

bank → financial institution → organization → artifact → physical object → entity → abstract entity → economic concept → money

Step 2: Count Edges

Step	Edge
1	bank → financial institution
2	→ organization
3	→ artifact
4	→ physical object
5	→ entity
6	→ abstract entity
7	→ economic concept
8	→ money

$$distance = 8$$

Step 3: Similarity

$$sim = \frac{1}{1+8} = \frac{1}{9} \approx 0.11$$

bank (river) vs money

Path

river bank → land → natural object → physical object → entity → abstract entity → economic concept → money

Distance

distance=7

Similarity

sim=1/8=0.125

bank (financial) vs deposit

Path

bank → financial institution → organization → artifact → physical object → entity → abstract entity → economic concept → deposit

Distance

Distance=8

Similarity

Sim=1/9=0.11

bank (river) vs deposit

Path

river bank → land → natural object → physical object → entity → abstract entity → economic concept → deposit

Distance

distance=7

Similarity

sim=1/8=0.125

Path-Based Observation

Sense	vs money	vs deposit	Total
Financial	0.11	0.11	0.22
River	0.125	0.125	0.25

Path method slightly favors **river** (due to structure limitation)

DEPTH-BASED SIMILARITY (Wu-Palmer)

Formula

$$sim(c_1, c_2) = \frac{2 \times depth(LCS)}{depth(c_1) + depth(c_2)}$$

bank (financial) vs money

Step 1: LCS

economic concept

Depth = 2

Step 2: Apply Formula

$$sim = \frac{2 \times 2}{5 + 3} = \frac{4}{8} = 0.5$$

bank (river) vs money

LCS

entity

Depth = 0

$$sim = \frac{2 \times 0}{4 + 3} = 0$$

bank (financial) vs deposit

Step 1: LCS

economic concept

Depth = 2

$$sim = \frac{4}{5+3} = 0.5$$

bank (river) vs deposit

Step 1: LCS

LCS

entity

Depth = 0

$$sim = \frac{2 \times 0}{4+3} = 0$$

Depth-Based Results

Sense	**vs money**	**vs deposit**	**Total**
Financial	0.5	0.5	**1.0**
River	0	0	**0**

Final Answer:

bank = financial institution

Distributional Methods for WSD

NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

Definition

Distributional methods are based on the **Distributional Hypothesis**:

Words that occur in similar contexts have similar meanings.

In WSD:

The correct sense of a word is the one whose **context vector is most similar to the context of the sentence**.

Core Idea

1. Represent words/senses as **vectors**
2. Represent context as a **vector**
3. Compute **similarity (cosine similarity)**
4. Choose sense with **highest similarity**

Mathematical Representation

Cosine Similarity Formula

$$sim(A, B) = \frac{A \cdot B}{|A| \times |B|}$$

Explanation

- $A \cdot B$ → dot product
- $|A|$ → magnitude of vector
- Output → similarity between 0 and 1

General Formula for WSD

$$\hat{s} = \arg\max_{s_i} sim(vector(s_i), vector(context))$$

Step-by-Step Example

Sentence

He deposited money in the bank

Target word:

bank

NLP UNIT-4: Word Sense Disambiguation, WSD using Supervised, Dictionary & Thesaurus, Bootstrapping methods – Word Similarity using Thesaurus and Distributional methods

Possible Senses

- s1: financial
- s2: river

Step 1: Extract Context Words

deposit, money

Step 2: Build Vectors

We create a vocabulary:

deposit, money, river, water

Context Vector
[deposit=1, money=1, river=0, water=0]

Sense Vectors

Financial Sense (bank$_1$)
[deposit=1, money=1, river=0, water=0]

River Sense (bank$_2$)
[deposit=0, money=0, river=1, water=1]

Step 3: Compute Cosine Similarity

For Financial Sense

Dot Product

$$(1 \times 1) + (1 \times 1) + (0 \times 0) + (0 \times 0) = 2$$

Magnitudes

$$|context| = \sqrt{1^2 + 1^2} = \sqrt{2}$$

$$|bank_1| = \sqrt{1^2 + 1^2} = \sqrt{2}$$

Cosine Similarity

$$sim = \frac{2}{\sqrt{2} \times \sqrt{2}} = \frac{2}{2} = 1$$

For River Sense

Dot Product

$$(1 \times 0) + (1 \times 0) + (0 \times 1) + (0 \times 1) = 0$$

Cosine Similarity

$$sim = 0$$

Step 4: Compare

Sense	**Similarity**
Financial	1.0
River	0

Final Answer: bank = financial institution