COP509

Natural Language Processing

The Vector Space Model

Part 3: Cosine Similarity and Ranked Retrieval

Prof. Georgina Cosma

Department of Computer Science
Loughborough University

g.cosma@lboro.ac.uk

Learning Outcomes

By the end of this session, you will be able to:

1. Represent documents and queries as vectors in a high-dimensional space
2. Explain why document length normalisation is necessary
3. Calculate cosine similarity between vectors
4. Apply the complete Vector Space Model for ranked retrieval
5. Understand the strengths and limitations of VSM

Recap from Parts 1 and 2

- Ranked retrieval assigns scores to documents
- TF-IDF weights capture term importance (local and global)
- Documents can be represented as vectors of TF-IDF weights

Mathematical Notation Guide

Before we begin, let's clarify the notation used in these slides:

Symbol	How to Say It	Meaning
$\vec{d}$	"vector d"	An arrow over a letter denotes a **vector** (a list of numbers)
$\hat{d}$	"d hat"	A hat (circumflex) denotes a **unit vector** (length = 1)
$\|\vec{d}\|$	"magnitude of d" or "norm of d"	The **length** of vector $\vec{d}$ (a single number)
$\|V\|$	"size of V" or "cardinality of V"	The **number of elements** in set V
$\vec{d} \cdot \vec{q}$	"d dot q"	The **dot product** of two vectors
d_i	"d sub i"	The i-th element of vector $\vec{d}$
$\sum_{i=1}^{n}$	"sum from i equals 1 to n"	Add up all values from $i = 1$ to n
$\sqrt{x}$	"square root of x"	The number that when squared gives x

Important: Two Different Uses of | | Bars

The vertical bars | | mean different things depending on context:

1. Vector Length (Magnitude/Norm)

When applied to a **vector**, $|\vec{d}|$ means its **length**:

$$|\vec{d}| = \sqrt{d_1^2 + d_2^2 + \ldots + d_n^2}$$

Example: If $\vec{d} = (3, 4)$, then $|\vec{d}| = \sqrt{3^2 + 4^2} = \sqrt{25} = 5$

2. Set Size (Cardinality)

When applied to a **set**, $|V|$ means the **number of elements**:

$$|V| = \text{count of items in } V$$

Example: If $V = \{\text{cat}, \text{dog}, \text{fish}\}$, then $|V| = 3$

How to tell them apart: Look at what's inside. If it's a vector ($\vec{d}$), the bars mean length. If it's a set name (V), the bars mean size.

Understanding Vectors: A Quick Refresher

What is a vector? A vector is simply an ordered list of numbers.

Example: $\vec{v} = (3, 1, 4)$ is a vector with 3 elements (say it as “vector v”)

- $v_1 = 3$ (first element, say “v sub 1”)
- $v_2 = 1$ (second element)
- $v_3 = 4$ (third element)

Vector length (also called **magnitude** or **norm**):
How “long” the vector is (uses Pythagoras in multiple dimensions):

$$|\vec{v}| = \sqrt{v_1^2 + v_2^2 + v_3^2} = \sqrt{3^2 + 1^2 + 4^2} = \sqrt{9 + 1 + 16} = \sqrt{26} \approx 5.1$$

Dot product (also called **inner product** or **scalar product**):
Multiply corresponding elements and add them up. The result is a single number:

$$\vec{a} \cdot \vec{b} = a_1 \times b_1 + a_2 \times b_2 + a_3 \times b_3$$

Example: $(3, 1, 4) \cdot (2, 5, 1) = (3 \times 2) + (1 \times 5) + (4 \times 1) = 6 + 5 + 4 = 15$

Outline for Part 3

1. Documents as Vectors
2. The Problem of Document Length
3. Cosine Similarity
4. The Complete Vector Space Model
5. Worked Example: Full VSM Pipeline
6. Summary and Key Takeaways

Section 1

Documents as Vectors

The Vector Space Model: Core Idea

Key Concept

Represent both **documents** and **queries** as vectors in a high-dimensional space, where each dimension corresponds to a term in the vocabulary.

If our vocabulary has 3 terms: $V = \{\text{cat, dog, fish}\}$

Each document becomes a 3-dimensional vector:

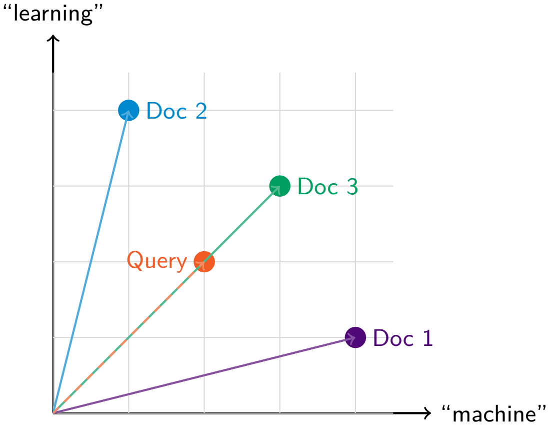
- Doc 1: "The cat sat" → $\vec{d_1} = (1, 0, 0)$ (only "cat" present)
- Doc 2: "Dog and cat" → $\vec{d_2} = (1, 1, 0)$ ("cat" and "dog" present)
- Doc 3: "Fish swim" → $\vec{d_3} = (0, 0, 1)$ (only "fish" present)

In reality:

- Vocabulary size $|V|$ might be 50,000+ terms
- Documents are vectors in 50,000-dimensional space!
- Most entries are 0 (documents don't contain most words)

Visualising the Vector Space

With 2 terms, we can visualise documents as points in 2D:



Intuition: Documents close to the query vector are more similar/relevant.
Doc 3 appears closest to the query, but how do we measure "closeness"?

TF-IDF Weighted Vectors

In practice, we don't use binary (0/1) vectors. We use **TF-IDF weights**:

Each dimension contains the TF-IDF weight of that term in the document:

$$\vec{d} = (w_{t_1,d}, w_{t_2,d}, \ldots, w_{t_{|V|},d})$$

where $w_{t,d} = \text{tf-idf}_{t,d}$

Example: Vocabulary = {algorithm, data, learning, machine}

	algorithm	data	learning	machine
Document 1	2.1	0.0	3.5	1.8
Document 2	0.0	4.2	2.1	0.0
Query	1.0	0.0	2.0	1.5

$\vec{d_1} = (2.1, 0.0, 3.5, 1.8)$, $\vec{d_2} = (0.0, 4.2, 2.1, 0.0)$, $\vec{q} = (1.0, 0.0, 2.0, 1.5)$

Section 2

The Problem of Document Length

Why We Can’t Just Use Dot Product

A natural way to compare vectors is the **dot product**:

$$\vec{d} \cdot \vec{q} = \sum_{i=1}^{|V|} d_i \times q_i$$

Problem: Longer documents have larger vectors!

Short document:
“Machine learning is useful.”

- Few terms
- Small TF-IDF values
- Small vector magnitude

Long document:
A 5000-word article on machine learning

- Many terms
- Large TF-IDF values
- Large vector magnitude

Result: The dot product favours longer documents, even if they’re not more relevant!
We need to **normalise** for document length.

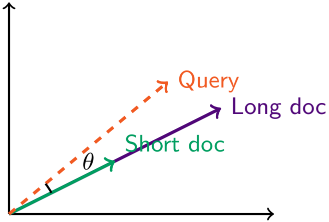
Euclidean Distance vs. Angle

Option 1: Euclidean Distance

$$\text{dist}(\vec{d}, \vec{q}) = \sqrt{\sum_{i=1}^{|V|} (d_i - q_i)^2}$$

Problem: Still affected by vector magnitude (document length).

Option 2: Angle Between Vectors



The short and long documents point in nearly the same direction. They should be equally similar to the query!

Solution: Use the **angle** between vectors, not the distance.

Section 3

Cosine Similarity

Cosine Similarity: The Core Idea

Goal: Measure how similar two vectors are based on their **direction**, ignoring their length.

Why Direction, Not Length?

- A long document about "cats" and a short document about "cats" should be equally similar to a query about "cats"
- We care about **what** the document is about, not **how long** it is
- The **angle** between vectors captures direction similarity

The cosine function:

- $\cos(0^\circ) = 1$: vectors point in the **same** direction (identical content)
- $\cos(90^\circ) = 0$: vectors are **perpendicular** (nothing in common)
- The smaller the angle, the higher the cosine, the more similar the vectors

Cosine Similarity: The Formula

Cosine Similarity Formula

$$\cos(\vec{d}, \vec{q}) = \frac{\vec{d} \cdot \vec{q}}{|\vec{d}| \times |\vec{q}|}$$

Let's break down each part:

$\vec{d} \cdot \vec{q}$	**Dot product** (numerator): Multiply matching elements and sum. Measures raw similarity.
$\|\vec{d}\|$	**Length of document vector**: $\sqrt{d_1^2 + d_2^2 + \ldots + d_n^2}$
$\|\vec{q}\|$	**Length of query vector**: $\sqrt{q_1^2 + q_2^2 + \ldots + q_n^2}$
$\|\vec{d}\| \times \|\vec{q}\|$	**Product of lengths** (denominator): This **normalises** the result

Result: Dividing by the lengths “cancels out” the effect of document size!

The Formula Expanded

Writing out the formula in full:

$$\cos(\vec{d}, \vec{q}) = \frac{\vec{d} \cdot \vec{q}}{|\vec{d}| \times |\vec{q}|} = \frac{\sum_{i=1}^{|V|} d_i \times q_i}{\sqrt{\sum_{i=1}^{|V|} d_i^2} \times \sqrt{\sum_{i=1}^{|V|} q_i^2}}$$

These are the SAME formula, just different ways of writing it:

Compact form	Expanded form
$\vec{d} \cdot \vec{q}$	$\sum_{i=1}^{\vert V\vert} d_i \times q_i = d_1 q_1 + d_2 q_2 + \ldots + d_n q_n$
$\vert\vec{d}\vert$	$\sqrt{\sum_{i=1}^{\vert V\vert} d_i^2} = \sqrt{d_1^2 + d_2^2 + \ldots + d_n^2}$
$\vert\vec{q}\vert$	$\sqrt{\sum_{i=1}^{\vert V\vert} q_i^2} = \sqrt{q_1^2 + q_2^2 + \ldots + q_n^2}$

Note: $|V|$ here means the vocabulary size (number of dimensions/terms).

Unit Vectors and Normalisation Explained

A unit vector (written $\hat{d}$, say "d hat") is a vector with length exactly 1.

To create a unit vector: Divide each element by the vector's length.

$$\hat{d} = \frac{\vec{d}}{|\vec{d}|} = \left(\frac{d_1}{|\vec{d}|}, \frac{d_2}{|\vec{d}|}, \ldots, \frac{d_n}{|\vec{d}|} \right)$$

Worked Example: Convert $\vec{d} = (3, 4)$ to a unit vector

1. Calculate length: $|\vec{d}| = \sqrt{3^2 + 4^2} = \sqrt{9 + 16} = \sqrt{25} = 5$
2. Divide each element by 5: $\hat{d} = \left(\frac{3}{5}, \frac{4}{5}\right) = (0.6, 0.8)$
3. Verify length is 1: $|\hat{d}| = \sqrt{0.6^2 + 0.8^2} = \sqrt{0.36 + 0.64} = \sqrt{1} = 1$ ✓

Key Insight

The unit vector $\hat{d}$ points in the **same direction** as $\vec{d}$, but has length 1. This process is called **normalising** the vector (or **L2 normalisation**).

Two Ways to Calculate Cosine Similarity

There are two equivalent approaches:

Method 1: Use the formula directly

$$\cos(\vec{d}, \vec{q}) = \frac{\vec{d} \cdot \vec{q}}{|\vec{d}| \times |\vec{q}|}$$

Calculate dot product, calculate both lengths, then divide.

Method 2: Normalise first, then dot product

$$\cos(\vec{d}, \vec{q}) = \hat{d} \cdot \hat{q}$$

Convert both vectors to unit vectors first, then just take their dot product.

Why Are These the Same?

$\hat{d} \cdot \hat{q} = \frac{\vec{d}}{|\vec{d}|} \cdot \frac{\vec{q}}{|\vec{q}|} = \frac{\vec{d} \cdot \vec{q}}{|\vec{d}| \times |\vec{q}|}$

The normalisation in Method 2 does the same division as the denominator in Method 1!

Why Have Two Methods?

Both give the same answer, but one may be more efficient:

Method 1: Formula directly

$$\cos(\vec{d}, \vec{q}) = \frac{\vec{d} \cdot \vec{q}}{|\vec{d}| \times |\vec{q}|}$$

- Good for one-off calculations
- No preprocessing needed
- Used in our worked examples

Method 2: Pre-normalise

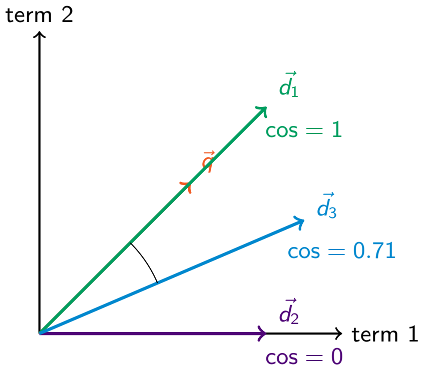
$$\cos(\vec{d}, \vec{q}) = \hat{d} \cdot \hat{q}$$

- Good for search engines
- Normalise documents **once**
- Each query just needs dot products
- Much faster at query time!

In Practice

Search engines pre-normalise all document vectors when indexing. Then for each query, they only need to normalise the query once and compute dot products. No square roots needed at search time!

Cosine Similarity: Visual Intuition



Interpretation:

- $\vec{d}_1$ points in same direction as $\vec{q} \to \cos = 1$ (maximum similarity)
- $\vec{d}_2$ is perpendicular to $\vec{q} \to \cos = 0$ (no similarity)
- $\vec{d}_3$ is at 45° from $\vec{q} \to \cos = 0.71$ (moderate similarity)

Cosine Similarity: Key Properties

Understanding the range and meaning of cosine values:

Cosine Value	**Angle**	**Meaning**
1	0°	Identical direction (most similar)
0.87	30°	Very similar
0.71	45°	Moderately similar
0.5	60°	Somewhat similar
0	90°	Orthogonal (nothing in common)

Important notes for TF-IDF vectors:

- TF-IDF weights are always ≥ 0 (no negative values)
- Therefore, cosine similarity is always in range $[0, 1]$
- Negative cosine values only occur with negative vector elements (not our case)
- Higher cosine $=$ more similar $=$ better match for retrieval

Cosine Similarity: Worked Example

Query: $\vec{q} = (1, 2, 3)$
Document: $\vec{d} = (2, 4, 6)$

Step 1: Calculate dot product

$$\vec{q} \cdot \vec{d} = (1 \times 2) + (2 \times 4) + (3 \times 6) = 2 + 8 + 18 = 28$$

Step 2: Calculate vector lengths

$$|\vec{q}| = \sqrt{1^2 + 2^2 + 3^2} = \sqrt{1 + 4 + 9} = \sqrt{14} \approx 3.74$$

$$|\vec{d}| = \sqrt{2^2 + 4^2 + 6^2} = \sqrt{4 + 16 + 36} = \sqrt{56} \approx 7.48$$

Step 3: Calculate cosine similarity

$$\cos(\vec{q}, \vec{d}) = \frac{28}{3.74 \times 7.48} = \frac{28}{27.98} \approx \mathbf{1.0}$$

Result: Cosine $= 1$ because $\vec{d} = 2 \times \vec{q}$ (same direction, different length)!

Another Worked Example

Query: $\vec{q} = (1, 0, 2)$
Document: $\vec{d} = (2, 1, 1)$

Step 1: Dot product

$$\vec{q} \cdot \vec{d} = (1 \times 2) + (0 \times 1) + (2 \times 1) = 2 + 0 + 2 = 4$$

Step 2: Vector lengths

$$|\vec{q}| = \sqrt{1^2 + 0^2 + 2^2} = \sqrt{5} \approx 2.24$$

$$|\vec{d}| = \sqrt{2^2 + 1^2 + 1^2} = \sqrt{6} \approx 2.45$$

Step 3: Cosine similarity

$$\cos(\vec{q}, \vec{d}) = \frac{4}{2.24 \times 2.45} = \frac{4}{5.49} \approx \mathbf{0.73}$$

Interpretation: Moderate similarity. The vectors share some but not all directions.

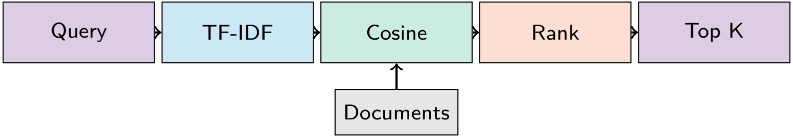
Section 4

The Complete Vector Space Model

Putting It All Together: The VSM Pipeline

Vector Space Model for Ranked Retrieval

1. Represent each document as a TF-IDF weighted vector
2. Represent the query as a TF-IDF weighted vector
3. Calculate cosine similarity between query and each document
4. Rank documents by cosine similarity (highest first)
5. Return the top K documents to the user



Why TF-IDF + Cosine Works

This combination solves all the problems we identified:

Problem	**How VSM Solves It**
Jaccard ignores term frequency	TF component counts occurrences (with log dampening)
All terms treated equally	IDF gives higher weight to rare, informative terms
Long documents favoured	Cosine similarity normalises for document length
No ranking	Cosine scores provide continuous ranking

The Vector Space Model

TF-IDF addresses *what* to measure (term importance)
Cosine similarity addresses *how* to compare (length-normalised)

Section 5

Worked Example

Full VSM Pipeline

Example Setup

Collection: 3 documents, vocabulary of 4 terms

Documents:

- Doc 1: “machine learning algorithms”
- Doc 2: “data science and machine learning”
- Doc 3: “algorithms for data analysis”

Query: “machine learning”

Vocabulary: $V = \{\text{algorithms, data, learning, machine}\}$

Given TF-IDF values:

	algorithms	data	learning	machine
Doc 1	1.5	0.0	1.2	1.0
Doc 2	0.0	0.8	1.2	1.0
Doc 3	1.5	0.8	0.0	0.0
Query	0.0	0.0	1.0	1.0

Step 1: Calculate Cosine for Doc 1

$\vec{q} = (0, 0, 1, 1)$ $\vec{d_1} = (1.5, 0, 1.2, 1)$

Dot product:

$$\vec{q} \cdot \vec{d_1} = (0)(1.5) + (0)(0) + (1)(1.2) + (1)(1) = 0 + 0 + 1.2 + 1 = 2.2$$

Vector lengths:

$$|\vec{q}| = \sqrt{0^2 + 0^2 + 1^2 + 1^2} = \sqrt{2} \approx 1.41$$

$$|\vec{d_1}| = \sqrt{1.5^2 + 0^2 + 1.2^2 + 1^2} = \sqrt{2.25 + 0 + 1.44 + 1} = \sqrt{4.69} \approx 2.17$$

Cosine similarity:

$$\cos(\vec{q}, \vec{d_1}) = \frac{2.2}{1.41 \times 2.17} = \frac{2.2}{3.06} \approx \mathbf{0.72}$$

Step 2: Calculate Cosine for Doc 2

$\vec{q} = (0, 0, 1, 1)$ $\qquad$ $\vec{d_2} = (0, 0.8, 1.2, 1)$

Dot product:

$$\vec{q} \cdot \vec{d_2} = (0)(0) + (0)(0.8) + (1)(1.2) + (1)(1) = 0 + 0 + 1.2 + 1 = 2.2$$

Vector lengths:

$$|\vec{q}| = \sqrt{2} \approx 1.41 \text{ (same as before)}$$

$$|\vec{d_2}| = \sqrt{0^2 + 0.8^2 + 1.2^2 + 1^2} = \sqrt{0 + 0.64 + 1.44 + 1} = \sqrt{3.08} \approx 1.75$$

Cosine similarity:

$$\cos(\vec{q}, \vec{d_2}) = \frac{2.2}{1.41 \times 1.75} = \frac{2.2}{2.47} \approx \mathbf{0.89}$$

Step 3: Calculate Cosine for Doc 3

$\vec{q} = (0, 0, 1, 1)$ $\qquad$ $\vec{d_3} = (1.5, 0.8, 0, 0)$

Dot product:

$$\vec{q} \cdot \vec{d_3} = (0)(1.5) + (0)(0.8) + (1)(0) + (1)(0) = 0 + 0 + 0 + 0 = 0$$

Cosine similarity:

$$\cos(\vec{q}, \vec{d_3}) = \frac{0}{|\vec{q}| \times |\vec{d_3}|} = \mathbf{0}$$

Interpretation: Doc 3 contains neither "machine" nor "learning". It has **zero similarity** with the query. The vectors are orthogonal (perpendicular).

Step 4: Rank Documents

Summary of cosine similarities:

Document	**Cosine Similarity**	**Rank**
Doc 2: “data science and machine learning”	0.89	**1**
Doc 1: “machine learning algorithms”	0.72	**2**
Doc 3: “algorithms for data analysis”	0.00	3

Final ranked results for query “machine learning”:

1. **Doc 2**: “data science and machine learning” (score: 0.89)
2. **Doc 1**: “machine learning algorithms” (score: 0.72)
3. Doc 3: “algorithms for data analysis” (score: 0.00)

Note: Doc 2 ranks higher because it’s a shorter document with the same query term weights. This is cosine normalisation at work!

Section 6

Summary and Key Takeaways

The Complete Vector Space Model

VSM Algorithm Summary

Preprocessing (done once):

1. Build vocabulary from all documents
2. Calculate DF for each term
3. For each document, calculate TF-IDF vector

Query processing (for each query):

1. Convert query to TF-IDF vector
2. Calculate cosine similarity with each document
3. Sort documents by similarity (descending)
4. Return top K results

Strengths and Limitations of VSM

Strengths:

- Simple and intuitive
- Efficient to compute
- Works well in practice
- Partial matching (no exact match required)
- Continuous ranking
- Well-understood mathematically

Limitations:

- Bag of words assumption (no word order)
- No understanding of synonyms
- High dimensionality (sparse vectors)
- Assumes term independence
- No semantic understanding

Modern Alternatives

Neural approaches (Word2Vec, BERT) capture semantics but VSM remains widely used due to its simplicity and interpretability.

Key Formulas: Quick Reference

TF-IDF Weight:

$$w_{t,d} = (1 + \log_{10} \mathrm{tf}_{t,d}) \times \log_{10} \frac{N}{\mathrm{df}_t}$$

Cosine Similarity:

$$\cos(\vec{d}, \vec{q}) = \frac{\vec{d} \cdot \vec{q}}{|\vec{d}| \times |\vec{q}|} = \frac{\sum_i d_i q_i}{\sqrt{\sum_i d_i^2} \times \sqrt{\sum_i q_i^2}}$$

Vector Length (Magnitude):

$$|\vec{v}| = \sqrt{\sum_{i=1}^{|V|} v_i^2}$$

Course Summary: The Journey So Far

What we've covered in the VSM lectures:

1. **Part 1:** From Boolean to ranked retrieval
   - Why we need scoring and ranking
   - Jaccard coefficient and its limitations

2. **Part 2:** Term weighting
   - Term Frequency (TF) and log scaling
   - Inverse Document Frequency (IDF)
   - TF-IDF combining local and global importance

3. **Part 3:** The Vector Space Model
   - Documents and queries as vectors
   - Cosine similarity for length normalisation
   - Complete ranked retrieval pipeline

Questions?

Prof. Georgina Cosma

g.cosma@lboro.ac.uk