-->

Chapter 1: Foundations of Text Representation and Similarity

Covered Material	Value
Chapter	Chapter 1
Related Week	Week 1
Related Slides / Files	week1.md, week1_inperson_slides.md

Index

Brief Content Summary

This chapter establishes the mathematical and conceptual foundations for processing natural language text computationally. It begins by examining why language presents unique challenges for computational systems, encompassing morphology, syntax, semantics, and discourse. The core contribution addresses the transformation of unstructured text into structured numerical representations suitable for machine learning algorithms. Three principal vectorization strategies are examined: binary representation, term frequency (TF), and Term Frequency–Inverse Document Frequency (TF-IDF). The chapter then formalizes three distance and similarity measures—Euclidean distance, Jaccard coefficient, and cosine similarity—that enable quantitative comparison of documents in vector space. Throughout, mathematical definitions are accompanied by worked examples demonstrating practical computation.

Abstract

Natural Language Processing (NLP) and Text Mining require the transformation of human language into mathematical representations amenable to computational analysis. This chapter provides a rigorous treatment of foundational text representation techniques, beginning with the linguistic complexities that render language processing non-trivial—including lexical ambiguity, syntactic structure, and pragmatic context. The Bag-of-Words (BoW) model is introduced as the canonical approach to text vectorization, with formal definitions of binary, term frequency, and TF-IDF weighting schemes. The TF-IDF formulation $w_{t,d} = \ln(\mathrm{tf}(t,d)+1) \cdot \log_{10}(N/\mathrm{df}_t)$ is derived and justified through information-theoretic principles. Subsequently, three similarity and distance measures are formalized: Euclidean distance for absolute positional difference, the Jaccard coefficient for set-based overlap, and cosine similarity for angular proximity in high-dimensional space. Vector normalization via the ℓ_2 norm is presented as essential preprocessing for length-invariant comparisons. Worked examples illustrate each computational procedure. These foundations enable document retrieval, classification, and clustering—core tasks in text mining pipelines.

Keywords

Natural Language Processing (NLP); Text Mining; Bag-of-Words (BoW); Term Frequency (TF); Inverse Document Frequency (IDF); TF-IDF; Vector Space Model; Euclidean Distance; Jaccard Coefficient; Cosine Similarity; ℓ_2 Normalization; Tokenization; Document Similarity

1. Introduction

1.1 Purpose and Scope

This chapter establishes the theoretical and mathematical foundations required for computational text analysis within the domains of Natural Language Processing and Text Mining.

The primary objectives are threefold: (i) to articulate the linguistic challenges inherent to computational language understanding, (ii) to formalize methods for converting text into numerical vector representations, and (iii) to define mathematical functions for measuring similarity and distance between document vectors. These foundations are prerequisite to subsequent topics including text classification, information retrieval, and machine learning on textual data.

1.2 The Challenge of Language for Computation

Language presents computational systems with challenges fundamentally different from those encountered in numerical or structured data processing. Whereas a database record possesses explicit schema and unambiguous semantics, a sentence such as "They saw a kid with a telescope" admits multiple valid interpretations depending on the attachment of the prepositional phrase.

The complexity of language manifests across multiple dimensions:

1. **Representation Complexity**: Text exists as sequences of characters forming words, sentences, and documents. Converting this sequential symbolic structure into fixed-dimensional numerical representations

—required by most machine learning algorithms—necessitates principled abstraction.

2. **Mathematical Interpretation**: Extracting quantitative patterns from inherently qualitative symbolic data requires careful definition of what constitutes meaningful features and how they should be weighted.

3. **Knowledge Inference**: Understanding text often requires world knowledge, cultural context, and reasoning capabilities that extend beyond the surface form of the words themselves.

4. **Understanding Depth**: True comprehension involves resolving ambiguity, tracking referents across sentences, and integrating information from multiple sources.

These challenges motivate the systematic approaches developed throughout this chapter.

2. Background

2.1 Historical Context of Natural Language Processing

The field of Natural Language Processing has undergone substantial paradigm shifts since its inception.

Key Historical Developments:

- **Pre-1970s**: Dominated by Chomskyan linguistics emphasizing innate language faculty. Research focused on hand-crafted grammars and symbolic ontologies rather than data-driven learning.

- **1970s–1980s**: The "AI Winter" emerged following unrealized expectations in machine translation. Funding contraction prompted methodological reassessment.

- **1980s–2010s**: Statistical methods gained prominence. Researchers developed problem-specific algorithms leveraging probabilistic models trained on linguistic corpora.

- **2010s–2020s**: Neural networks achieved dominance in NLP. Deep learning architectures, particularly recurrent and attention-based models, achieved state-of-the-art performance across tasks.

- **2020s–Present**: Transformer architectures and Large Language Models (LLMs) have fundamentally altered the landscape, enabling few-shot learning and emergent capabilities.

Despite advances in end-to-end learning, understanding foundational representations remains essential for model interpretability, debugging, and informed architectural choices.

2.2 Related Disciplines

Several interconnected fields contribute to and benefit from advances in computational language processing:

- **Natural Language Processing (NLP)**: Construction of systems for specific tasks including machine translation, text-to-speech synthesis, summarization, language generation, and question answering. Contemporary approaches predominantly employ machine learning.

- **Computational Linguistics**: Application of computational methods to linguistically-motivated research questions, bridging computer science and theoretical linguistics.
- **Text Mining**: Extraction of structured information and patterns from unstructured textual data, typically for knowledge discovery in large document collections.
- **Linguistics**: Theoretical study of language structure including morphology (word formation), syntax (sentence structure), semantics (meaning), pragmatics (contextual use), and phonetics (sound systems).
- **Psycholinguistics**: Investigation of the cognitive processes underlying language comprehension, production, and acquisition in the human brain.

3. Linguistic Foundations

3.1 Levels of Linguistic Analysis

Understanding language computationally requires appreciation of the multiple levels at which linguistic structure operates. Each level presents distinct challenges for automatic processing.

Surface Form (Words): At the most basic level, language consists of words—sequences of characters delimited by spaces and punctuation in written form.

Morphology: Words exhibit internal structure. The verb "is" represents the third-person singular present tense form of the lemma "be." Morphological analysis recovers these underlying forms and grammatical features:

Surface Form	Lemma	Features
is	be	3sg, present

Part-of-Speech (POS) Tags: Words belong to grammatical categories that constrain their syntactic behavior. For the sentence "This is a simple sentence":

This	is	a	simple	sentence
DT	VBZ	DT	JJ	NN

Where DT denotes determiner, VBZ the third-person singular present verb, JJ an adjective, and NN a noun.

Syntax: Words combine into phrases according to grammatical rules, forming hierarchical constituent structures. A noun phrase (NP) such as "a simple sentence" contains a determiner, adjective, and noun dominated by the NP node.

Semantics: Words and phrases carry meaning. "Simple" in context denotes "having few parts," while "sentence" refers to "a string of words satisfying the grammatical rules of a language."

Discourse: Sentences relate to one another through discourse relations. The sequence "This is a simple sentence. But it is an instructive one." exhibits a CONTRAST relation connecting the two propositions.

3.2 Sources of Ambiguity

Language admits multiple interpretations at every level of analysis, creating fundamental challenges for computational systems.

Lexical Ambiguity: Individual words may have multiple meanings. "Bank" may refer to a financial institution or a river's edge.

Structural Ambiguity: Identical word sequences may receive different syntactic analyses with correspondingly different meanings:

- "They saw a kid with a telescope."
  - Interpretation 1: They used a telescope to see a kid.
  - Interpretation 2: They saw a kid who possessed a telescope.
- "Flying planes can be dangerous."
  - Interpretation 1: The act of flying planes is dangerous.
  - Interpretation 2: Planes that are flying can be dangerous.
- "Time flies like an arrow."
  - Interpretation 1: Time passes quickly (metaphorical).
  - Interpretation 2: A species of flies called "time flies" are fond of arrows (syntactic reanalysis).

3.3 Practical Challenges in Language Understanding

Beyond structural ambiguity, several phenomena complicate practical NLP applications:

Negation: Negative constructions reverse polarity. "This movie is definitely not bad" expresses positive sentiment despite containing the word "bad."

Long-Range Dependencies: Interpretation may require tracking relationships across substantial textual spans. In "Whoever thinks this film is incredibly bad is an idiot," the negative evaluation "bad" is embedded within an insult toward those holding that view—the overall stance toward the film may be positive.

Sarcasm and Irony: Surface meaning may be inverted. "This is the best movie ever, lol" likely expresses negative sentiment through sarcastic exaggeration.

World Knowledge: Comparative statements require external knowledge. Understanding "This movie is only slightly better than Jaws" requires knowing that Jaws received exceptionally high ratings (97% on Rotten Tomatoes, 8/10 on IMDb).

Common Sense Reasoning: Textual entailment and inference require reasoning beyond explicit content. Given "The purchase of Houston-based LexCorp by BMI for $2Bn...,”onecaninferthatBMIacquiredanAmericancompany, butnotthatthepurchasepricewas$ 3.4 Bn.

4. Tokenization

4.1 Definition and Purpose

Tokenization is the process of segmenting continuous text into discrete units called tokens. This transformation constitutes the first step in virtually all text processing pipelines, converting raw character sequences into structured input suitable for subsequent analysis.

Tokens typically correspond to words, but may also include punctuation marks, numbers, or subword units depending on the application and tokenization strategy employed.

4.2 Tokenization Strategies

Several approaches to tokenization exist, each with distinct trade-offs:

Whitespace Tokenization: The simplest approach splits text at whitespace characters. For "the cat sat on the mat," this yields `["the", "cat", "sat", "on", "the", "mat"]`. This approach fails to handle punctuation appropriately.

Punctuation-Aware Tokenization: Extends whitespace tokenization by treating punctuation marks as separate tokens. For "the cat sat on the mat.", the result is `["the", "cat", "sat", "on", "the", "mat", "."]`.

Regular Expression Tokenization: Employs pattern matching for flexible token boundary detection, accommodating contractions, hyphenated words, and other complex cases.

Subword Tokenization: Modern neural approaches (e.g., Byte-Pair Encoding, WordPiece) segment words into smaller units, enabling handling of out-of-vocabulary words and morphological variation.

4.3 Formal Definition

Let s denote a document represented as a character string. A tokenization function τ maps s to an ordered sequence of tokens:

$$\tau : \Sigma^* \to T^*$$

where Σ is the character alphabet and T is the token vocabulary. For a document d, we write:

$$\tau(d) = \langle t_1, t_2, \ldots, t_n \rangle$$

The tokenization preserves sequential order, which may or may not be utilized in subsequent representation steps.

5. Text Vectorization

5.1 The Bag-of-Words Model

The **Bag-of-Words (BoW)** model represents documents as unordered collections of words, discarding sequential information while retaining word occurrence statistics. This representation transforms variable-length text into fixed-dimensional vectors suitable for machine learning algorithms.

The term "bag" emphasizes that word order is ignored—only the presence or frequency of words matters. Two documents with identical word frequencies but different word orderings receive identical representations.

Given a corpus vocabulary $\mathbf{T} = \{t_1, t_2, \ldots, t_M\}$ containing M unique terms, each document is represented as an M-dimensional vector.

5.2 Binary Representation

The simplest vectorization encodes only the presence or absence of terms:

$$\vec{d}_i[j] = \begin{cases} 1 & \text{if } t_j \in d_i \\ 0 & \text{otherwise} \end{cases}$$

Example: Consider the documents:

- d_0: "the cat sat on the mat"
- d_1: "my cat sat on my cat"

The vocabulary is $\mathbf{T} = [\text{cat}, \text{mat}, \text{my}, \text{on}, \text{sat}, \text{the}]$ (alphabetically sorted).

The binary representation yields:

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

where row i corresponds to document d_i and column j to term t_j.

Advantages:

- Compact memory representation
- Well-suited for algorithms operating on binary features (e.g., certain decision tree variants)

Disadvantages:

- Loses information about term importance via frequency
- Cannot distinguish between documents where a term appears once versus many times

5.3 Term Frequency Representation

Term frequency $\text{tf}(t, d)$ counts the number of occurrences of term t in document d:

$$\text{tf}(t, d) = |\{i : \tau(d)_i = t\}|$$

A document vector under term frequency representation is:

$$\vec{d_i} = \langle \text{tf}(t_1, d_i), \text{tf}(t_2, d_i), \ldots, \text{tf}(t_M, d_i) \rangle$$

Example: For the same documents:

$$\mathbf{X}_{\text{tf}} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 2 \\ 2 & 0 & 2 & 1 & 1 & 0 \end{bmatrix}$$

Note that "the" appears twice in d_0 and "cat" and "my" each appear twice in d_1.

5.4 The Term-Document Matrix

The complete collection of document vectors forms the **term-document matrix** (or document-term matrix, depending on orientation convention):

$$\mathbf{X} \in \mathbb{R}^{N \times M}$$

where:

- N = number of documents in the corpus
- M = size of the vocabulary $|\mathbf{T}|$
- $\mathbf{X}_{i,j}$ = representation value for term t_j in document d_i

Formal Notation:

Let $\mathbf{D} = \{d_1, d_2, \ldots, d_N\}$ be a corpus of documents and $\mathbf{T} = \{t_1, t_2, \ldots, t_M\}$ the index terms for $\mathbf{D}$. Each document $d_i \in \mathbf{D}$ can be represented as a vector:

$$\vec{d_i} = \langle f(t_1, d_i), f(t_2, d_i), \ldots, f(t_M, d_i) \rangle$$

where $f(t, d)$ denotes the feature function mapping term-document pairs to real values (binary, tf, or tf-idf).

6. Term Weighting: TF-IDF

6.1 Motivation for Weighting

Raw term frequencies exhibit several deficiencies as document representations:

1. **Frequency Saturation**: The difference between $\text{tf} = 10$ and $\text{tf} = 100$ is less meaningful than the difference between $\text{tf} = 0$ and $\text{tf} = 10$. A word occurring 100 times is not ten times more important than one occurring 10 times.

2. **Document Length Bias**: Longer documents naturally contain more word occurrences, inflating term frequencies without proportional increase in topical relevance.

3. **Common Words Dominate**: Highly frequent words (e.g., "the," "is," "and") occur in virtually all documents, providing little discriminative information yet receiving high frequency counts.

4. **Rare Term Informativeness**: Rare terms that occur in few documents may carry significant discriminative value for identifying document similarity.

Term Frequency–Inverse Document Frequency (TF-IDF) weighting addresses these issues through two complementary transformations.

6.2 Term Frequency with Logarithmic Dampening

To mitigate frequency saturation, the raw term frequency is transformed via the natural logarithm:

$$\mathrm{tf}_{\log}(t, d) = \ln(\mathrm{tf}(t, d) + 1)$$

The addition of 1 (additive smoothing) prevents $\ln(0) = -\infty$ for terms not present in a document.

Information-Theoretic Justification: The logarithmic transformation aligns with information-theoretic principles where the information content of an event with probability p is $-\log(p)$. Repeated occurrences of a term contribute diminishing marginal information.

Example: Consider term frequencies across Wikipedia articles for the term "learning":

Document	tf	$\ln(\mathrm{tf} + 1)$
NLP.wiki	27	3.33
IR.wiki	2	1.10
AI.wiki	46	3.85
ML.wiki	134	4.91
TM.wiki	6	1.95
CV.wiki	10	2.40

The logarithm compresses the range: the ratio between ML (134) and IR (2) is 67:1 in raw frequency but only 4.5:1 in log-transformed values.

6.3 Inverse Document Frequency

Document frequency df_t counts the number of documents containing term t:

$$\mathrm{df}_t = |\{d \in \mathbf{D} : t \in d\}|$$

Inverse Document Frequency (IDF) measures term specificity across the corpus:

$$\mathrm{idf}_t = \log_b \frac{N}{\mathrm{df}_t}$$

where:

- N = total number of documents in the corpus
- df_t = document frequency of term t
- b = logarithm base (typically 10)

Properties of IDF:

1. Terms appearing in all documents: $\mathrm{df}_t = N \Rightarrow \mathrm{idf}_t = \log_b(1) = 0$
2. Terms appearing in one document: $\mathrm{df}_t = 1 \Rightarrow \mathrm{idf}_t = \log_b(N)$ (maximum)
3. IDF increases as term rarity increases

Intuition: Common terms provide little discriminative power for distinguishing documents. The IDF weight penalizes ubiquitous terms and rewards rare, potentially content-bearing terms.

Example: For a corpus of 6 documents where "naive" appears in 3:

$$\mathrm{idf}_{\text{naive}} = \log_{10} \frac{6}{3} = \log_{10}(2) \approx 0.301$$

6.4 The TF-IDF Formula

The complete TF-IDF weight combines dampened term frequency with inverse document frequency:

$$w_{t,d} = \ln(\mathrm{tf}(t,d) + 1) \cdot \log_b \frac{N}{\mathrm{df}_t}$$

Component Roles:

- $\ln(\mathrm{tf}(t,d) + 1)$: Local weight capturing term importance within the specific document
- $\log_b(N/\mathrm{df}_t)$: Global weight capturing term specificity across the corpus

Interpretation: A term receives high TF-IDF weight when it:

1. Occurs frequently in the target document (high local importance)
2. Occurs rarely across the corpus (high global specificity)

Common function words (e.g., "the," "is") receive low weights because despite high term frequencies, their IDF approaches zero due to near-universal document occurrence.

6.5 Worked Example: Computing a TF-IDF Matrix

Given Documents:

1. "the cat sat on the mat"
2. "my cat sat on my cat"
3. "my cat sat on the mat on my cat"

Step 1: Construct Term Frequency Matrix

Vocabulary: $\mathbf{T} = [\mathrm{the}, \mathrm{cat}, \mathrm{sat}, \mathrm{on}, \mathrm{mat}, \mathrm{my}]$

$$\mathbf{X}_{\text{tf}} = \begin{array}{c|cccccc} & \text{the} & \text{cat} & \text{sat} & \text{on} & \text{mat} & \text{my} \\ \hline d_1 & 2 & 1 & 1 & 1 & 1 & 0 \\ d_2 & 0 & 2 & 1 & 1 & 0 & 2 \\ d_3 & 1 & 2 & 1 & 2 & 1 & 2 \end{array}$$

Step 2: Compute Document Frequencies

Term	Documents Containing	df_t
the	d_1, d_3	2
cat	d_1, d_2, d_3	3
sat	d_1, d_2, d_3	3
on	d_1, d_2, d_3	3
mat	d_1, d_3	2
my	d_2, d_3	2

Step 3: Compute IDF Values

With $N = 3$ documents and base $b = 10$:

Term	$\text{idf}_t = \log_{10}(3/\text{df}_t)$
the	$\log_{10}(3/2) \approx 0.176$
cat	$\log_{10}(3/3) = 0$
sat	$\log_{10}(3/3) = 0$
on	$\log_{10}(3/3) = 0$
mat	$\log_{10}(3/2) \approx 0.176$
my	$\log_{10}(3/2) \approx 0.176$

Step 4: Compute TF-IDF Weights

For d_1, term "the": $w_{\text{the},d_1} = \ln(2+1) \cdot 0.176 = 1.099 \cdot 0.176 \approx 0.193$

For d_1, term "cat": $w_{\text{cat},d_1} = \ln(1+1) \cdot 0 = 0.693 \cdot 0 = 0$

For d_1, term "mat": $w_{\text{mat},d_1} = \ln(1+1) \cdot 0.176 = 0.693 \cdot 0.176 \approx 0.122$

Complete TF-IDF Matrix:

$$\mathbf{X}_{\text{tf-idf}} = \begin{array}{c|cccccc} & \text{the} & \text{cat} & \text{sat} & \text{on} & \text{mat} & \text{my} \\ \hline d_1 & 0.193 & 0.000 & 0.000 & 0.000 & 0.122 & 0.000 \\ d_2 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.193 \\ d_3 & 0.122 & 0.000 & 0.000 & 0.000 & 0.122 & 0.193 \end{array}$$

Observation: Terms appearing in all three documents (cat, sat, on) receive zero weight, as they provide no discriminative information. The matrix effectively highlights which terms distinguish each document.

7. Vector Space Model and Similarity Measures

7.1 Documents as Vectors

Under the vector space model, each document is represented as a point (or equivalently, a vector from the origin) in an M-dimensional space where each dimension corresponds to a vocabulary term.

$$\vec{d} \in \mathbb{R}^M$$

This geometric interpretation enables the application of distance and similarity measures to quantify document relatedness.

Core Insight: Documents that are topically similar will tend to use similar words with similar frequencies, and thus their vector representations will be proximate in the vector space.

7.2 Euclidean Distance

The **Euclidean distance** (or ℓ_2 distance) measures the straight-line distance between two points in the vector space:

$$d_E(\vec{p}, \vec{q}) = \sqrt{\sum_{i=1}^{M} (\vec{p}_i - \vec{q}_i)^2} = \|\vec{p} - \vec{q}\|_2$$

Properties:

- Metric: satisfies non-negativity, identity of indiscernibles, symmetry, and triangle inequality
- Range: $[0, \infty)$
- Interpretation: Absolute positional difference in vector space

Limitation: Euclidean distance is sensitive to vector magnitude. Long documents with many word occurrences will be distant from short documents even if topically identical, simply due to scale differences.

7.3 The Jaccard Coefficient

The **Jaccard coefficient** measures similarity between sets based on the ratio of intersection to union:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

where A and B are the sets of terms present in the respective documents.

Properties:

- Range: $[0, 1]$
- $J(A, B) = 1$ if and only if $A = B$
- $J(A, B) = 0$ if and only if $A \cap B = \emptyset$

Interpretation:

- Numerator: number of terms appearing in both documents
- Denominator: number of terms appearing in either document

Note: The Jaccard coefficient operates on sets (presence/absence) and does not account for term frequencies. It is appropriate for binary document representations.

7.4 Cosine Similarity

Cosine similarity measures the cosine of the angle between two vectors, providing a magnitude-invariant similarity measure:

$$\cos(\vec{p}, \vec{q}) = \frac{\vec{p} \cdot \vec{q}}{||\vec{p}||_2 \cdot ||\vec{q}||_2}$$

where the **dot product** is defined as:

$$\vec{p} \cdot \vec{q} = \sum_{i=1}^{M} \vec{p}_i \cdot \vec{q}_i$$

and the ℓ_2 **norm** (Euclidean norm) is:

$$||\vec{p}||_2 = \sqrt{\sum_{i=1}^{M} \vec{p}_i^2} = \sqrt{\vec{p} \cdot \vec{p}}$$

Properties:

- Range: $[-1, 1]$ in general; $[0, 1]$ for non-negative vectors (as in TF-IDF)
- $\cos(\vec{p}, \vec{q}) = 1$ when vectors point in the same direction
- $\cos(\vec{p}, \vec{q}) = 0$ when vectors are orthogonal (no shared terms)
- Invariant to vector magnitude: $\cos(c\vec{p}, \vec{q}) = \cos(\vec{p}, \vec{q})$ for $c > 0$

Advantage over Euclidean Distance: Document length does not affect similarity. A short document and a long document on the same topic (with proportionally similar term distributions) will have high cosine similarity despite large Euclidean distance.

7.5 Vector Normalization

ℓ_2 **normalization** transforms a vector to unit length:

$$\hat{\vec{p}} = \frac{\vec{p}}{||\vec{p}||_2}$$

After normalization, $||\hat{\vec{p}}||_2 = 1$.

Utility: For ℓ_2-normalized vectors, cosine similarity simplifies to the dot product:

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

This simplification reduces computational cost when computing pairwise similarities across large document collections.

Relationship to Euclidean Distance: For unit-normalized vectors:

$$d_E(\hat{\vec{p}}, \hat{\vec{q}})^2 = 2(1 - \cos(\hat{\vec{p}}, \hat{\vec{q}}))$$

Thus, on the unit hypersphere, Euclidean distance and cosine similarity are monotonically related.

7.6 Worked Examples: Computing Distances and Similarities

Given Vectors:

	dim_1	dim_2
$\vec{x}_1$	6.6	6.2
$\vec{x}_2$	9.7	9.9
$\vec{x}_3$	1.3	2.7
$\vec{x}_4$	1.3	1.3

Example 1: Euclidean Distance between $\vec{x}_1$ and $\vec{x}_4$

$$d_E(\vec{x}_1, \vec{x}_4) = \sqrt{(6.6 - 1.3)^2 + (6.2 - 1.3)^2}$$

$$= \sqrt{(5.3)^2 + (4.9)^2} = \sqrt{28.09 + 24.01} = \sqrt{52.10} \approx 7.218$$

Example 2: Cosine Similarity between $\vec{x}_2$ and $\vec{x}_3$

Dot product:

$$\vec{x}_2 \cdot \vec{x}_3 = 9.7 \times 1.3 + 9.9 \times 2.7 = 12.61 + 26.73 = 39.34$$

Norms:

$$\|\vec{x}_2\|_2 = \sqrt{9.7^2 + 9.9^2} = \sqrt{94.09 + 98.01} = \sqrt{192.10} \approx 13.86$$

$$\|\vec{x}_3\|_2 = \sqrt{1.3^2 + 2.7^2} = \sqrt{1.69 + 7.29} = \sqrt{8.98} \approx 2.997$$

Cosine similarity:

$$\cos(\vec{x}_2, \vec{x}_3) = \frac{39.34}{13.86 \times 2.997} \approx \frac{39.34}{41.54} \approx 0.947$$

Example 3: Jaccard Coefficient

Given binary vectors:

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
$\vec{x}_1$	1	0	1	1	1	0	1	1	0	0
$\vec{x}_2$	0	1	0	1	1	0	1	1	0	0

Intersection (both have 1): positions 4, 5, 7, 8 → $|A \cap B| = 4$

Union (at least one has 1): positions 1, 2, 3, 4, 5, 7, 8 → $|A \cup B| = 7$

$$J(\vec{x}_1, \vec{x}_2) = \frac{4}{7} \approx 0.571$$

8. Conclusion

This chapter has established the foundational mathematical framework for computational text analysis. The progression from raw text through tokenization, vectorization, term weighting, and similarity computation constitutes the standard preprocessing pipeline for text mining and natural language processing applications.

Key Contributions:

1. **Linguistic Complexity**: Language understanding requires processing at multiple levels—morphological, syntactic, semantic, and pragmatic—each introducing potential ambiguity.
2. **Vector Representations**: The Bag-of-Words model transforms variable-length text into fixed-dimensional vectors, with binary, term frequency, and TF-IDF variants offering increasing sophistication.
3. **TF-IDF Weighting**: The formulation $w_{t,d} = \ln(\mathrm{tf}(t,d) + 1) \cdot \log_b(N/\mathrm{df}_t)$ balances local term importance against global term specificity, suppressing common words while highlighting discriminative terms.
4. **Similarity Measures**: Euclidean distance, Jaccard coefficient, and cosine similarity provide complementary perspectives on document relatedness, with cosine similarity preferred for its magnitude invariance.
5. **Normalization**: ℓ_2 normalization enables efficient similarity computation and ensures comparability across documents of different lengths.

These foundations support downstream applications including document classification (Week 3), information retrieval, clustering, and serve as the conceptual basis for understanding more sophisticated representation learning approaches (Week 4).

9. References

1. Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11), 613–620.
2. Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1), 11–21.
3. Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

4. Jurafsky, D., & Martin, J. H. (2023). *Speech and Language Processing* (3rd ed. draft). Retrieved from https://web.stanford.edu/~jurafsky/slp3/

Appendix A: Mathematical Notation Summary

Symbol	Definition
d, d_i	Document, i-th document
$\mathbf{D}$	Corpus (collection of documents)
t, t_j	Term (token), j-th term
$\mathbf{T}$	Vocabulary (set of unique terms)
V	Alternative notation for vocabulary
N	Number of documents in corpus
M	Size of vocabulary
$\vec{d}$, $\vec{x}$	Document vector
$\mathbf{X}$	Term-document matrix
$\mathrm{tf}(t, d)$	Term frequency of t in d
df_t	Document frequency of term t
idf_t	Inverse document frequency of t
$w_{t,d}$	TF-IDF weight of term t in document d
$\vec{p} \cdot \vec{q}$	Dot product of vectors $\vec{p}$ and $\vec{q}$
$\vec{p} \bullet \vec{q}$	Alternative notation for dot product
$|\vec{p}|_2$	ℓ_2 (Euclidean) norm of $\vec{p}$
$d_E(\vec{p}, \vec{q})$	Euclidean distance
$J(A, B)$	Jaccard coefficient
$\cos(\vec{p}, \vec{q})$	Cosine similarity
$\ln$	Natural logarithm (base e)
$\log$	Logarithm (context-dependent base)
$\lg$	Logarithm base 10
$\log_b$	Logarithm base b

Glossary

Bag-of-Words (BoW): A document representation that treats text as an unordered collection of words, discarding sequence information.

Cosine Similarity: A similarity measure equal to the cosine of the angle between two vectors; ranges from -1 to 1 (0 to 1 for non-negative vectors).

Document Frequency (df_t): The number of documents in a corpus containing a given term t.

Dot Product: The sum of element-wise products of two vectors; $\vec{p} \cdot \vec{q} = \sum_i p_i q_i$.

Euclidean Distance: The straight-line distance between two points in Euclidean space; the ℓ_2 norm of the difference vector.

Inverse Document Frequency (IDF): A measure of term specificity; $\mathrm{idf}_t = \log(N/\mathrm{df}_t)$.

Jaccard Coefficient: A set similarity measure; the ratio of intersection size to union size.

ℓ_2 **Norm**: The Euclidean length of a vector; $\|\vec{p}\|_2 = \sqrt{\sum_i p_i^2}$.

Term-Document Matrix: A matrix where rows represent documents, columns represent terms, and entries represent term weights.

Term Frequency ($\mathrm{tf}(t, d)$): The count of occurrences of term t in document d.

TF-IDF: Term Frequency–Inverse Document Frequency; a term weighting scheme combining local and global term importance.

Tokenization: The process of segmenting text into discrete units (tokens).

Vector Space Model: A representation framework where documents are points in a high-dimensional space defined by vocabulary terms.

Vocabulary: The set of unique terms across a document collection.