Cosine Similarity and Its Applications in the Domains of Artificial Intelligence

< D R A F T >

David Oniani
Luther College
oniada01@luther.edu

April 23, 2020

Abstract

Choosing the right metric [1] can be crucial to designing performant artificial intelligence models. Thousands of packages and libraries have been built and written just for providing these metrics. Cosine similarity is one of many metrics used extensively in natural language processing (NLP) tasks. The paper will introduce the technique and discuss its advantages and disadvantages as well as compare it to other approaches. Additionally, sample implementations of the technique will also be provided.

Table of Contents

1 Introduction

There are many approaches for determining whether two texts are semantically similar to each other. Cosine similarity is one of those methods. Derived from the law of cosines [2], the technique is now widely used in natural language processing (and other) problems.

In order to understand how cosine similarity works, let us first discuss how a text document can be modeled.

1.1 Text Document Modeling

There are a number of ways in which a text document can be modeled. This includes a *bag-of-words* modeling, where the frequency of a term in a text document represents its weight and therefore, and *tf-idf* vectorization. The primary goal of text document modeling is to transform the textual data into the numeric data (in this case, term vectors). Once the numeric data is obtained, we can then apply mathematical techniques including text semantic similarity approaches such as cosine similarity in order to compare if two documents are similar to each other.

1.2 Cosine Similarity

Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them [3]. For two vectors $\vec{a}$ and $\vec{b}$, the cosine can be computed as $\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}| \cos\theta$, where θ is the angle between the vectors. This is also known as the Euclidean dot product formula. The cosine of two vectors can also be computed using their coordinates: if $\vec{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$ and $\vec{b} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$, then $\vec{a} \cdot \vec{b} = a_1 b_1 + a_2 b_2$. Hence, for two vectors $\vec{a}$ and $\vec{b}$ with coordinates a_1, a_2 and b_1, b_2 respectively, the following holds:

$$\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}| \cos\theta = a_1 b_1 + a_2 b_2$$

If the vector coordinates are known, then one can also compute the magnitude of the vector. For vector $\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}$, the magnitude is $\sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$. Hence, if we have two vectors $\vec{a} = \begin{bmatrix} a_1 \\ a_2 \\ \cdots \\ a_n \end{bmatrix}$

and $\vec{b} = \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_n \end{bmatrix}$, then the cosine of an angle between them is:

$$\cos\theta = \frac{a_1 b_1 + a_2 b_2 + \cdots + a_n b_n}{\sqrt{a_1^2 + a_2^2 + \cdots + a_n^2}\sqrt{b_1^2 + b_2^2 + \cdots + b_n^2}}$$

In other words, if we have two vectors with all coordinates specified, it is possible to compute the cosine similarity between them.

1.3 Example

As an example, consider two vectors $\vec{a} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$ and $\vec{b} = \begin{bmatrix} 5 \\ 12 \end{bmatrix}$. Then the magnitude of a is $\sqrt{3^2 + 4^2} = 5$ and the magnitude of b is $\sqrt{5^2 + 12^2} = 13$. It follows that $\vec{a} \cdot \vec{b} = 15 + 48 = 63$ and $|\vec{a}| \times |\vec{b}| = 5 \times 13 = 65$. Therefore, the cosine of an angle between the vectors is $\frac{63}{65} \approx 0.969$ which is the similarity measure between these two vectors.

For those who like to think visually, below is the plot from which one could derive the cosine of the angle θ (e.g., by applying the law of cosines).

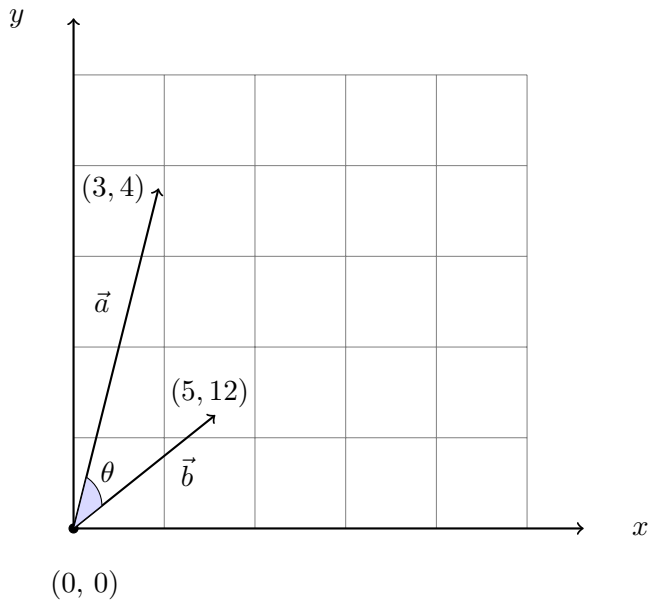

Figure 1: Vector Cosine Illustration.

2 Applications

Cosine similarity is one of the most commonly used approaches in calculating semantic similarity of texts. Therefore, it is naturally employed in natural language processing tasks. Many NLP applications need to compute the semantic similarity between two short texts. Search engines, for instance, need to model the relevance of a document to a query on the semantic level. Similarly, Q&A websites such as Stack Overflow and Quora need to determine whether a question has already been asked before. Cosine similarity is usually one of the typical first solutions to such problems.

Suppose that we have two documents D_1 and D_2 modeled as term vectors $\vec{t_1}$ and $\vec{t_2}$ respectively. Then the similarity of two documents corresponds to the correlation between the vectors and can be quantified as a cosine of the angle between the vectors. Thus, the similarity formula [4] is:

$$SIM_C(\vec{t_1}, \vec{t_2}) = \frac{\vec{t_1} \cdot \vec{t_2}}{|\vec{t_1}| \times |\vec{t_2}|}.$$

As a result, the similarity value is non-negative, bounded by the closed interval $[0, 1]$.

It is important to note that the metric is a measurement of orientation and not the magnitude. It can be thought of as a comparison of documents on a normalized space since it does not only consider the magnitude of word counts of each document, but also the angle between the documents.

Below find a simple Python implementation of cosine similarity.

```python
import numpy as np


def cosine_similarity(a: np.array, b: np.array) -> float:
    """Returns the cosine similarity value of two vectors."""

    dot_product: float = np.dot(a, b)
    magnitude_a: float = np.linalg.norm(a)
    magnitude_b: float = np.linalg.norm(b)
    similarity: float = dot_product / (magnitude_a * magnitude_b)

    return similarity


if __name__ == "__main__":
    a: np.array = np.array([1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1])
    b: np.array = np.array([0, 2, 1, 0, 1, 3, 0, 0, 0, 1, 0])
    print(cosine_similarity(a, b))  # Prints out 0.5
```

Figure 2: Sample Python Implementation of Cosine Similarity.

The program is fairly simplistic (yet performant) and as shown above, `cosine_similarity` function takes two vectors and prints out the cosine similarity between them. Vectors `a` and `b` can be thought of as vectorized text (i.e., the vectorization algorithm has already been aplied to two texts).

3 Performance and Comparison to Other Approaches

Performance of cosine similarity can be altered by the approach used for vectorization [5] (e.g., tf-idf vectorization might yield better results than Word2Vec vectorization). Needless to say that the right vectorization strategy could drastically increase the performance, while the poor vectorization will, unsurprisingly, result in the poor performance.

There are many other approaches for finding semantic similarity between the texts. The recent advancements in the domains of artificial intelligence and natural language processing allowed for development of models such as BERT [6], BioBERT [7], and Universal Sentence Encoder (USE) [8], all of which can be used for high-accuracy text semantic similarity score computations. The ad-

vantage of the cosine similarity, however, is its simplicity and the fact that one could run it over massive datasets. Doing the same with models like BioBERT could possibly take a lot of time and hence, be highly inefficient.

4 Current Work

Cosine similarity is very much used in the modern, state-of-the-art papers, such as ones cited in this paper. Its flexibility allows one to apply it under virtually any settings, as long as documents can be represented as vectors. Besides, it is widely used for benchmarking purposes and is usually a standard against which the new approaches are tested.

5 Summary

We have introduced cosine similarity and walked through examples to strengthen the understanding of the concept. Besides, we discussed the importance of cosine similarity in the domains of artificial intelligence and natural language processing and its prevalence in the state-of-the-art work. Sample Python implementation of the algorithm was also provided.

References

[1] Rachel Thomas and David Uminsky. "The Problem with Metrics is a Fundamental Problem for AI". In: (2020). URL: https://arxiv.org/abs/2002.08512.

[2] Wikipedia The Free Encyclopedia. *Cosine Formula for Dot Product*. URL: https://proofwiki.org/wiki/Cosine_Formula_for_Dot_Product.

[3] Wikipedia The Free Encyclopedia. *Cosine similarity*. URL: https://en.wikipedia.org/wiki/Cosine_similarity.

[4] Anna Huang. "Similarity measures for text document clustering". In: *Proceedings of the Sixth New Zealand Computer Science Research Student Conference (NZCSRSC2008), Christchurch, New Zealand*. 2008.

[5] Pinky Sitikhu et al. "A Comparison of Semantic Similarity Methods for Maximum Human Interpretability". In: (2019).

[6] Iulia Turc et al. “Well-Read Students Learn Better: On the Importance of Pre-training Compact Models”. In: *arXiv preprint arXiv:1908.08962v2* (2019).

[7] Jinhyuk Lee et al. “BioBERT: a pre-trained biomedical language representation model for biomedical text mining”. In: *Bioinformatics* (Sept. 2019). ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz682. URL: https://doi.org/10.1093/bioinformatics/btz682.

[8] Daniel Cer et al. “Universal Sentence Encoder”. In: (2018). eprint: arXiv:1803.11175.