PRECOG TASK

PHRASE & SENTENCE SIMILARITY

SWARANG JOSHI
2022114010

Applications of Sentence Similarity

Various CL tasks

1. Data Retrieval
2. Conversational Dialogue Systems
3. Text Mining
4. Text Summarization & Categorization
5. Machine Translations Systems

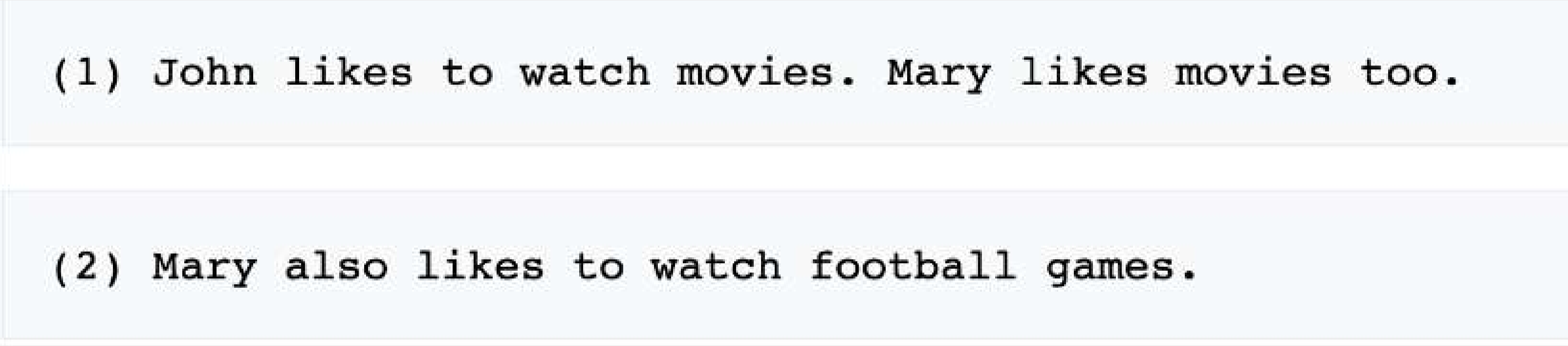
Aim

To find sentence similarity between short sentences with context.

Bag Of Words

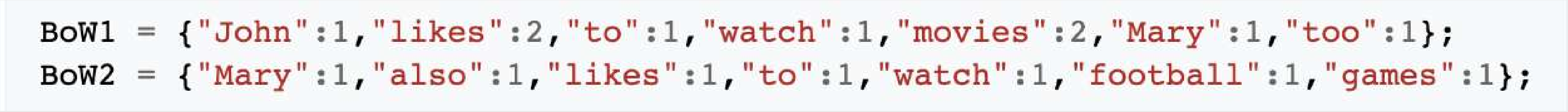
In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity.

For Example-



```
(1) John likes to watch movies. Mary likes movies too.

(2) Mary also likes to watch football games.
```



```
BoW1 = {"John":1,"likes":2,"to":1,"watch":1,"movies":2,"Mary":1,"too":1};
BoW2 = {"Mary":1,"also":1,"likes":1,"to":1,"watch":1,"football":1,"games":1};
```

Descriptive Features-based methods

The feature vector method tries to represent a sentence using a set of predefined features. Basically, a word in a sentence is represented using semantic features, for example, nouns may have features such as HUMAN (with value of human or nonhuman), SOFTNESS (soft or hard), and POINTNESS (pointed or rounded).

LET’S TRY TO COUNTER THE DRAWBACKS

Approach and Method

Use of Lexical Resources too...

So to approach the problem statement, we need to consider,

1. Associating the words with sense.
2. Combining the Word sense to represent a sentence with some sense suitable for further computation.
3. Incorporating Word Order into our computation.

Associating word with a sense

WordNet

The primary structure of the WordNet is based on synonymy.

WSD and Pywsd;

Max similarity Algo. Pre-Defined in pyWSD

Sentence Similarity

$$S(T_1, T_2) = \delta S_s + (1 - \delta) S_r$$

And the Example Sentence that we will be going through our method are :

- T1 : RAM keeps things being worked with.
- T2 : The CPU uses RAM as a short-term memory store.

Computing Joint Word Set

The joint word set T contains all the distinct words from T1 and T2.

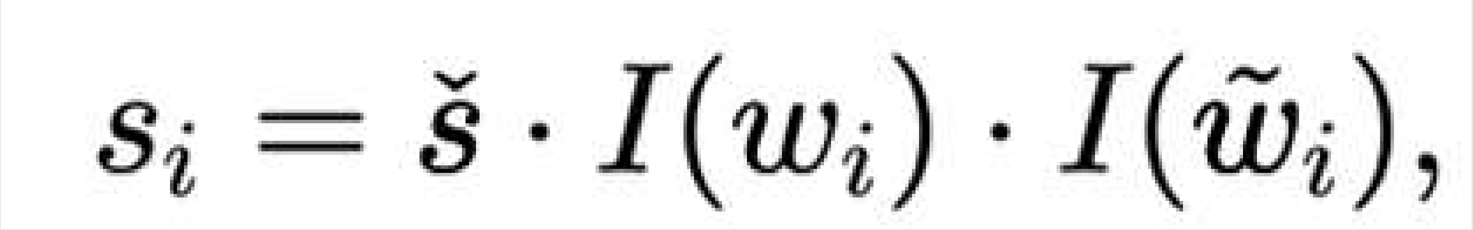
The Joint Word Set T is:

T = {RAM keeps things being worked with The CPU uses as a short-term memory store}

Computing Semantic Vector

So our Semantic Vector should would just be the word similarity score for each word from the Joint Word Set to the most similar word in our Sentence whose Semantic Vector is being calculated

So we define the ith coordinate Semantic Vector as



$$s_i = \check{s} \cdot I(w_i) \cdot I(\tilde{w}_i),$$

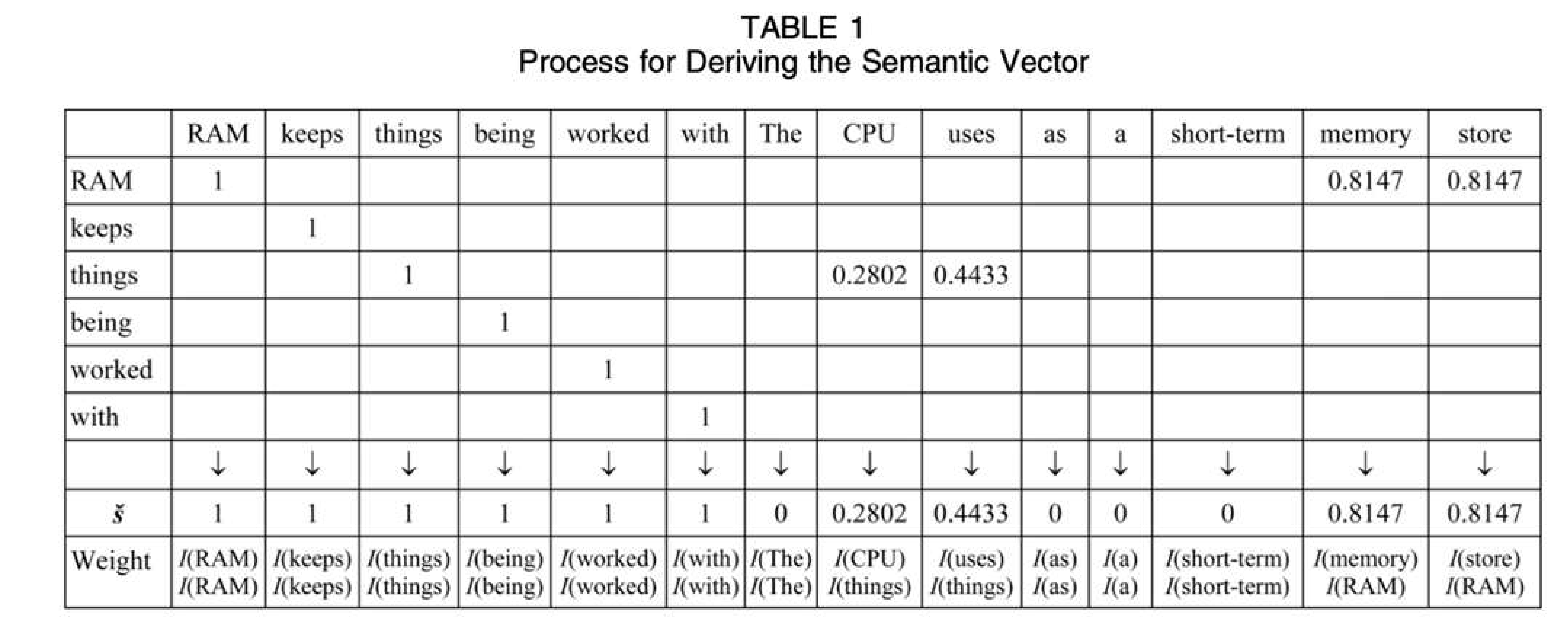
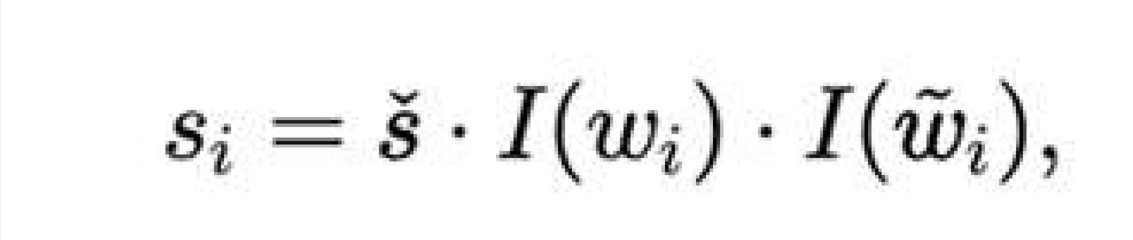


TABLE 1
Process for Deriving the Semantic Vector

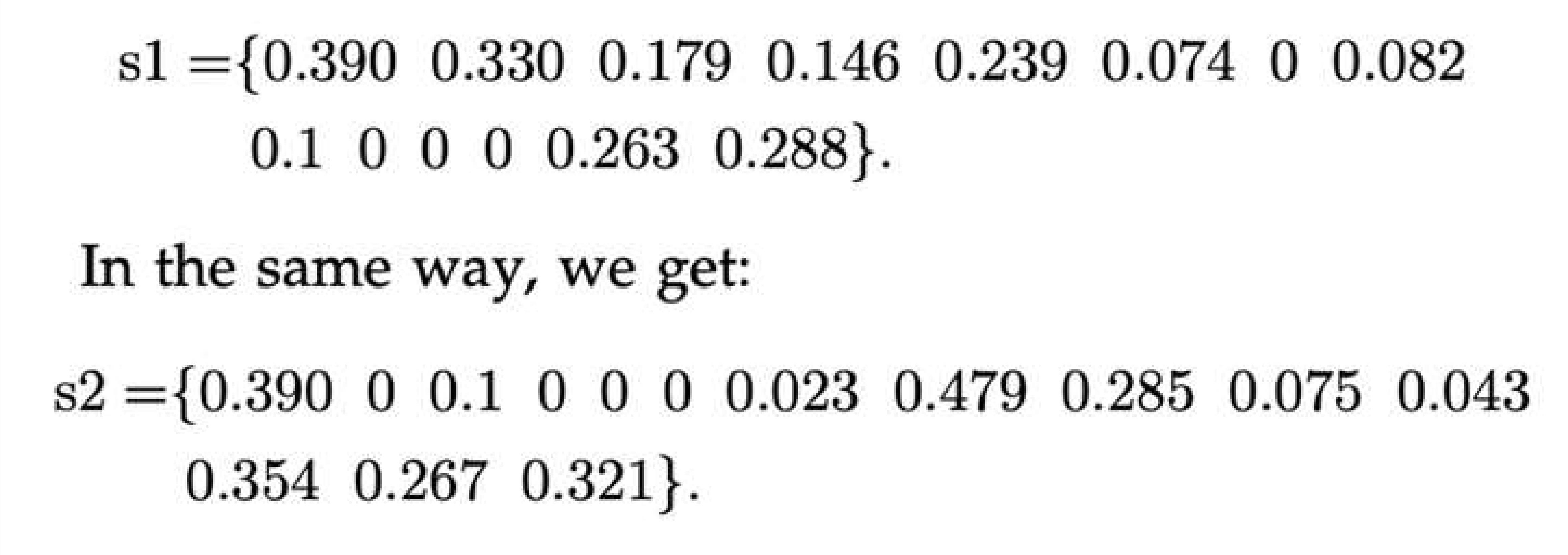
	RAM	keeps	things	being	worked	with	The	CPU	uses	as	a	short-term	memory	store
RAM	1												0.8147	0.8147
keeps		1												
things			1					0.2802	0.4433					
being				1										
worked					1									
with						1								
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
š	1	1	1	1	1	1	0	0.2802	0.4433	0	0	0	0.8147	0.8147
Weight	*I*(RAM) *I*(RAM)	*I*(keeps) *I*(keeps)	*I*(things) *I*(things)	*I*(being) *I*(being)	*I*(worked) *I*(worked)	*I*(with) *I*(with)	*I*(The) *I*(The)	*I*(CPU) *I*(things)	*I*(uses) *I*(things)	*I*(as) *I*(as)	*I*(a) *I*(a)	*I*(short-term) *I*(short-term)	*I*(memory) *I*(RAM)	*I*(store) *I*(RAM)

The first row lists words in the joint word set T, the first column lists words in sentence T1 and all words are listed in the order as they appear in T and T1.

- The last row lists the corresponding information content for weighting the significance of the word. As a result, the semantic vector for T1 is:



$$s_i = \check{s} \cdot I(w_i) \cdot I(\tilde{w}_i),$$



s1 ={0.390 0.330 0.179 0.146 0.239 0.074 0 0.082
0.1 0 0 0 0.263 0.288}.

In the same way, we get:

s2 ={0.390 0 0.1 0 0 0 0.023 0.479 0.285 0.075 0.043
0.354 0.267 0.321}.

Computing Semantic Similarity Score

Present with ease and wow any audience with Canva.

1. As we can represent the two semantic Vector for the respective sentence as a simple vectors pointing to some direction in the nD space.

2. The more the angle between the vector, the more dissimilar the sentence are being most dissimilar when being orthogonal and being most similar when parallel to each other.

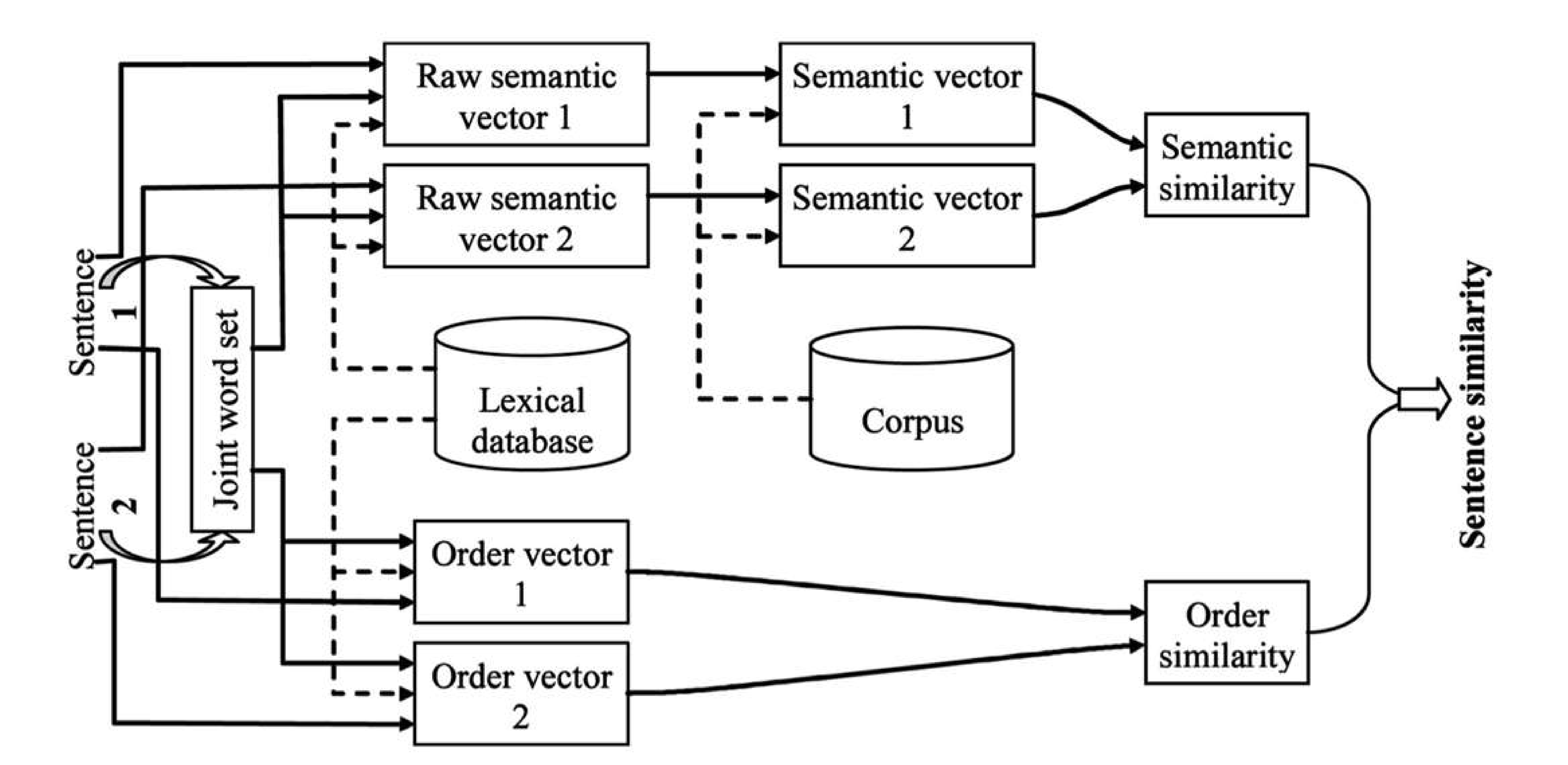
3. Hence we calculate the cosine of the angle between the vector which the most similar give score of 1
and in orthogonal gives 0 semantic similarity Score.

- T1: RAM keeps things being worked with.
- T2: The CPU uses RAM as a short-term memory store.

$$\begin{aligned}\mathrm{s1} =\{&0.390\ \ 0.330\ \ 0.179\ \ 0.146\ \ 0.239\ \ 0.074\ \ 0\ \ 0.082\\ &0.1\ \ 0\ \ 0\ \ 0\ \ 0.263\ \ 0.288\}.\end{aligned}$$

$$\begin{aligned}\mathrm{s2} =\{&0.390\ \ 0\ \ 0.1\ \ 0\ \ 0\ \ 0\ \ 0.023\ \ 0.479\ \ 0.285\ \ 0.075\ \ 0.043\\ &0.354\ \ 0.267\ \ 0.321\}.\end{aligned}$$

- As we defined, $$S_s = \frac{\mathbf{s_1} \cdot \mathbf{s_2}}{\|\ \mathbf{s_i}\ \| \cdot \|\ \mathbf{s_2}\ \|}.$$
- From s1 and s2, the semantic similarity between the two sentences is S_s = 0.2023


Sentence 1
Sentence 2
Joint word set
Raw semantic vector 1
Raw semantic vector 2
Semantic vector 1
Semantic vector 2
Semantic similarity
Lexical database
Corpus
Order vector 1
Order vector 2
Order similarity
Sentence similarity

Computing Word Order Vector

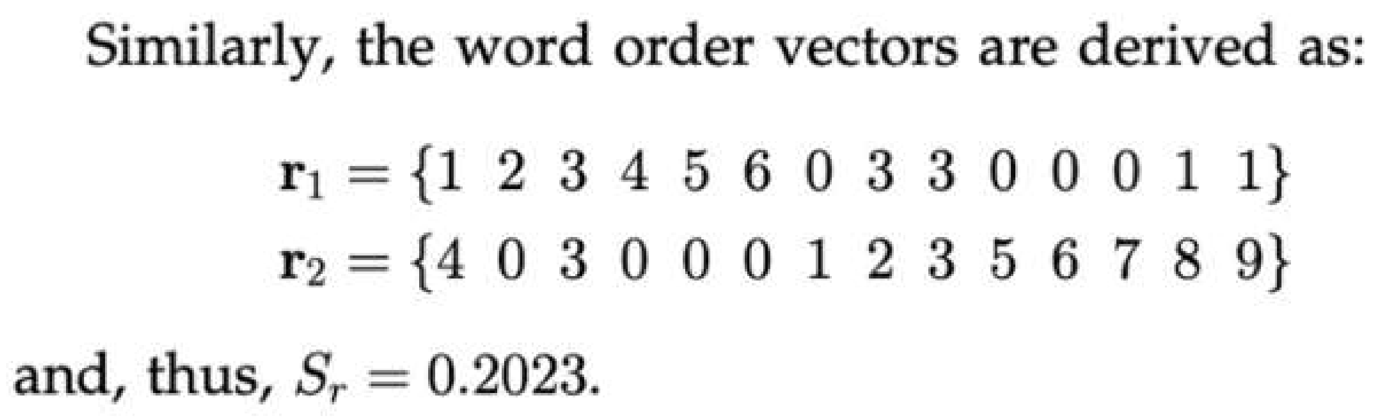
- Taking a simple example here :
- T1 : A quick brown dog jumps over the lazy fox.
- T2 : A quick brown fox jumps over the lazy dog.

Since these two sentences contain the same words, any methods based on ”bag of words” will give a decision that T1 and T2 are exactly the same. However, it is clear for a human interpreter that T1 and T2 are only similar to some extent. The dissimilarity between T1 and T2 is the result of the different word order.

For the example pair of sentences T1 and T2 , the joint word set is:

- T = {A quick brown dog jumps over the lazy fox}

Computing Word order Similarity



Similarly, the word order vectors are derived as:

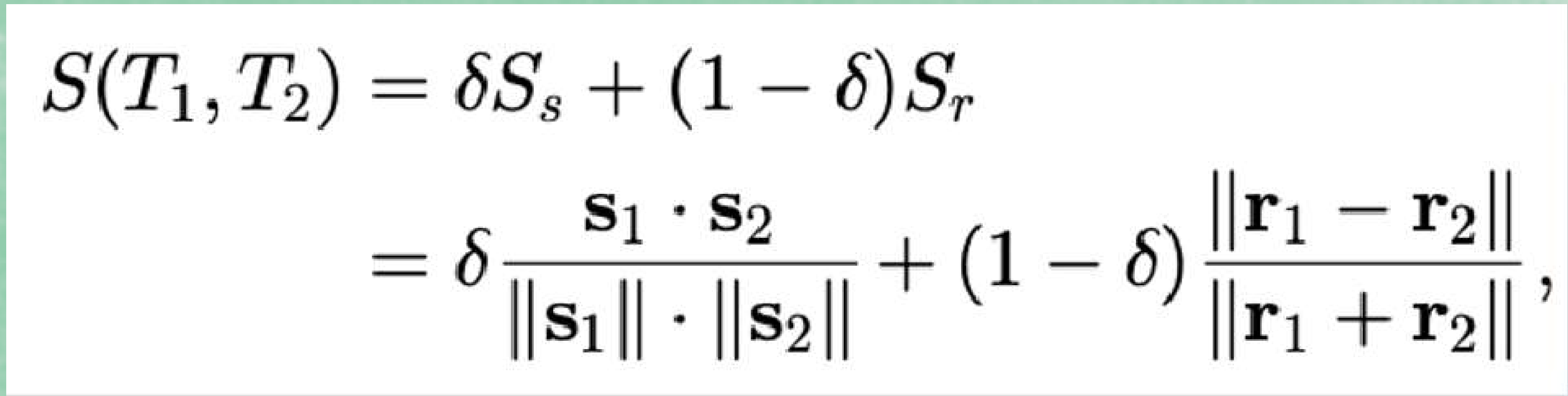
$$\mathbf{r}_1 = \{1\ 2\ 3\ 4\ 5\ 6\ 0\ 3\ 3\ 0\ 0\ 0\ 1\ 1\}$$
$$\mathbf{r}_2 = \{4\ 0\ 3\ 0\ 0\ 0\ 1\ 2\ 3\ 5\ 6\ 7\ 8\ 9\}$$

and, thus, $S_r = 0.2023$.

- T1: RAM keeps things being worked with.
- T2: The CPU uses RAM as a short-term memory store.

- We define Word Order Similarity as

$$S_r = 1 - \frac{\| \mathbf{r}_1 - \mathbf{r}_2 \|}{\| \mathbf{r}_1 + \mathbf{r}_2 \|}.$$

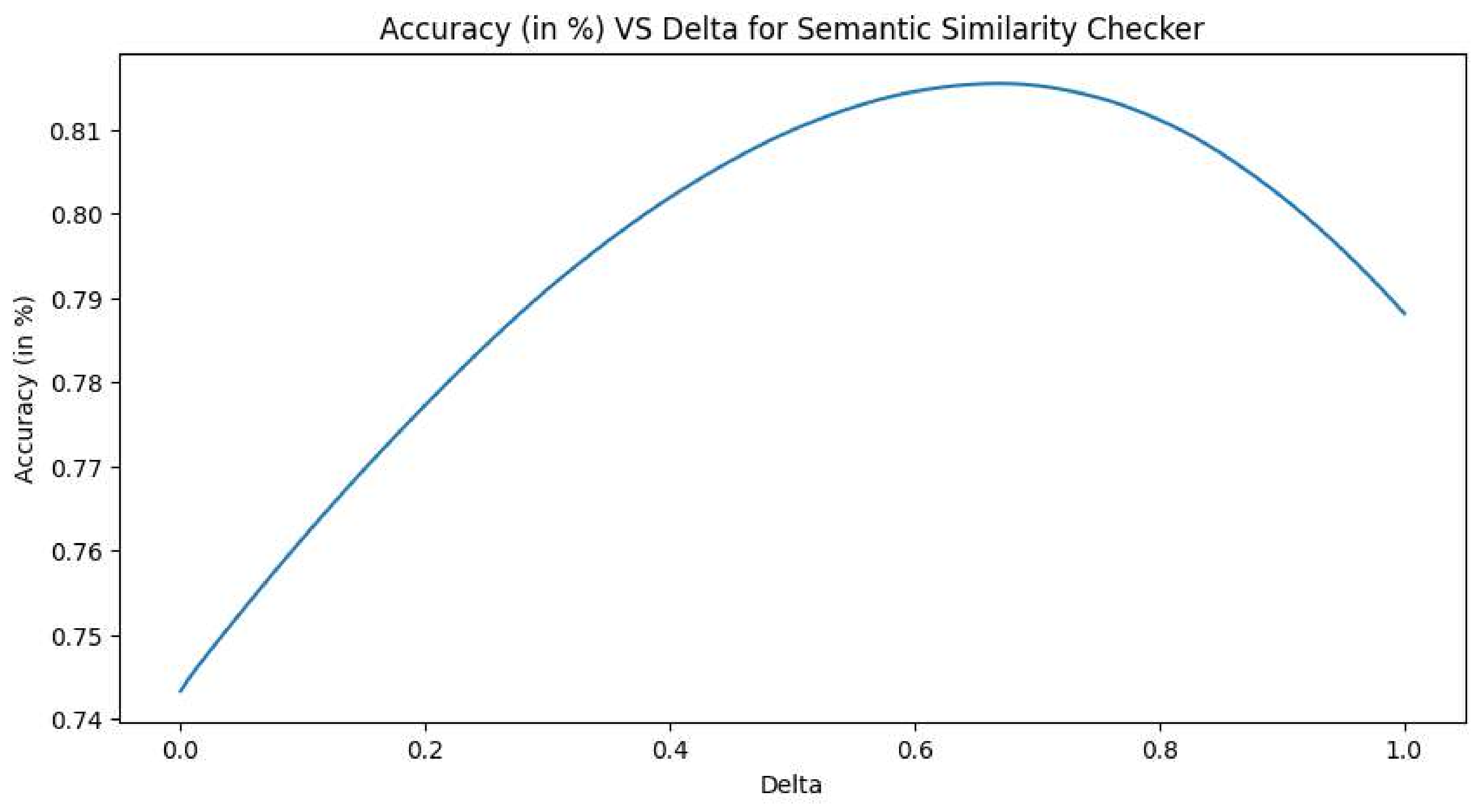


$$\begin{aligned} S(T_1, T_2) &= \delta S_s + (1-\delta) S_r \\ &= \delta \frac{\mathbf{s}_1 \cdot \mathbf{s}_2}{\|\mathbf{s}_1\| \cdot \|\mathbf{s}_2\|} + (1-\delta) \frac{\|\mathbf{r}_1 - \mathbf{r}_2\|}{\|\mathbf{r}_1 + \mathbf{r}_2\|}, \end{aligned}$$

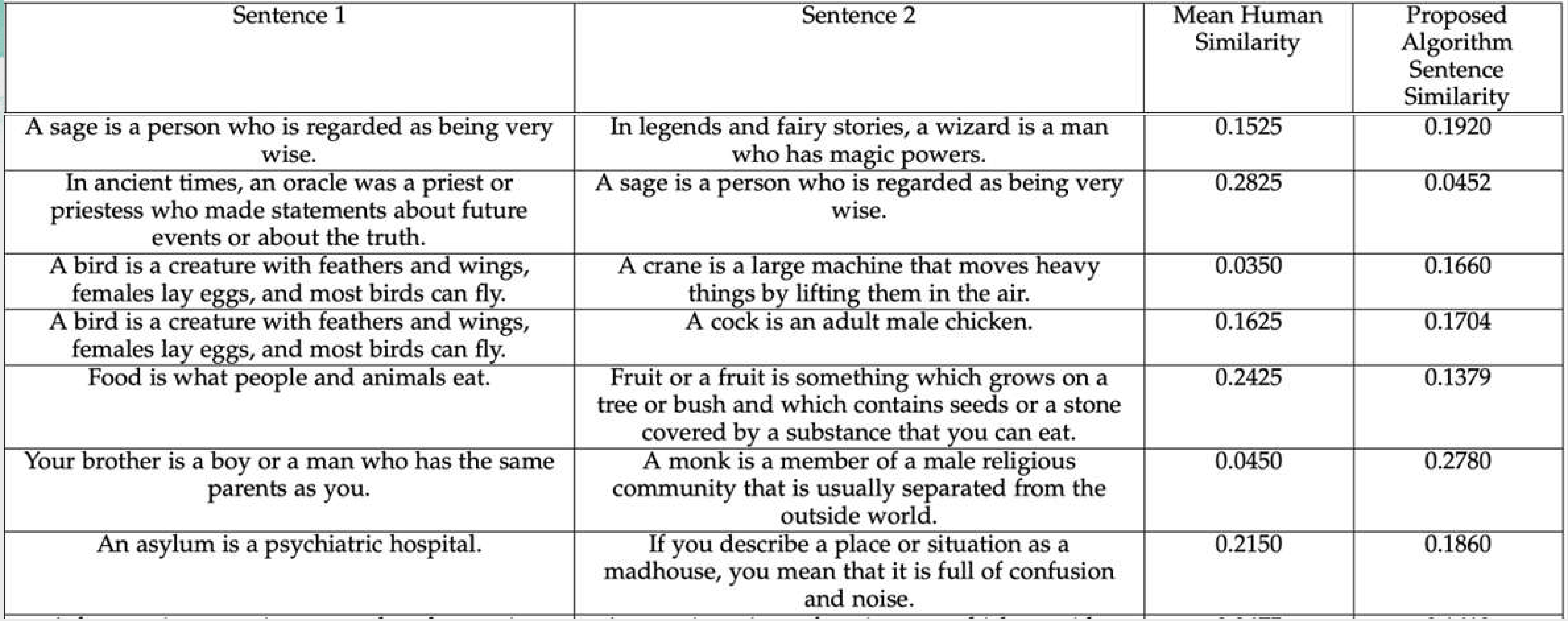
Finally,

the similarity between the sentences “RAM keeps things being worked with” and “The CPU uses RAM as a short-term memory store” is 0.5522, using 0.85 for delta (***δ***).

Optimising DELTA



Results



Sentence 1	Sentence 2	Mean Human Similarity	Proposed Algorithm Sentence Similarity
A sage is a person who is regarded as being very wise.	In legends and fairy stories, a wizard is a man who has magic powers.	0.1525	0.1920
In ancient times, an oracle was a priest or priestess who made statements about future events or about the truth.	A sage is a person who is regarded as being very wise.	0.2825	0.0452
A bird is a creature with feathers and wings, females lay eggs, and most birds can fly.	A crane is a large machine that moves heavy things by lifting them in the air.	0.0350	0.1660
A bird is a creature with feathers and wings, females lay eggs, and most birds can fly.	A cock is an adult male chicken.	0.1625	0.1704
Food is what people and animals eat.	Fruit or a fruit is something which grows on a tree or bush and which contains seeds or a stone covered by a substance that you can eat.	0.2425	0.1379
Your brother is a boy or a man who has the same parents as you.	A monk is a member of a male religious community that is usually separated from the outside world.	0.0450	0.2780
An asylum is a psychiatric hospital.	If you describe a place or situation as a madhouse, you mean that it is full of confusion and noise.	0.2150	0.1860

References

Yuhua Li, David McLean & Others

Sentence Similarity Based on Semantic Nets and Corpus Statistics

Xiaofei Sun, Yuxian Meng & Others

Sentence Similarity Based on Contexts

Thank you!