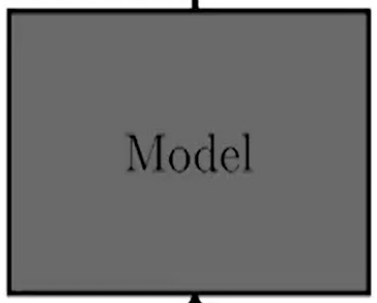
Word Representation

1. One-hot representation of words

- Let us start with a very simple motivation for why we are interested in vectorial representations of words

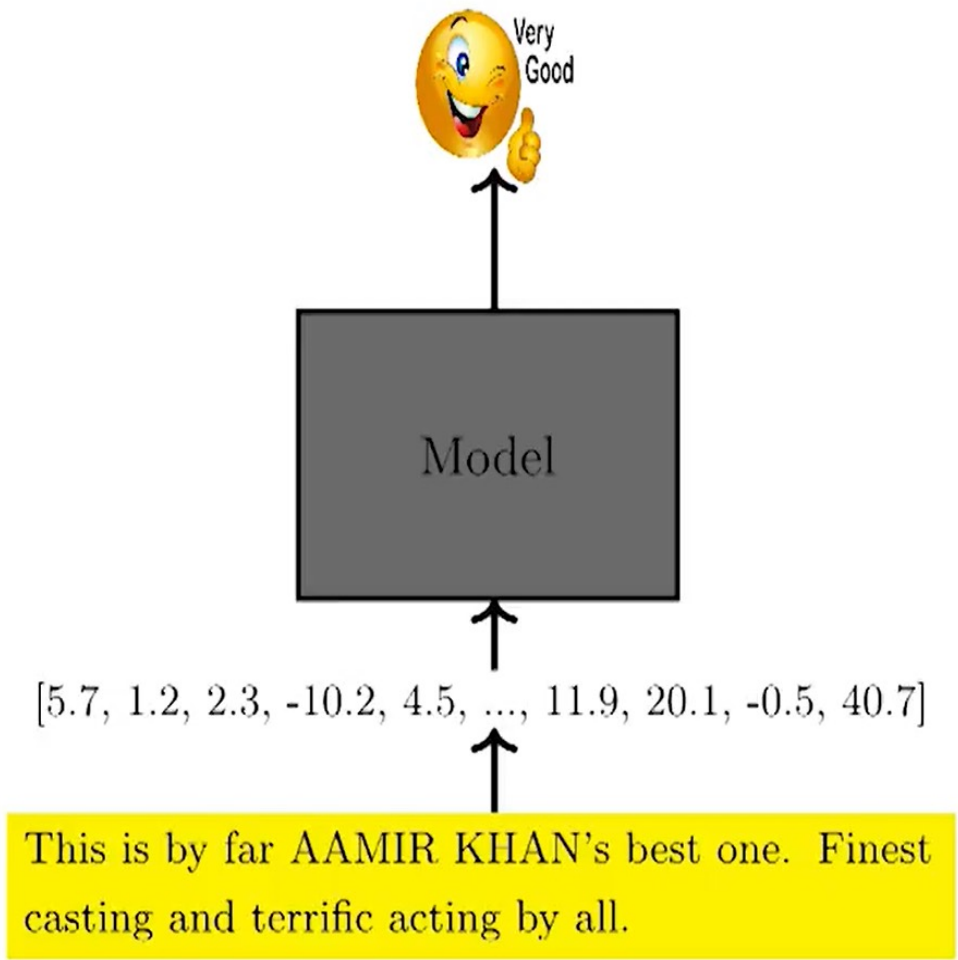
- Let us start with a very simple motivation for why we are interested in vectorial representations of words
- Suppose we are given an input stream of words (sentence, document, etc.) and we are interested in learning some function of it (say, $\hat{y} = sentiments(words)$)

This is by far AAMIR KHAN's best one. Finest casting and terrific acting by all.



- Let us start with a very simple motivation for why we are interested in vectorial representations of words
- Suppose we are given an input stream of words (sentence, document, etc.) and we are interested in learning some function of it (say, $\hat{y} = sentiments(words)$)
- Say, we employ a machine learning algorithm (some mathematical model) for learning such a function ($\hat{y} = f(\mathbf{x})$)

This is by far Aamir khan's best one.
Finest casting and terrific acting by all.



- Let us start with a very simple motivation for why we are interested in vectorial representations of words
- Suppose we are given an input stream of words (sentence, document, etc.) and we are interested in learning some function of it (say, $\hat{y} = sentiments(words)$)
- Say, we employ a machine learning algorithm (some mathematical model) for learning such a function ($\hat{y} = f(\mathbf{x})$)
- We first need a way of converting the input stream (or each word in the stream) to a vector $\mathbf{x}$ (a mathematical quantity)

Corpus:

- Human machine interface for computer applications
- User opinion of computer system response time
- User interface management system
- System engineering for improved response time

- Given a corpus, consider the set V of all unique words across all input streams (*i.e.*, all sentences or documents)

Corpus:

- Human machine interface for computer applications
- User opinion of computer system response time
- User interface management system
- System engineering for improved response time

V = [human,machine, interface, for, computer, applications, user, opinion, of, system, response, time, interface, management, engineering, improved]

- Given a corpus, consider the set V of all unique words across all input streams (*i.e.*, all sentences or documents)
- V is called the **vocabulary** of the corpus (*i.e.*, all sentences or documents)

Corpus:

- Human machine interface for computer applications
- User opinion of computer system response time
- User interface management system
- System engineering for improved response time

V = [human,machine, interface, for, computer, applications, user, opinion, of, system, response, time, interface, management, engineering, improved]

- Given a corpus, consider the set V of all unique words across all input streams (*i.e.*, all sentences or documents)
- V is called the **vocabulary** of the corpus (*i.e.*, all sentences or documents)
- We need a representation for every word in V
- One very simple way of doing this is to use one-hot vectors of size $|V|$

machine:

0	1	0	...	0	0	0

cat:	0	0	0	0	0	1	0
dog:	0	1	0	0	0	0	0
truck:	0	0	0	1	0	0	0

Problems:

- V tends to be very large (for example, 50K for PTB, 13M for Google 1T corpus)

cat:	0	0	0	0	0	1	0
dog:	0	1	0	0	0	0	0
truck:	0	0	0	1	0	0	0

Problems:

- V tends to be very large (for example, 50K for PTB, 13M for Google 1T corpus)
- These representations do not capture any notion of similarity
- Ideally, we would want the representations of cat and dog (both domestic animals) to be closer to each other than the representations of cat and truck

cat:	0	0	0	0	0	1	0
dog:	0	1	0	0	0	0	0
truck:	0	0	0	1	0	0	0

$$euclid\_dist(\textbf{cat}, \textbf{dog}) = \sqrt{2}$$
$$euclid\_dist(\textbf{dog}, \textbf{truck}) = \sqrt{2}$$
$$cosine\_sim(\textbf{cat}, \textbf{dog}) = 0$$
$$cosine\_sim(\textbf{dog}, \textbf{truck}) = 0$$

Problems:

- V tends to be very large (for example, 50K for PTB, 13M for Google 1T corpus)
- These representations do not capture any notion of similarity
- Ideally, we would want the representations of cat and dog (both domestic animals) to be closer to each other than the representations of cat and truck
- However, with 1-hot representations, the Euclidean distance between **any two words** in the vocabulary in $\sqrt{2}$
- And the cosine similarity between **any two words** in the vocabulary is 0

Suppose that our goal is to calculate the cosine similarity of the two documents given below.

- Document 1 = 'the best data science course'
- Document 2 = 'data science is popular'

After creating a word table from the documents, the documents can be represented by the following vectors:

	the	best	data	science	course	is	popular
D1	1	1	1	1	1	0	0
D2	0	0	1	1	0	1	1

- $D1 = [1, 1, 1, 1, 1, 0, 0]$
- $D2 = [0, 0, 1, 1, 0, 1, 1]$

Using these two vectors we can calculate cosine similarity. First, we calculate the dot product of the vectors:

$$D1 \cdot D2 = 1 \times 0 + 1 \times 0 + 1 \times 1 + 1 \times 1 + 1 \times 0 + 0 \times 1 + 0 \times 1 = 2$$

Second, we calculate the magnitude of the vectors:

$$\|D1\| = \sqrt{1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 0^2 + 0^2} = \sqrt{5}$$

$$\|D2\| = \sqrt{0^2 + 0^2 + 1^2 + 1^2 + 0^2 + 1^2 + 1^2} = \sqrt{4}$$

2. Distributional Representation

- *You shall know a word by the company it keeps - Firth, J. R. 1957:11*

A bank is a **financial** institution that accepts **deposits** from the public and creates **credit**.

- *You shall know a word by the company it keeps - Firth, J. R. 1957:11*
- Distributional similarity based representations
- This leads us to the idea of co-occurrence matrix

A bank is a **financial** institution that accepts **deposits** from the public and creates **credit**.

The idea is to use the accompanying words (financial, deposits, credit) to represent bank

Corpus:

- Human machine interface for computer applications
- User opinion of computer system response time
- User interface management system
- System engineering for improved response time

- A co-occurrence matrix is a **terms** × **terms** matrix which captures the number of times a term appears in the context of another term

Corpus:

- Human machine interface for computer applications
- User opinion of computer system response time
- User interface management system
- System engineering for improved response time

	human	machine	system	for	...	user
human	0	1	0	1	...	0
machine	1	0	0	1	...	0
system	0	0	0	1	...	2
for	1	1	1	0	...	0
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
user	0	0	2	0	...	0

Co-occurence Matrix

- A co-occurrence matrix is a **terms** × **terms** matrix which captures the number of times a term appears in the context of another term
- The context is defined as a window of k words around the terms
- Let us build a co-occurrence matrix for this toy corpus with $k = 2$
- This is also known as a **word** × **context** matrix

Corpus:

- Human machine interface for computer applications
- User opinion of computer system response time
- User interface management system
- System engineering for improved response time

	human	machine	system	for	...	user
human	0	1	0	1	...	0
machine	1	0	0	1	...	0
system	0	0	0	1	...	2
for	1	1	1	0	...	0
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
user	0	0	2	0	...	0

Co-occurence Matrix

- A co-occurrence matrix is a **terms** × **terms** matrix which captures the number of times a term appears in the context of another term
- The context is defined as a window of k words around the terms
- Let us build a co-occurrence matrix for this toy corpus with $k = 2$
- This is also known as a **word** × **context** matrix
- You could choose the set of **words** and **contexts** to be same or different
- Each row (column) of the co-occurrence matrix gives a vectorial representation of the corresponding word (context)

Some (fixable) problems

- Stop words (a, the, for, etc.) are very frequent → these counts will be very high

	human	machine	system	for	...	user
human	0	1	0	1	...	0
machine	1	0	0	1	...	0
system	0	0	0	1	...	2
for	1	1	1	0	...	0
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
user	0	0	2	0	...	0

	human	machine	system	for	...	user
human	0	1	0	x	...	0
machine	1	0	0	x	...	0
system	0	0	0	x	...	2
for	x	x	x	x	...	x
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
user	0	0	2	x	...	0

Some (fixable) problems

- Stop words (a, the, for, etc.) are very frequent → these counts will be very high
- Solution 1: Ignore very frequent words
- Solution 2: Use a threshold t (say, t = 100) .

$$X_{ij} = min(count(w_i, c_j), t),$$

where w is word and c is context.

Some (fixable) problems

- Solution 3: Instead of $count(w, c)$ use $PMI(w, c)$

$$PMI(w, c) = \log \frac{p(c|w)}{p(c)}$$

$$= \log \frac{count(w, c) * N}{count(c) * count(w)}$$

N *is the total number of words*

	human	machine	system	for	...	user
human	0	2.944	0	2.25	...	0
machine	2.944	0	0	2.25	...	0
system	0	0	0	1.15	...	1.84
for	2.25	2.25	1.15	0	...	0
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
user	0	0	1.84	0	...	0

Some (fixable) problems

- Solution 3: Instead of $count(w, c)$ use $PMI(w, c)$

$$PMI(w, c) = \log \frac{p(c|w)}{p(c)}$$

$$= \log \frac{count(w, c) * N}{count(c) * count(w)}$$

N is the total number of words

- If $count(w, c) = 0$, $PMI(w, c) = -\infty$

Instead use,

$$PMI_0(w, c) = PMI(w, c) \quad if\ count(w, c) > 0$$
$$= 0 \quad otherwise$$

or

$$PPMI(w, c) = PMI(w, c) \quad if\ PMI(w, c) > 0$$
$$= 0 \quad otherwise$$

	human	machine	system	for	...	user
human	0	2.944	0	2.25	...	0
machine	2.944	0	0	2.25	...	0
system	0	0	0	1.15	...	1.84
for	2.25	2.25	1.15	0	...	0
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
user	0	0	1.84	0	...	0

Some (severe) problems

- Very high dimensional ($|V|$)
- Very sparse
- Grows with the size of the vocabulary
- **Solution:** Use dimensionality reduction (SVD)