upGrad

SEMANTIC PROCESSING

Syntactic processing involved the study of grammar to understand the meaning of sentences. However, there is no single encompassing rule book that defines grammar because language is always evolving. On the other hand, the text data that is generated is mostly grammatically incorrect.

However, we do not rely on grammar to understand the language. Remember when you were a child, you had not learnt about grammar but still could understand the language easily.
Example: 'I is a data scientist'. Although this is grammatically incorrect, we can understand it perfectly.

Semantic processing is probably the most challenging area in the field of NLP partly because the concept of 'meaning' itself is quite wide and because it is difficult to make machines understand the text the same way in which as we do–inferring the intent of a statement and the meanings of ambiguous words, dealing with synonyms, detecting sarcasm, etc.

We rely on the meaning of the words to understand the language; hence, we need semantic processing.You will learn about the following methods:

1. Knowledge Graphs
2. Distributional Semantics
3. Topic Modelling

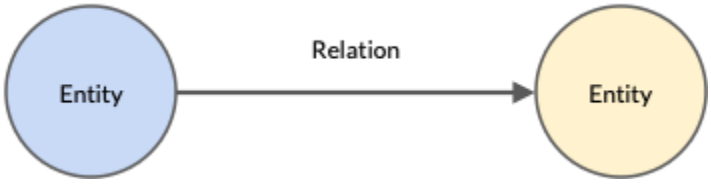
upGrad

Knowledge Graphs

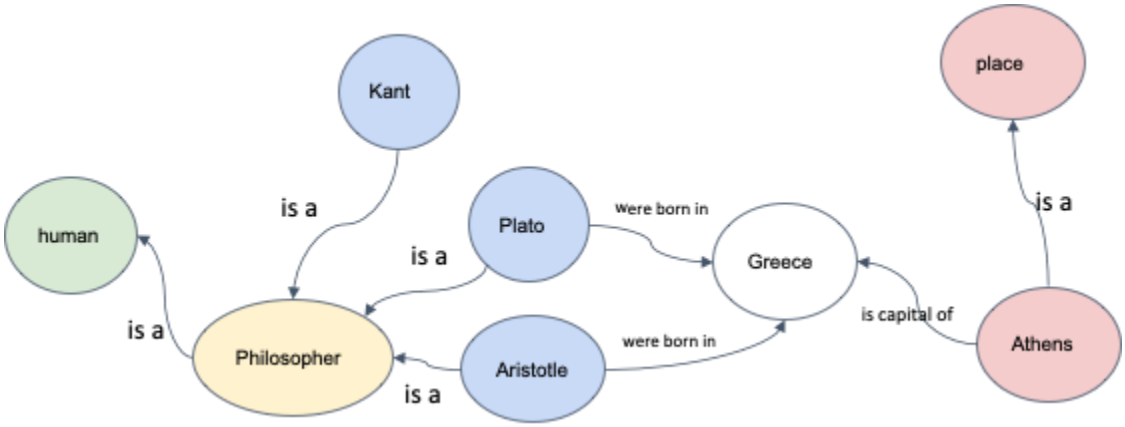
Introduction to Knowledge Graphs

A graph data structure consists of a finite set of vertices (also called nodes or points), where some pairs are connected through edges.

In the case of semantic processing, the text data is aggregated on the graph-based data structure. In the knowledge graph, entities are represented by nodes, where some pairs of the entities are related in some sense. These relations are represented by edges.



These graphs contain the semantic information of different entities. An example of a knowledge graph is given below.



You can see the different types of entities, which are human, occupation, names of people and names of places.

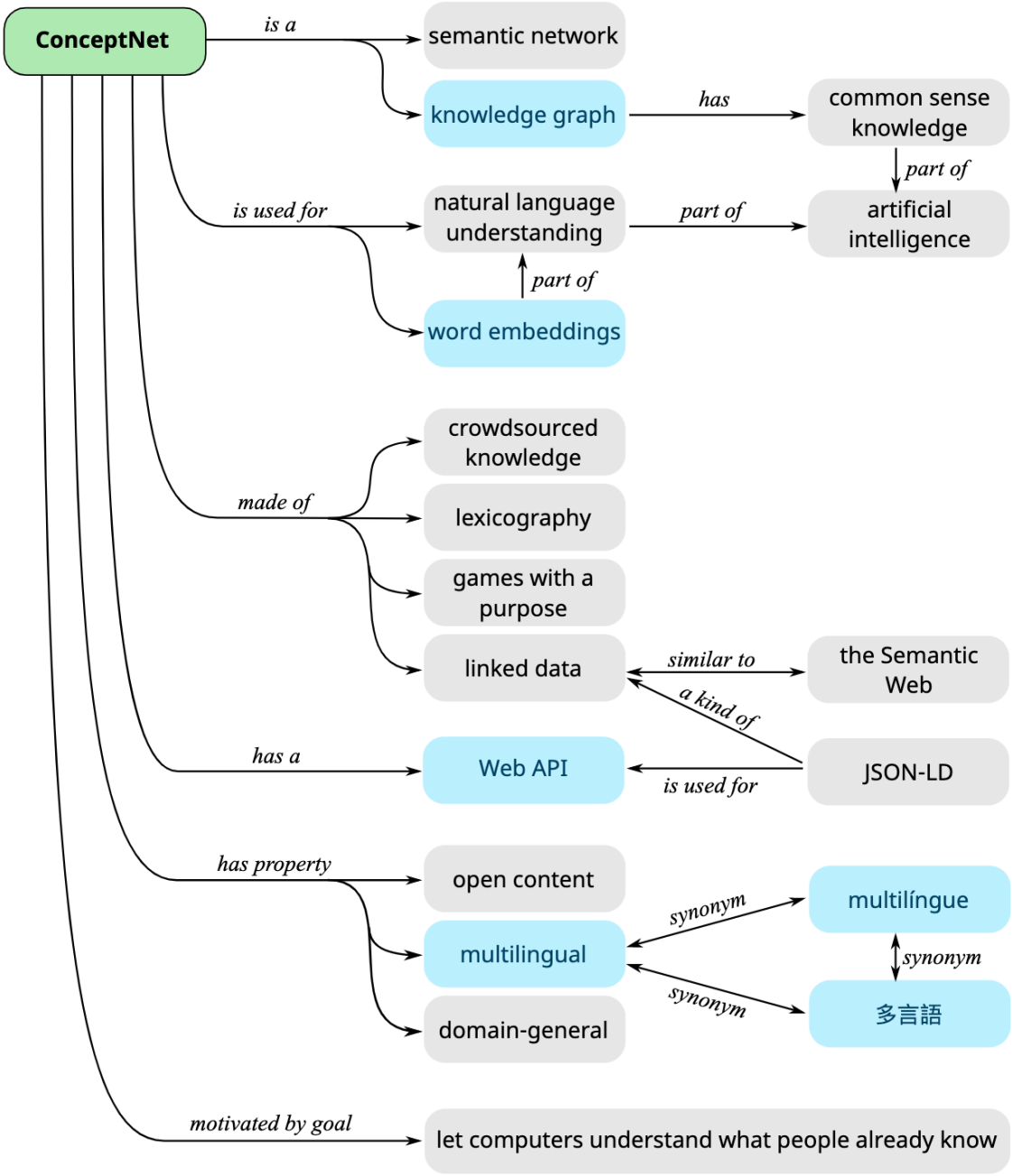
The different types of relationships are ‘is a’, ‘was born in’ and ‘is capital of’.

This is a simple representation of graphs however, graphs are generally very complex structures.

The different types of knowledge graphs are as follows:

- WordNet: This is a lexical database of semantic relations between words. It is developed by Princeton University.
- ConceptNet: This is a freely available semantic network that is designed to help computers understand the meanings of words that people use. It is developed by MIT.

The graph that describes the conceptnet is given below.



Both WordNet and ConceptNet are used for natural language understanding. At the end of this session, we will use WordNet to solve a use of Word Sense Disambiguation.

Another type of knowledge graph is UMLS.

- Unified Medical Language System (UMLS): It is a set of files and software that brings together many health and biomedical

vocabularies and standards to enable interoperability between computer systems.

Suppose you need to understand the text data that is related to the medical field. If you use WordNet or ConceptNet, you will have words that do not have relevance, and the results will be inaccurate. Hence, you require domain-specific knowledge graphs.

We know the famous knowledge graph called Google Search.

Although these are openly available knowledge graphs, many companies create their own knowledge graphs according to their company requirements.

- **Microsoft** uses knowledge graphs for the Bing search engine, LinkedIn data and academics.
- **Facebook** develops connections between people, events and ideas, focusing mainly on news, people and events related to the social network.
- **IBM** provides a framework for other companies and/or industries to develop internal knowledge graphs.
- **eBay** is currently developing a knowledge graph that functions to provide connections between users and the products present on the website.

upGrad

Wordnet

WordNet is a part of NLTK, and you will use it later in this module to identify the 'correct' sense of a word (i.e., for word sense disambiguation).

WordNet® is a large lexical database of English words developed by Princeton University. It can be accessed here: https://wordnet.princeton.edu/ .

Word to search for: bank Search WordNet

Display Options: (Select option to change) Change

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **bank** (sloping land (especially the slope beside a body of water)) *"they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents"*
- S: (n) depository financial institution, **bank**, banking concern, banking company (a financial institution that accepts deposits and channels the money into lending activities) *"he cashed a check at the bank"; "that bank holds the mortgage on my home"*
- S: (n) **bank** (a long ridge or pile) *"a huge bank of earth"*
- S: (n) **bank** (an arrangement of similar objects in a row or in tiers) *"he operated a bank of switches"*
- S: (n) **bank** (a supply or stock held in reserve for future use (especially in emergencies))
- S: (n) **bank** (the funds held by a gambling house or the dealer in some gambling games) *"he tried to break the bank at Monte Carlo"*
- S: (n) **bank**, cant, camber (a slope in the turn of a road or track; the outside is higher than the inside in order to reduce the effects of centrifugal force)
- S: (n) savings bank, coin bank, money box, **bank** (a container (usually with a slot in the top) for keeping money at home) *"the coin bank was empty"*
- S: (n) **bank**, bank building (a building in which the business of banking transacted) *"the bank is on the corner of Nassau and Witherspoon"*
- S: (n) **bank** (a flight maneuver; aircraft tips laterally about its longitudinal axis (especially in turning)) *"the plane went into a steep bank"*

Verb

- S: (v) **bank** (tip laterally) *"the pilot had to bank the aircraft"*
- S: (v) **bank** (enclose with a bank) *"bank roads"*
- S: (v) **bank** (do business with a bank or keep an account at a bank) *"Where do you bank in this town?"*
- S: (v) **bank** (act as the banker in a game or in gambling)
- S: (v) **bank** (be in the banking business)
- S: (v) deposit, **bank** (put into a bank account) *"She deposits her paycheck every month"*

In the diagram given above, each word sense of the word bank is grouped into its nouns and verbs. A set of all these senses is called a synset.
Each sense of the word has a gloss or meaning of the word and an example as in the dictionary.
For example, the first verb sense of the word has a gloss or a meaning as 'tip literally' and the example sentence as 'the pilot had to bank the aircraft'.
Similarly, each word sense contains a gloss and an example sentence.
If meanings are available in the dictionary as well, what makes WordNet unique?
Each of these senses of the words is related to other senses through some relations.

The types of relationship between different words can be grouped as follows:

1. **Synonym**: A relation between two similar concepts

   Example: Large is a synonym of big.

2. **Antonym**: A relation between two opposite concepts

   Example: Large is an antonym of big.

3. **Hypernym**: A relation between a concept and its superordinate

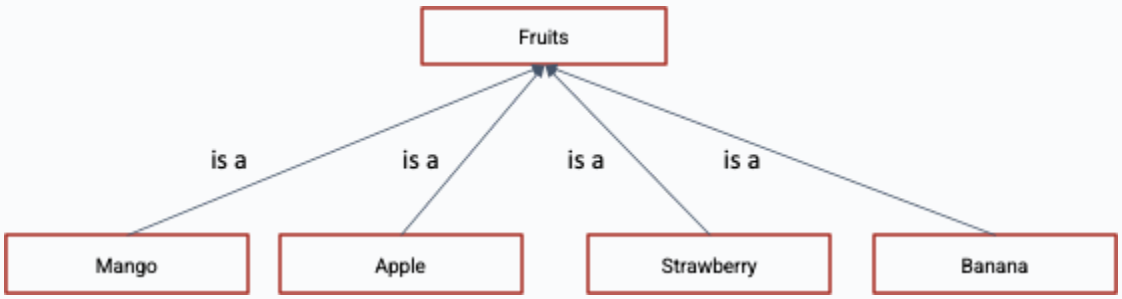
   A superordinate is all-encompassing entity

   Example: Fruits is the hypernym of mango.

4. **Hyponym**: A relation between a concept and its subordinate

   Example: Apple is the hyponym of fruits.

You can refer to the diagram given below to understand hyponyms and hypernyms. Any word that is connected with its hypernyms has an 'is a' relationship



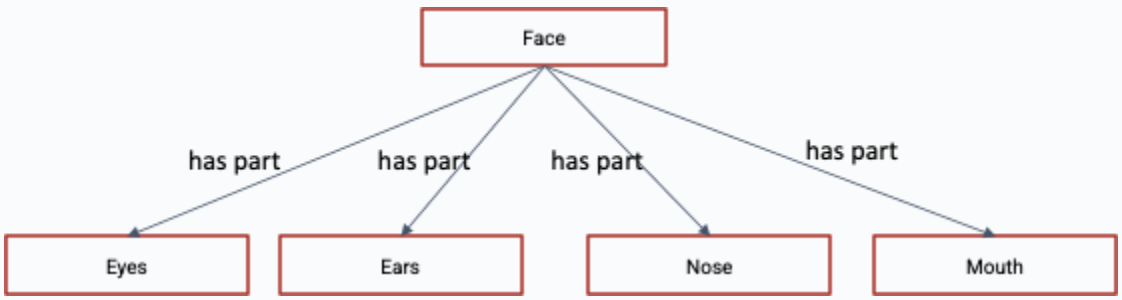
5. **Holonym:** A relation between whole and its parts

   Example: Face is the holonym of eyes.

6. **Meronym**: A relation between a part and its whole.

   Example: Eyes is the meronym of human body

You can refer to the diagram given below to gain an understanding of holonyms and meronyms. Any word is connected with its holonym by a 'has part' relationship. A term 'A' is said to be a holonym of a term 'B' if 'A' has a part of 'B' (and the term 'B' is said to be a meronym of the term 'A').



Homonymy is when a word has multiple meanings. For example, consider the word ‘pupil’. It can either refer to students or eye pupils depending on the context in which it is used.
Suppose you are searching for the word ‘pupil’. The search engines sometimes give data relevant to the context and sometimes give irrelevant data. Such things happen because the query may contain words whose meaning is ambiguous or those having several possible meanings.

To solve this ambiguity problem, the Lesk algorithm is used.
You can consider the definitions corresponding to the different senses of the ambiguous word and determine the definition that overlaps the maximum with the neighbouring words of the ambiguous word. The sense that has the maximum overlap with the surrounding words is then chosen as the ‘correct sense’.

“She booked the flight tickets to Delhi in advance”

Gloss 1	arrange for and reserve (something for someone else) in advance
Examples 1	"reserve me a seat on a flight"; "The agent booked tickets to the show for the whole family"; "please hold a table at Maxim's"
Gloss 2	a written work or composition that has been published (printed on pages bound together)
Examples 2	"I am reading a good book on economics”

In this example, you will notice that the gloss or meaning 1 has more overlap than gloss 2. Hence, we consider gloss 1 as the correct sense.

Although we have shown only two senses, a lesk algorithm parses through all the word senses and outputs the gloss that has the maximum overlap.

upGrad

Distributional Semantics

Introduction

From a technical point of view, grammar is a set of rules that need to be followed while constructing a sentence. However, several sentences are bound by the rules of grammar yet are gibberish. For example, take a look at the following sentence by Norm Chomsky who is an American linguist.

“Colorless green ideas sleep furiously.”

This sentence is structured correctly from a grammatical point of view. The noun (ideas) is described by an adjective (green) and followed by a verb (sleep). Yet, the meaning of the sentence is ambiguous.
Additionally, since language is an evolving concept, grammar keeps changing to match the needs of the era. Most of the communication done nowadays is via text or social media. Here, a more relaxed and informal tone of writing is used yet, it is still packed with information.

In both these cases, one needs to stop relying on syntactics and use semantic processing.

The dictionary definitions are not quite straightforward. Understanding a definition refers to understanding all the words within the definition of the word. However, we do not rely on a dictionary every time we don’t understand the meaning of a word.
You understand the meaning of the word from the overall context of the surrounding words. For example, let us assume that one does not know the meaning of the word ‘credit”.

After reading the sentence ‘The money was credited to my bank account’, one can easily infer that the word ‘credit’ is related to the exchange of currency. The words ‘money’ and ‘account’ set a context to the sentence that implies the predicted meaning. Through intelligent predictions such as this one, the meaning of words in a sentence becomes quite intuitive.

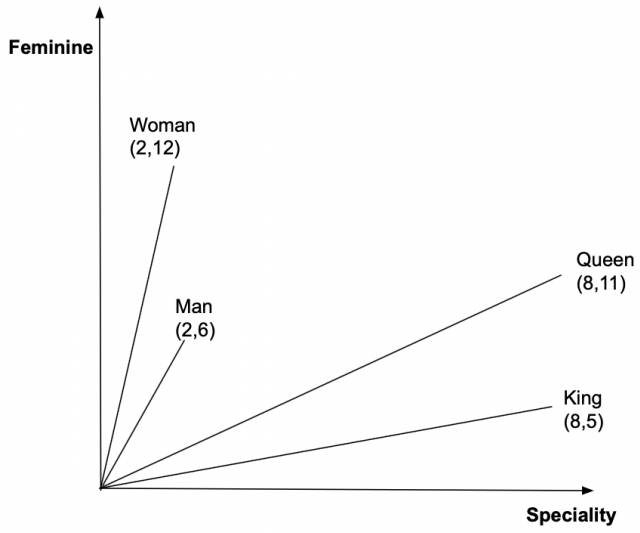
Hence, it was rightly said by the English linguist John Firth in 1957 -

"You shall know a word by the company it keeps."

Distributional semantics creates word vectors such that the word's meaning is captured from its context.

Geometric representation of meaning

Distributional semantics lets you capture the meaning of the word as vectors. The different aspects of meaning can be captured using geometry.
A vector consists of a magnitude and a direction.
Now, let's try to capture the meaning of a word using geometry. Let's consider two dimensions, speciality and femininity, to represent the meaning of the word. A classic example of this would be plotting the words King, Queen, Man and Woman on a plot of Speciality vs Femininity.
A King and Queen are equally special; however, a Queen is more feminine than a King. Similarly, a man is as special as a woman, but a woman is more feminine than a man. Interestingly, a man/woman is not as special as a king/queen. With all this in mind, the plot would look something like this.



Note that the meaning of the word is restricted to only two features; hence, this is not the complete picture. However, this gives an intuitive understanding of how the meaning of words are represented in geometry.

After converting words into vectors, the next step is to perform vector arithmetic! The movement from a woman to a queen is simply an increase in the speciality with relatively the same femininity.

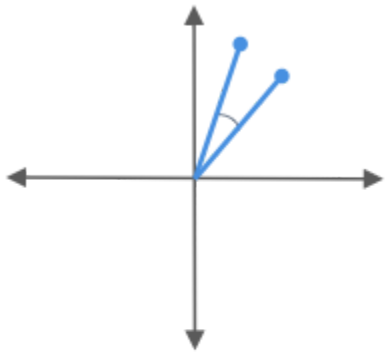
On the other hand, the movement from a man to a woman is an increase in femininity with relatively the same speciality.

These transformations correspond to vector addition and subtraction to capture the shift in terms of a vector. A major advantage of representing words as vectors is the ability to compare them through the vector operations depicted above.

Cosine Similarity

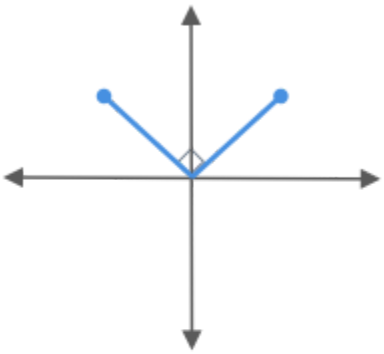
The fundamental way of measuring the similarity between two vectors is the cosine similarity.
According to the value of cos, one can infer several factors about a word:

1. For smaller a θ, the cosine value is close to 1. Such words are called synonymous words.



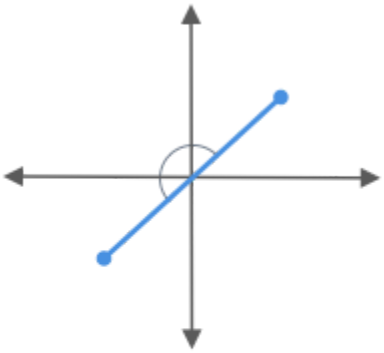
Synonymous words

2. $\theta = 90$ corresponds to unrelated words.

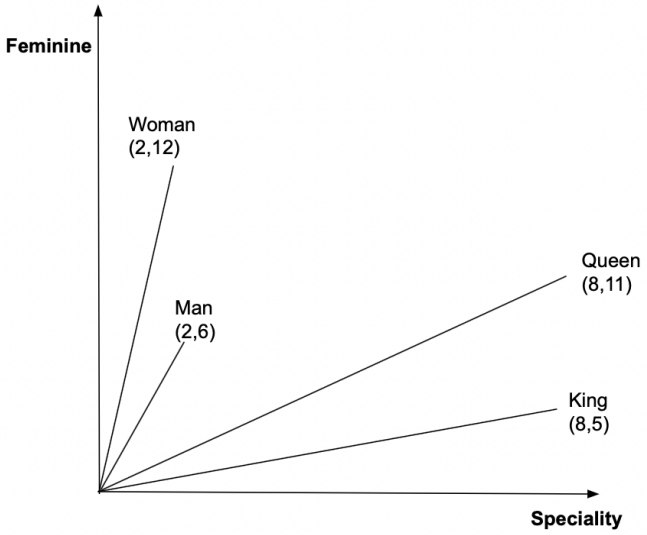


Unrelated words

3. For the θ values where the cosine value is negative, words are called antonyms.



Antonyms



$$S(\boldsymbol{Woman, Man}) = \cos(\boldsymbol{\theta}) = \frac{\boldsymbol{Woman.Man}}{||\boldsymbol{Woman}|| * ||\boldsymbol{Man}||}$$

$$\boldsymbol{Woman.Man} = 2 * 2 + 12 * 6 = 76$$

$$||\boldsymbol{Woman}||, = \sqrt{2^2 + 12^2} = \sqrt{4 + 144} = \sqrt{148}$$

$$||\boldsymbol{Man}|| = \sqrt{2^2 + 6^2} = \sqrt{4 + 36} = \sqrt{40}$$

$$S(\boldsymbol{Woman, Man}) = \frac{76}{\sqrt{148} * \sqrt{40}} = 0.98776$$

The cosine similarity of the words man and woman is 0.98776, which indicates that they are closely related. This is an accurate result.

upGrad

Bag of Words Representation

Bag of Words is a representation of text that describes the occurrence of words within a corpus of text, treating each word independently.

For example, the following text extract from *A Tale of Two Cities* authored by Charles Dickens can be converted into a bag-of-words representation:

"It was the best of times,
it was the worst of times.
It was the age of wisdom,
it was the age of foolishness.
It was the season of Light,
it was the season of Darkness.
It was the spring of hope,it was the winter of despair"

	best	worst	wisdom	foolishness	hope	despair	spring	winter	light	season	times	age
It was the best of times,	1	0	0	0	0	0	0	0	0	0	1	0
it was the worst of times.	0	1	0	0	0	0	0	0	0	0	1	0
It was the age of wisdom,	0	0	1	0	0	0	0	0	0	0	0	1
it was the age of foolishness.	0	0	0	1	0	0	0	0	0	0	0	1
It was the season of Light,	0	0	0	0	0	0	0	0	1	1	0	0
it was the season of Darkness.	0	0	0	0	0	0	0	0	0	1	0	0
It was the spring of hope,	0	0	0	0	1	0	1	0	0	0	0	0
it was the winter of despair.	0	0	0	0	0	1	0	1	0	0	0	0

This table represents the one-hot encoded vector for the text extract. If a word is present in the sentence, a value of '1' is assigned; otherwise, '0' is assigned. Note that attention is not given to the meaning, sequence or context of the words in this representation.

To understand the relationship between words, cosine similarity can be applied on the one-hot encoded representation of this text corpus. For example, on taking the words 'best' and 'worst' and applying cosine similarity, we get 0. This means that the words are completely unrelated.

	best	worst
It was the best of times,	1	0
it was the worst of times.	0	1
It was the age of wisdom,	0	0
it was the age of foolishness.	0	0
It was the season of Light,	0	0
it was the season of Darkness.	0	0
It was the spring of hope,	0	0
it was the winter of despair.	0	0

$$S = \cos(x, y) = \frac{x.y}{||x|| * ||y||}$$

$$S(best, worst) = \frac{0}{1 * 1} = 0$$

However, 'best' and 'worst' are antonyms, and their cosine similarity should be negative as you learnt earlier.
Similarly:

S(wisdom, foolishness) = 0

S(wisdom, winter) = 0

S(winter,light) = 0

S(winter, season) = 0

S(spring, season) = 0

The cosine similarity for unrelated words, related words and antonyms is zero. This proves that the bag-of-words representation is not the most accurate way to convert words into its vectors. The vectors captured by Bag of Words do not capture the meaning of words.

upGrad

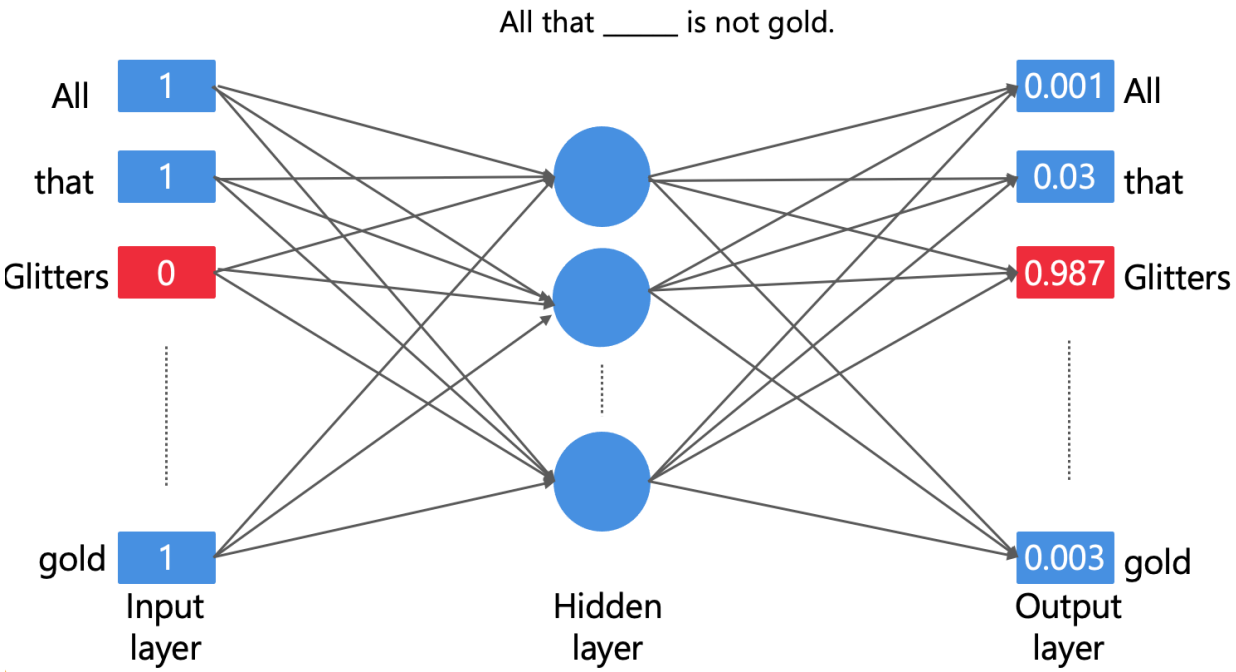
Word2Vec

Intuition of Word2Vec Models

In 1957, the English linguist John Firth said, “You shall know a word by the company it keeps”.

We, as humans, can decipher the meaning of an unknown word simply by taking a look at its surrounding words.
The steps involved in this are as follows:
Step 1: Make a neural network that **predicts a word**, given the nearby words.

Step 2: Wait for it to predict well enough. Train the neural network such that the probability of the glitters is the highest in the output layer, as the network is trying to predict glitters.



Step 3: Find out what it has learnt

Surprisingly, the output of the neural network is **not** what we are interested in, although this is what is generally extracted from neural networks. Instead, we are interested in the **weight matrices** which contain the semantic information, known as word embeddings.

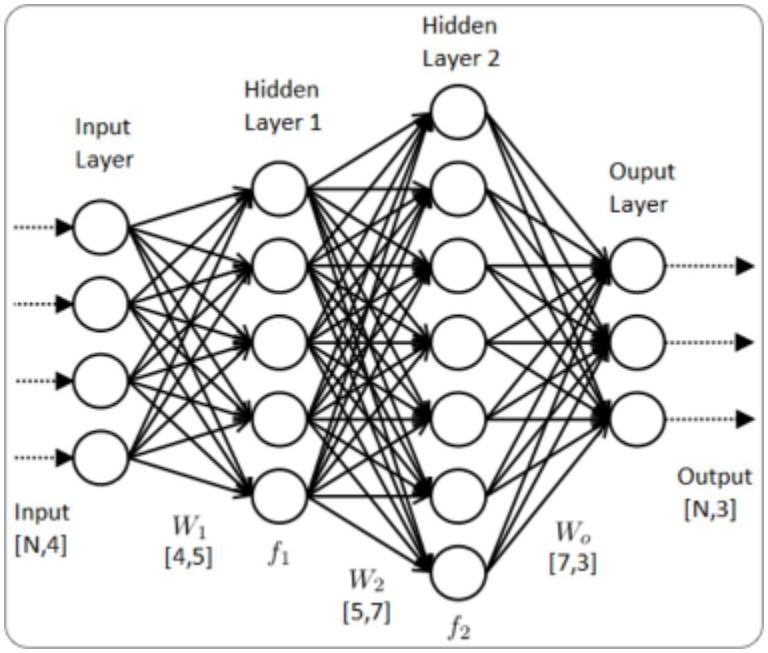
Do keep in mind that the word vectors are not found in the output of neural networks but in the weight matrices!

To summarise, if a machine is able to predict the words from its surroundings, it has captured the meaning of the word. In the next segments, you will learn about two different algorithms of the Word2Vec model, which are Skipgram model and the Continuous Bag Of Words (CBOW) model.

Forward Pass

A general representation of the Input Data Matrix is given below wherein n represents the number of features and m represents the number of samples. In the Iris data set, you saw that the number of features n is 4–sepal length, sepal width, petal length and petal width.

We designed the neural network that had 4 input neurons, as we had 4 features in the Iris data set and 3 neurons in the output layer, as we had 3 classes, which are setosa, virginica and versicolor. The number of neurons in the hidden layer is arbitrarily decided.



The different components in a fully connected neural network are as follows:

- Input Layer
- Weight Matrix (W_1)
- Hidden Layer 1
- Activation function of layer 1 (f_1)
- Weight Matrix (W_2)
- Hidden Layer 2
- Activation function of layer 2 (f_2)
- Weight Matrix (W_o)
- Output Layer
- Activation function of output layer (f_0)

The dimensions of the weight matrix are deduced from the numbers of neurons in the previous layer and in the next layer.
Number of rows of the weight matrix = Number of neurons in the previous layer
Number of columns of the weight matrix = Number of neurons in the next layer

You learnt how one input sample goes through the different layers of neural network, and this is given below:

- Input to the first layer: $1_1^{in} =< X, W1 >+ b1$ I_{1in}=<X,W_1>+b_1
- Output of the first layer:I_{1out}=f_1(I_{1in})
- Input to the second layer: I_{2in}=<I_{1out},W_2>+b_2
- Output of the second layer:I_{2out}=f_2(I_{2in})

NOTE: The notations used in the formulae may vary. The dot product between two vectors X, Y can be denoted as <X,Y> or Y_TX.

In many cases, you will encounter W_TX+b instead of <X,W>+b as one of the notations. Do not be alarmed; they both denote the same thing.

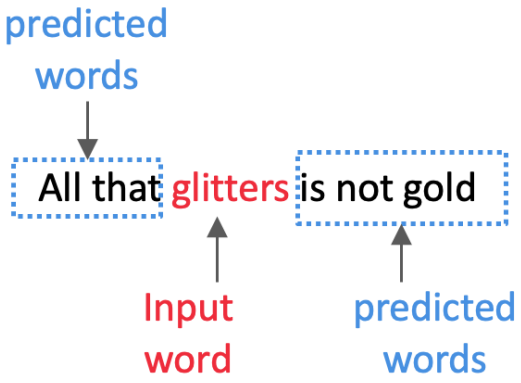
- Input to the first layer: I_in^1=<X,W_1>+b_1
- Output of the first layer: I_out^1=f_1 (I_in^1)
- Input to the second layer: I_in^2 =< I_out^1,W2>+ b_2
- Output of the second layer: I_out^2 =f2 (I_in^2)
- Input to the output layer: I_in^3 =< I_out^2,W3>+ b_3
- Output of the output layer: I_out^3 =f3 (I_in^3)

NOTE: The notations used in the formulae may vary. The dot product between two vectors X, Y can be denoted as <X, Y> or Y^TX.
In many cases, you will encounter W^TX +b instead of <X,W>+b as one of the notations. Do not be alarmed; they both denote the same thing.

upGrad

Input Data for CBOW

The two different approaches for Word2Vec are Skipgram model and Continuous Bag Of Words (CBOW) model
The Skipgram model predicts words within a certain range before and after the input word.



Continuous Bag of Words predicts the middle word based on the surrounding words.

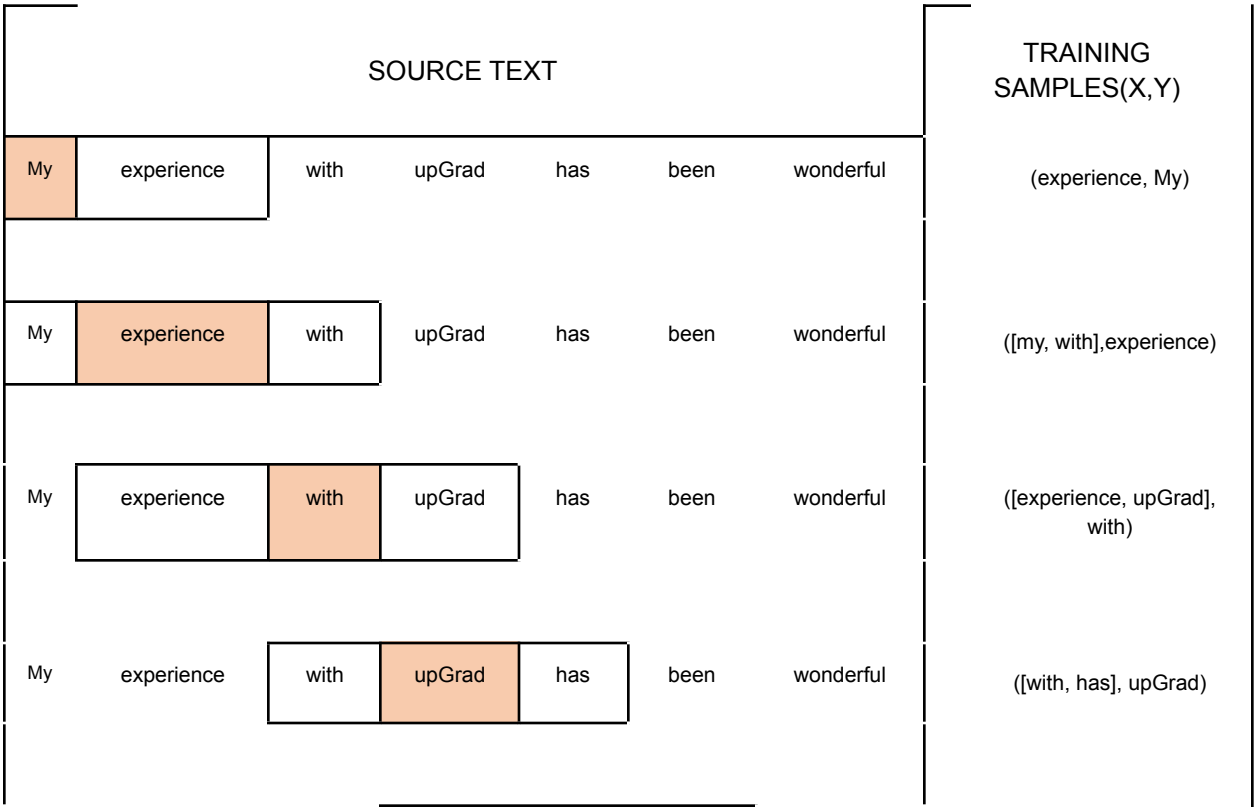
upGrad

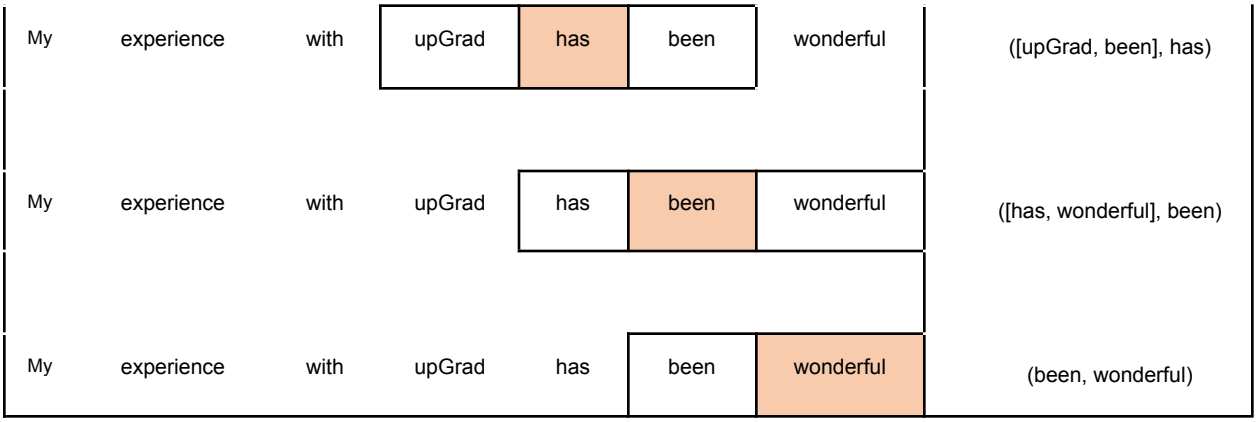
Continuous Bag of Words

Training Input Data

Let us assume that we have only one sentence in our corpus, which is '*My experience with upGrad has been wonderful*'. The vocabulary size is the number of unique words in the corpus, which is 7 in our case.
The context size determines the number of words that are present before and after a given word would be included as the context words of the given word. This is a parameter that we need to decide before creating training data. For the sake of simplicity, we have considered the context size to be 1.

The training data for the CBOW model for our sentence with a context size of 1 looks like this.





We can represent our training data as follows.

TRAINING SAMPLES (X,Y)	X	Y
(experience, My)	[experience]	My
([my, with],experience)	[my, with]	experience
([experience, upGrad], with)	[experience, upGrad]	with
([with, has], upGrad)	[with, has]	upGrad
([upGrad, been], has)	[upGrad, been]	has
([has, wonderful], been)	[has, wonderful]	been
([been], wonderful)	[been]	wonderful

We have created the training samples that need to be fed into a neural network, but neural networks do not take words as inputs.

One Hot Encoding is used to convert words into a numeric format. The words can be arranged in alphabetical order or based on frequency or any other heuristic. One hot encoding of the words in our example can be represented as follows.

	been	experience	Has	My	upGrad	with	wonderful
been	1	0	0	0	0	0	0
experience	0	1	0	0	0	0	0
has	0	0	1	0	0	0	0
My	0	0	0	1	0	0	0
upGrad	0	0	0	0	1	0	0
with	0	0	0	0	0	1	0
wonderful	0	0	0	0	0	0	1

From this table, the one hot encoding of ‘experience’ is as follows. The position of the ‘1’ is in the second position as the mapping has been done according to the table above.

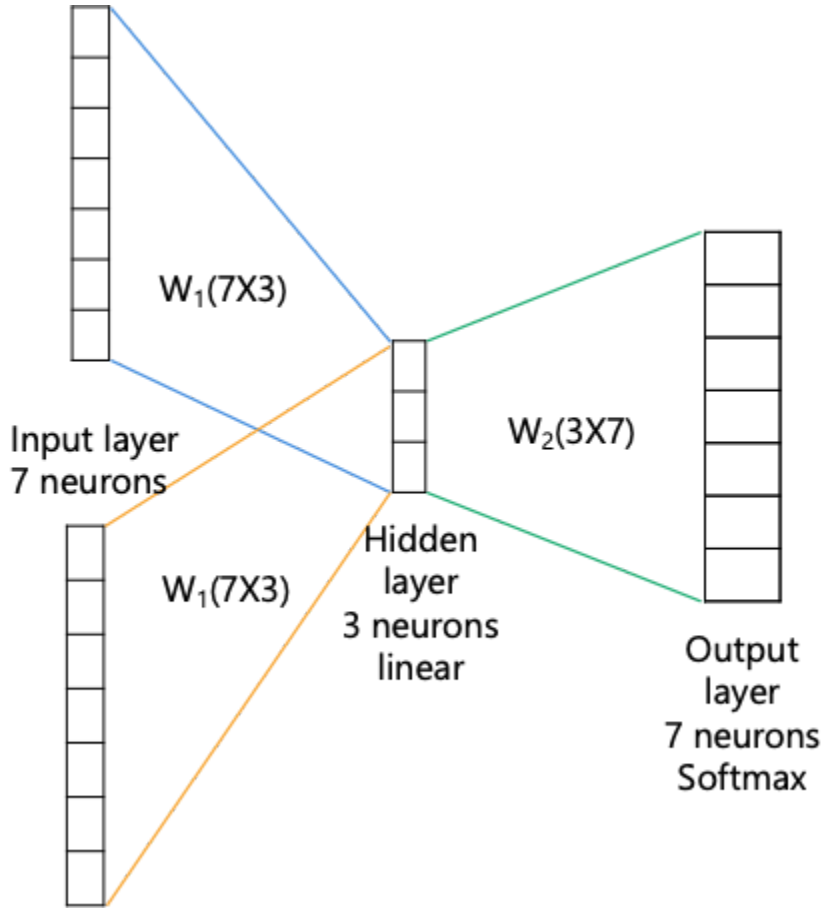
0	1	0	0	0	0	0

upGrad

Training of CBOW model

The architecture of CBOW is shown below. The neural network is a shallow neural network with one hidden layer.

The input layer and the output layer have 7 neurons each, which is equal to the size of vocabulary. You decide the number of neurons in the hidden layer according to the number of dimensions that you want in your word embeddings. In our case, the hidden layer has a linear activation function, whereas the output layer has the softmax activation function.

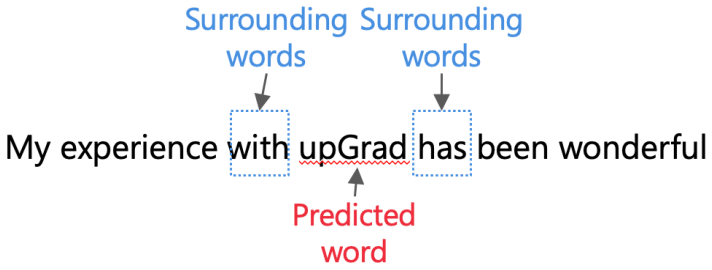


For a training pair ([with, has], upGrad)

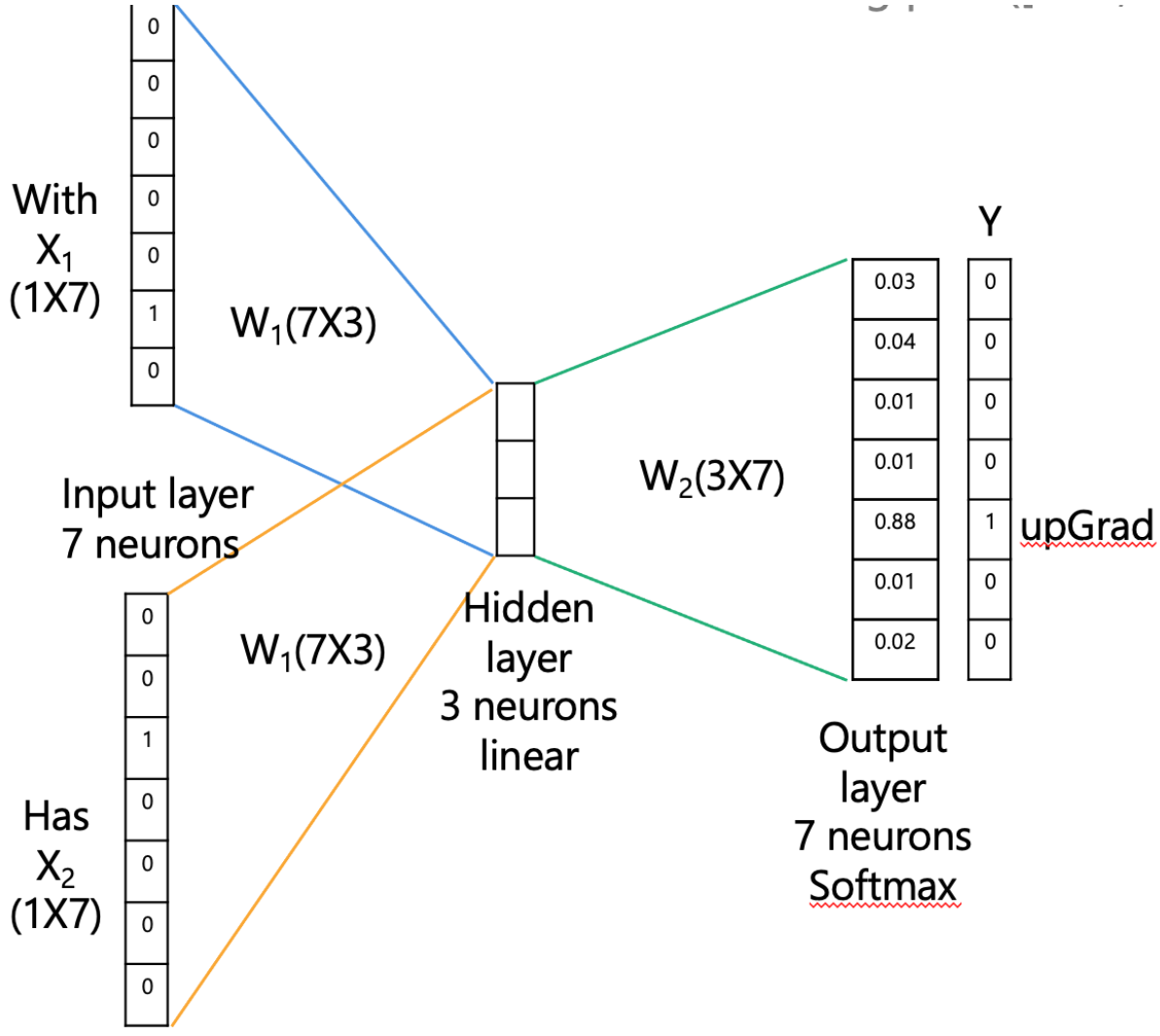
input X = [with,has]

output Y = upGrad

The task of the neural network is to predict the output, given the following input.



One hot encoding of the words 'with' and 'has' will be the inputs of the neural network, and the output of the network should be upGrad.



The OHE of the inputs flows through the network in the forward pass in the following manner:

$$Output\ of\ hidden\ layer\ =\ l_1^{\ out}\ =\ (<\times_1, W_1>+<\times_2, W_1>)/2$$

$l_1^{out}\ is\ (1X3)\ vector$

$Output\ of\ last\ layer\ =\ l_0^{out} = f(< l_1^{out}, W_2 >)$

$l_0^{out}\ is\ (1X7)\ vector$

Note: The output of the first hidden layer is calculated by taking the average of the input vectors as specified in the https://cs224d.stanford.edu/lecture_notes/notes1.pdf https://web.stanford.edu/~jurafsky/slp3/6.pdf*. Other heuristics can also be used.*

The output of the neural network will not give a one hot encoding vector of upGrad. As the output layer has the activation function of softmax, we will train the network such that the probability of the target word is high in the output.
As shown in the diagram given above, the probability is the highest in the fifth position of upGrad.

The neural network uses a backpropagation algorithm that compares the actual output and predicted output and tries to minimise the loss.Now that our training is complete, we can predict a word given its context words. Remember weight matrices

Weight Matrices

You learnt how to create input training samples, architecture, forward pass and backpropagation for the CBOW model. However, this is not what we are actually interested in. We are interested in what the model has learnt and is stored in weight matrices.
Word embeddings are stored in weight matrices. W1 is considered as our embedding matrix.
If you multiply the one hot encoding form of the word with the matrix, you will obtain word embedding of that word. The OHE of upGrad is as follows:

0	0	0	0	1	0	0

upGrad

W1 (the weight matrix of the hidden layer) =

0.05	0.67	0.56
0.45	0.56	0.34
0.56	0.53	0.34
0.23	0.45	0.23
0.56	0.21	0.67
0.56	0.45	0.34
0.56	0.67	0.32

The word embedding or word vector of upGrad is as follows.

0.56	0.21	0.67

You will observe that every row in the weight matrix W_1 is a word embedding. Rather than multiplying the OHE by a weight matrix, we can directly pick the row according to the position of 1 in OHE and use W_1 as a look-up matrix.
The word embedding of a word not only captures the meaning of the word but also reduces the dimensions of the word vector.
The dimension of word vector (OHE) was 7, which was reduced to 3 in the word embedding.
The neural network learns by minimising the loss function and maximising the dot product between the input word vector and the output word vector. If we apply this logic, we can conclude weight matrices are word embedding matrices.

Note: W_1 and W_2 are not transposes of each other, although we have considered W_1 to be our word embedding matrix in our calculations.

Skipgram Model

In the case of Skip gram, we try to predict the context words given the target word. So, the training data for the skipgram model for the context size 1 looks like this.

SOURCE TEXT							TRAINING SAMPLES
My	**experience**	with	upGrad	has	been	wonderful	(My, experience)
My	experience	**with**	upGrad	has	been	wonderful	(experience,[my, with])
My	**experience**	with	**upGrad**	has	been	wonderful	(with,[experience, upGrad])
My	experience	**with**	upGrad	**has**	been	wonderful	(upGrad,[with, has])
My	experience	with	**upGrad**	has	**been**	wonderful	(has, [upGrad, been])
My	experience	with	upGrad	**has**	been	**wonderful**	(been, [has, wonderful])
My	experience	with	upGrad	has	**been**	wonderful	(wonderful, [been])

The input of CBOW becomes the output of skipgram and vice versa.
An example is given below.

Model	**X(input)**	**Y(output)**
CBOW	[with, has]	upGrad
Skipgram	upGrad	[with, has]

Remember that these training samples need to be converted to OHE before feeding into the neural network.

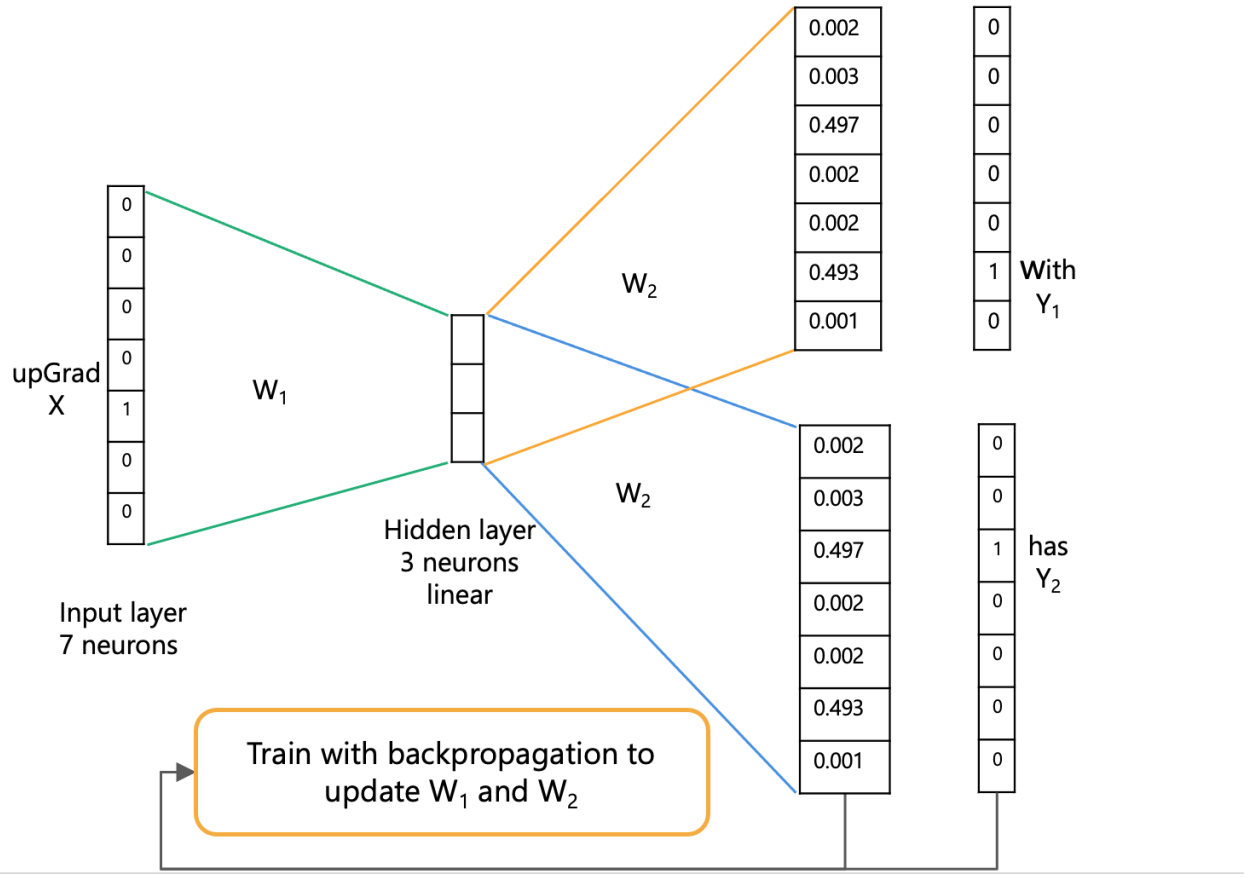
The architecture of skipgram is the same as the CBOW model.
The neural network is a shallow neural network with 1 hidden layer.
The input layer and the output layer have 7 neurons, and this is equal to the size of vocabulary.
The hidden layer has a linear activation function, whereas the output layer has the softmax activation function.

The difference appears when you perform forward pass and backpropagation.
The main difference is in the output of the first layer step, which appears when you take the average of two input words in CBOW, but it is not necessary to consider the average of the input words, as only one input word is present in the skipgram model.

The output of the first layer in CBOW is $l_1^{out} = (< \times_1, W_1 > + < \times_2, W_1 >)/2$

On the contrary, the output of the first layer in skipgram is $l_1^{out} = < \times_1, W_1 >$

The differentiating factor in the backpropagation step is that the error needs to be calculated for the two output words [with,has] in the skipgram model as shown below.



After training the skipgram model, the weight matrix W_1 is a word embedding matrix, which is the same as the CBOW model.

We use gensim library to create the skipgram and CBOW models.

upGrad

Architecture of Neural Networks for Binary Classification

Binary Classification is heuristic that is required to decrease the computational load.

Previously, our text corpus had a vocabulary of size 7. In reality, the text corpus has a huge vocabulary size of approximately half a million.

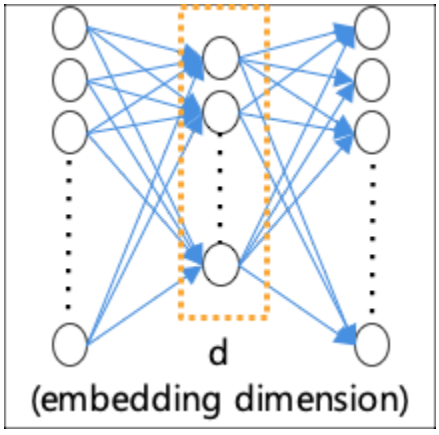
You will notice the number of unique words in different languages in the following table.

Language	Approximate Number of Words	Dictionary
English	4,70,000	Webster
Dutch	4,00,000	Woordenboek
Tamil	3,80,000	Sorkuvai
Chinese	3,78,103	Hanyu Da Cidian

For a text corpus that has half a million vocabulary size, the output layer will have to compute the softmax function for half a million neurons.

Let's consider an example sentence: 'An aardvark ate a zyzzyva.' One Hot Encoding of this sentence is as follows:

The neural network for embedding the dimension 'd' looks like this.



The vocabulary V is approximately half million; hence, the shape of W1, the word embedding matrix, is (V,d), which is huge.
In the original CBOW model, we had to predict a centre word given one or more context words. In the skip gram model, we had to predict context words given a centre word. In both cases, it was a classification problem with half a million classes.In binary classification, if given a pair of words, it predicts if they are in each other's context.
The input to the input layer will be concatenation/addition/other heuristic of the OHE of a pair of words. The output neuron will predict 1 if the pairs occur in each other's context or will output 0 if the pair does not occur in each other's context. If we consider the following example.
'*The quick brown fox jumps over the lazy dog'*
For the context size of 2:
The pair (quick, fox) will give an output of 1.
The pair (quick, jumps) will give an output of 0.

Training Input Data for Negative Sampling

Corpus:
"It was the best of times,
it was the worst of times.
It was the age of wisdom,
it was the age of foolishness.
It was the season of Light,
it was the season of Darkness.
It was the spring of hope,
it was the winter of despair."

Now, we will make a pair of the samples that occur in the context size of 2. We need to make positive and negative samples.
The positive samples are shown below.

X	Y
('best', 'times')	1

(‘worst’, ‘times’)	1
(‘age’, ’wisdom’)	1
(‘age, ’foolishness’)	1
(‘season’,’light’)	1
(‘season’, ‘darkness’)	1
(‘spring’, ‘hope’)	1
(‘winter’, ‘despair’)	1

For a classifier, we require negative samples, as it cannot classify if there is only one label. Now, you will learn how to create negative samples.

The steps involved in negative sampling are given below:

- When parsing the corpus, pick a word as a context word.
- Pick a **random, infrequent** word from the vocabulary as the target word.
- Label this pair as a negative sample.

You will obtain the following table as negative samples.

X	Y
(‘best’, ‘spring’)	0
(‘times’, ’hope’)	0
(‘worst’, ’spring’)	0
(‘times’, ’despair’)	0
(‘age’, ‘winter’)	0
(‘best’, ‘despair’)	0
(‘wisdom’, ’despair’)	0
(‘worst’, ’hope’)	0

upGrad

Topic Modelling

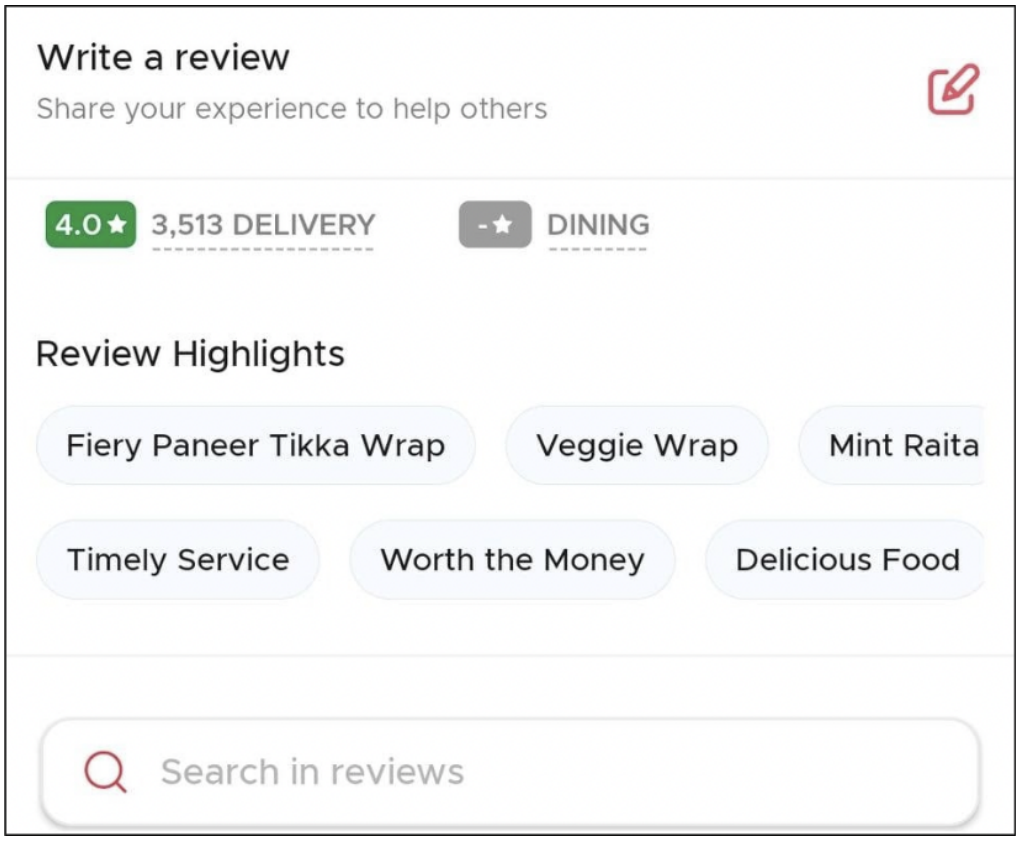
Introduction

Suppose you are a product manager at Amazon and want to understand the features of a recently released product (say, Amazon Alexa) that customers are discussing in their reviews. Say you are able to identify that 50% of the customers are talking about the hardware, 30% are talking about features related to music, and 20% are discussing the packaging of the product. This information would be pretty useful for you.

Similarly, imagine that you have a large corpus of scientific documents (such as research papers), and you want to build a search engine for this corpus. Suppose you can infer that a particular paper talks about ‘topics’ such as diabetes, cardiac arrest and obesity. With this information, conducting a topic-specific search will become easier.

Suppose various pieces of text are being sent from different sources. These are unstructured and unlabeled. Topic modelling can help determine the major themes or labels of this text.

An ideal example of this is customer review data. For example, reviews for a restaurant can vary according to food items, length, sentiment, etc. The job of a topic modelling algorithm is to skim through each review and figure out the major themes and keywords. These can be later displayed under a ‘Review Highlight’ section as shown in the image given below.



If a customer visits the site, it becomes easier for them to find relevant reviews. Suppose you are on a budget and want to know if a certain dish is affordable. ‘Worth the Money’ would be the ideal label in this case or you are interested in ordering the ‘Fiery Paneer Tikka Wrap’; however, you are not sure if it will be any good. You can easily figure this out by clicking on the corresponding food label.

How does topic modelling work? Before understanding various algorithms, you will learn how a human would infer a topic from a given sentence.

upGrad

Suppose you are given this sentence:

Croatia fought hard before succumbing to France's deadly attack, lost the finals 2 goals to 4.

On reading the first part of the sentence 'Croatia fought hard before succumbing to France's deadly attack', one would infer that the context of the sentence is war. This inference is primarily due to the usage of 'fought hard' and 'deadly attack'. However, on reading the next part 'lost the finals 2 goals to 4', the context changes to sports, specifically soccer. This is due to the usage of the words 'finals' and 'goals'.

Although the topic of a sentence that is related to soccer or some sport is not explicitly mentioned in the sentence, we were able to understand that this was the latent topic.

The goal of this exercise was to impart the same thinking in a machine learning algorithm. This domain of computer science is known as 'information retrieval' and involves the identification of dominating themes from a sample of text.
Let's take a look at a specific example to understand the working of topic modelling.

Topic 1	Model, data, validation, neural networks, supervised, **machine learning**, etc.
Topic 2	**Cancer**, hospitals, medicine, X-ray, disease, etc.
Topic 3	Chef, restaurants, chocolate, spicy, food, etc.

Suppose the topic modelling algorithm has analysed a corpus of text and clustered groups of words based on their context in a document. As you can see, the words have been divided into major themes. Topic 1 is 'Data Science', topic 2 is 'Medicine' and topic 3 is about 'Culinary Arts'.
If you are given a sentence such as this one

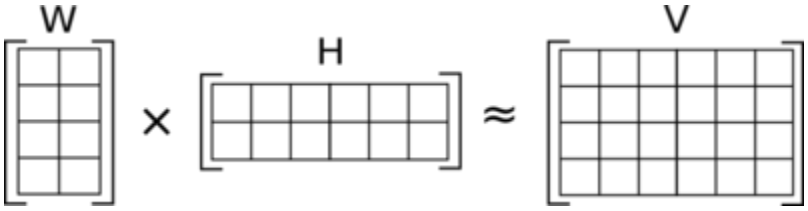
'Machine learning can aid and improve early detection of breast cancer.'

Which topics from the table given above does this sentence fall under? The words ‘machine learning’ belong to Topic 1, and the words ‘breast cancer’ belong to Topic 3. This means that the sentence contains both topics.

Non-Negative Matrix Factorization

The three types of algorithms in topic modelling are as follows:

1. Non-negative matrix factorisation
2. Latent dirichlet allocation
3. Latent semantic analysis



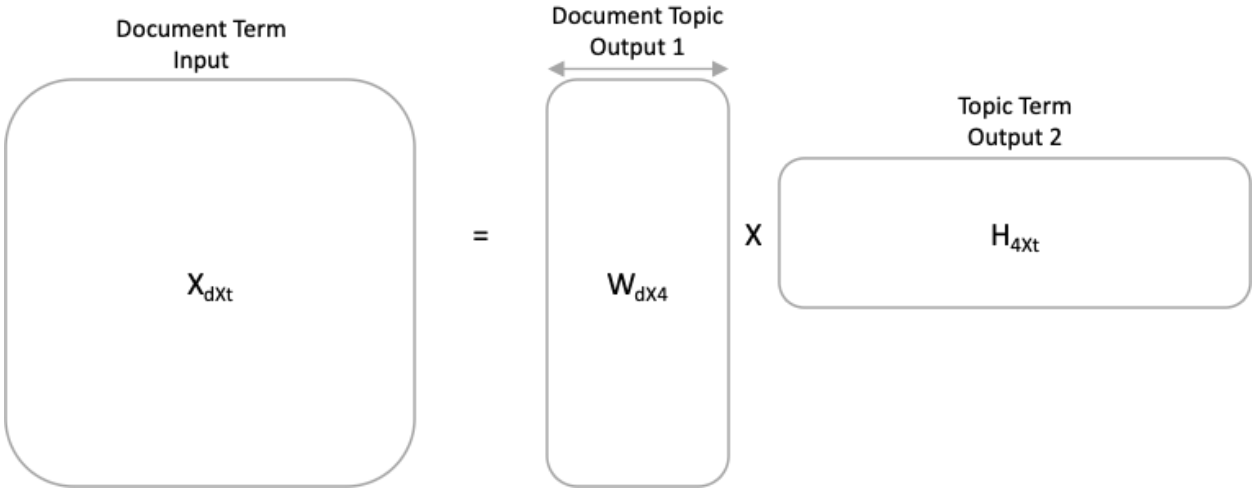
NMF approximates a matrix X as a product of two matrices W, H such that X ~ W * H where W >=0 and H>=0.
This is done by minimising the square difference between X and W * H through a formula known as the Frobenius Norm. This is an optimisation problem, as the equation tries to get as close to the minimum value as possible.

$$\min||X - WH||^2 such\ that\ W \geq 0\ and\ H \geq 0$$

To calculate the frobenius norm, we calculate the sum of the square of all the values of the X-WH matrix.
The formula for calculating the frobenius norm for matrix A is as follows.

$$||A||_F \equiv \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$$

The dimensions of W, H depend on the number of topics. The original matrix X is of the size (number of documents X number of terms). This is broken down into W of the size (number of documents X number of topics) and H of the size (number of topics X number of terms).



You need to keep in mind that NMF or any other topic modelling algorithm does not give you the topics explicitly. Topics are neither coherent or self-contained nor meaningful ideas or concepts. A topic is a bag of words. It is the responsibility of data scientists to assign topics using domain knowledge and logic.