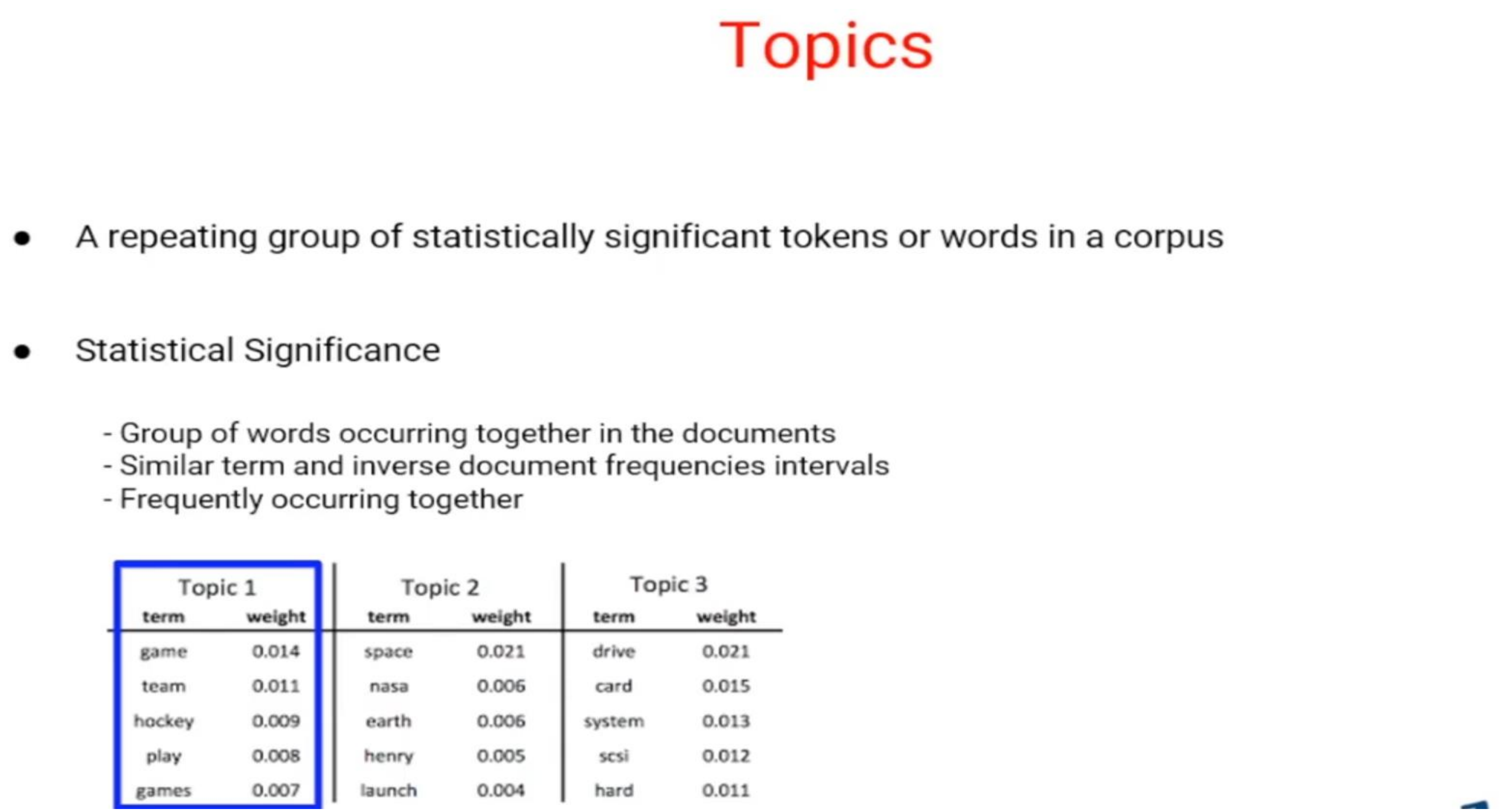
Word Embedding NLP Training

By – Amit Dhomne

Topic Modeling



Topics

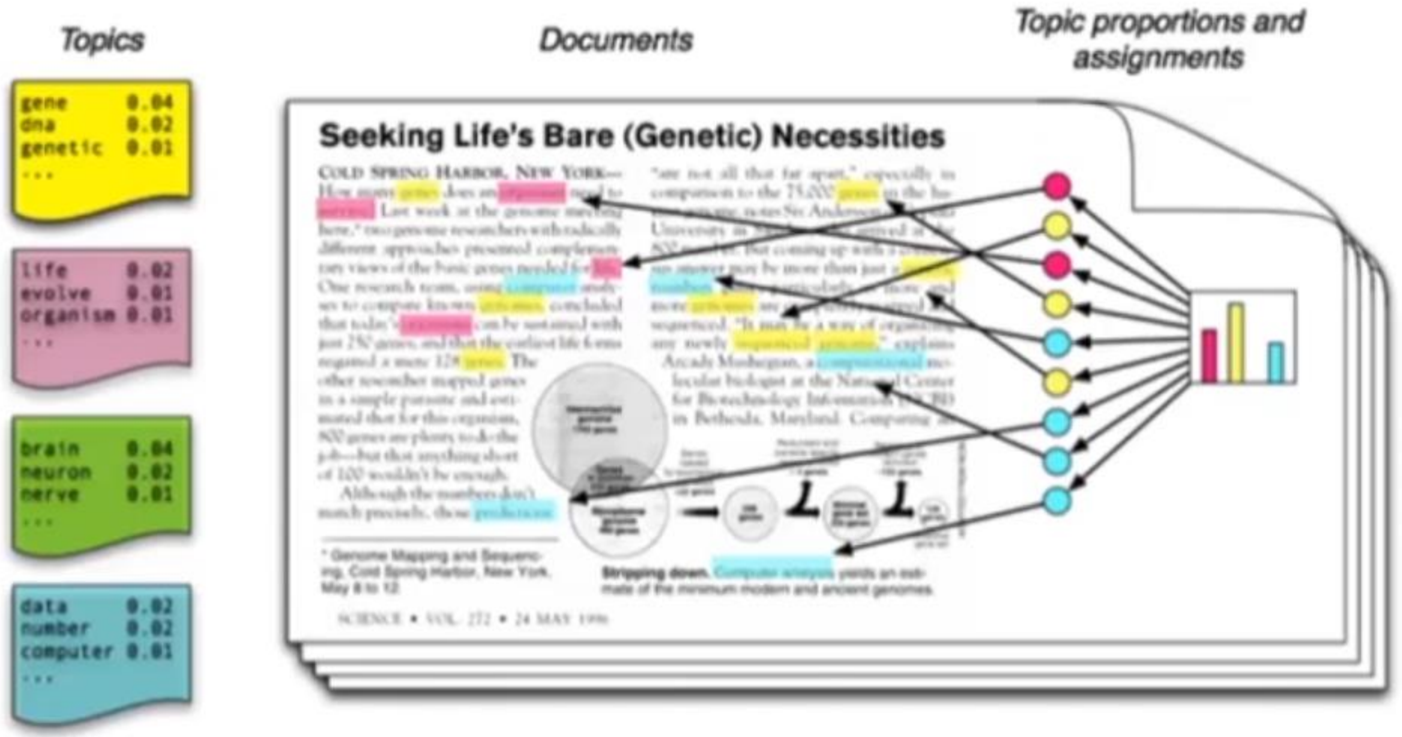
- A repeating group of statistically significant tokens or words in a corpus

- Statistical Significance

  - Group of words occurring together in the documents
  - Similar term and inverse document frequencies intervals
  - Frequently occurring together

Topic 1		Topic 2		Topic 3	
term	**weight**	**term**	**weight**	**term**	**weight**
game	0.014	space	0.021	drive	0.021
team	0.011	nasa	0.006	card	0.015
hockey	0.009	earth	0.006	system	0.013
play	0.008	henry	0.005	scsi	0.012
games	0.007	launch	0.004	hard	0.011

Topic Modelling



- Process to find the topics form documents in an unsupervised manner
- Text mining approach to find recurring patterns in the text documents

What is Topic Modeling?

1 .Topic modeling is a machine learning technique that automatically analyzes text data to determine cluster words for a set of documents. This is known as 'unsupervised' machine learning because it doesn't require a predefined list of tags or training data that's been previously classified by humans.

2 .Since topic modeling doesn't require training, it's a quick and easy way to start analyzing your data. However, you can't guarantee you'll receive accurate results, which is why many businesses opt to invest time training a topic classification model.

3 .Since topic modeling doesn't require training, it's a quick and easy way to start analyzing your data. However, you can't guarantee you'll receive accurate results, which is why many businesses opt to invest time training a topic classification model.

4 .Since topic classification models require training, they're known as 'supervised' machine learning techniques. What does that mean? Well, as opposed to text modeling, topic classification needs to know the topics of a set of texts before analyzing them. Using these topics, data is tagged manually so that a topic classifier can learn and later make predictions by itself.

5.For example, let's say you're a software company that's released a new data analysis feature, and you want to analyze what customers are saying about it. You'd first need to create a list of tags (topics) that are relevant to the new feature e.g. *Data Analysis, Features, User Experience*, then you'd need to use data samples to teach your topic classifier how to tag each text using these predefined topic tags.

6 .Although topic classification involves extra legwork, this topic analysis technique delivers more accurate results than unsupervised techniques, which means you'll get more valuable insights that help you make better, data-based decisions. You could say that unsupervised techniques are a short-term or quick-fix solution, while supervised techniques are more of a long-term solution that will help your business grow.

Examples of Topic Modeling and Topic Classification

Let’s take a look at some examples, to help you better understand the differences between automatic **topic modeling** and **topic classification**.

Topic modeling could be used to identify the topics of a set of customer reviews by detecting patterns and recurring words. Let’s take a look at how an ‘unsupervised’ technique would group the below review for Eventbrite, for example:
“The nice thing about Eventbrite is that it’s free to use as long as you’re not charging for the event. There is a fee if you are charging for the event – 2.5% plus a $0.99 transaction fee.”

By identifying words and expressions such as *free to use*, *fee*, *charging*, *2.5% plus 99 cents transaction fee*, topic modeling can group this review with other reviews that talk about similar things (these may or may not be about pricing).

A topic classification model could also be used to determine what customers are talking about in customer reviews, open-ended survey responses, and on social media, to name just a few. However, these supervised techniques use a different approach. Rather than inferring what similarity cluster the review belongs to, classification models are able to automatically label a review with predefined topic tags. Take this review about SurveyMonkey, for example:

“We have the gold level plan and use it for everything, love the features! It is one of the best bang for buck possible.”
A topic classification model that’s been trained to understand these expressions (gold level plan, love the features, and best bang for buck) would be able to tag this review as topics *Features* and *Price*.
In short, topic modeling algorithms churn out collections of expressions and words that it thinks are related, leaving you to figure out what these relations mean, while topic classification delivers neatly packaged topics, with labels such as *Price,* and *Features,* eliminating any guesswork.

Text Classification vs Text Modeling

Text Classification is a form of supervised learning, hence the set of possible classes are known/defined in advance, and won't change.

Text classification often involves mutually-exclusive classes -- think of these as buckets. But it doesn't have to: given the right kind of labeled input data, you can set of a series of non-mutually-exclusive binary classifiers.

Topic Modeling is a form of unsupervised learning (akin to clustering), so the set of possible topics are unknown prior. They're defined as part of generating the **topic models**.
With a non-deterministic algorithm like LDA, you'll get different topics each time you run the algorithm.

Topic modeling is generally not mutually-exclusive: the same document can have its probability distribution spread across many topics. In addition, there are also hierarchical topic modeling methods.

E.g. you may have a class "football", but there may be topics inside this class that relate to particular matches or teams.
The challenge with topics is that they change over time; consider the matches example above. Such topics may emerge, and disappear again.

Topic Modeling

- **Input:** A document-term matrix. Each topic will consist of a set of words where order doesn’t matter, so we’re going to start with the bag of words format.

- **gensim:** gensim is a Python toolkit built by Radim Řehůřek specifically for topic modeling. We’re going to a popular topic modeling technique called Latent Dirichlet Allocation (LDA). We’re also going to use **nltk** for some parts-of-speech tagging.

Topic Modelling Techniques

- LDA – Latent Dirichlet Allocation
- NNMF – Non-Negative Matrix Factorization
- LSA – Latent Semantic Allocation

Latent Dirichlet Allocation

- Document 1: I want to have fruits for my breakfast.
  Document 2: I like to eat almonds, eggs and fruits.
  Document 3: I will take fruits and biscuits with me while going to Zoo
  Document 4: The zookeeper feeds the lion very carefully
  Document 5: One should give good quality biscuits to their dogs

- Topic 1: 30% fruits, 15% eggs, 10% biscuits… (… food)
  Topic 2: 20% lion, 10% dogs, 5% zoo… (… animals)

- Documents 1 and 2: 100% Topic 1
  Documents 3: 100% Topic 1
  Document 4 and Document 5: 70% Topic 1, 30% Topic 2

Latent Dirichlet Allocation

I like bananas and oranges	Frogs and fish live in ponds	Kittens and puppies are fluffy	I had a spinach and apple smoothie	My kitten loves kale
100% Topic A	100% Topic B	100% Topic B	100% Topic A	60% Topic A 40% Topic B

Latent Dirichlet Allocation

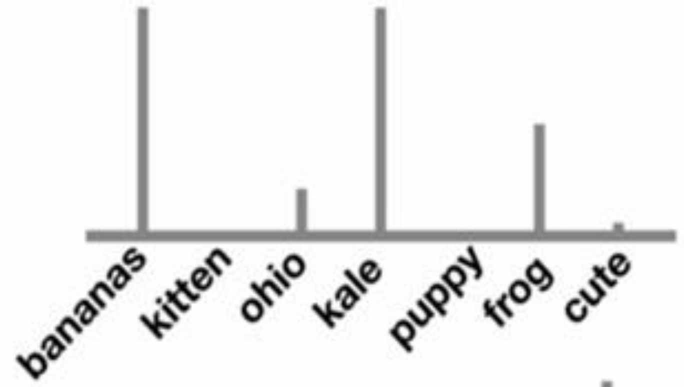
Every **document** consists of a mix of **topics**

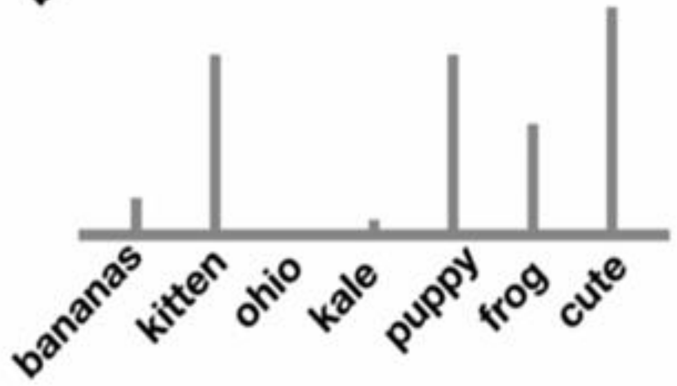
100% Topic A

100% Topic B

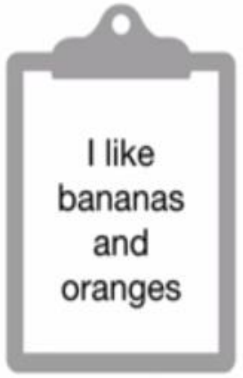
60% Topic A
40% Topic B

Every **topic** consists of a mix of **words**





Latent Dirichlet Allocation


I like bananas and oranges

Document #1

Topic A: **Food**

Topic B: **Animals**

How LDA Works

- Goal: You want LDA to learn the topic mix in each document, and the word mix in each topic
- Choose the number of topics you think there are in your corpus
  Example: K = 2
- Randomly assign each word in each document to one of 2 topics
  Example: The word 'banana' in Document #1 is randomly assigned to Topic B (animal-like topic)
- Go through every word and its topic assignment in each document. Look at (1) how often the topic occurs in the document and (2) how often the word occurs in the topic overall. Based on this info, assign the word a new topic.
  Example: It looks like (1) animals don't occur often in Document #1 and (2) 'banana' doesn't occur much in Topic B, so the word 'banana' probably should be assigned to Topic A instead
- Go through multiple iterations of this. Eventually, the topics will start making sense - interpret them.

Latent Dirichlet Allocation

I like bananas and oranges

Document #1

Topic A: **Food**

Topic B: **Animals**

How LDA Works

- Goal: You want LDA to learn the topic mix in each document, and the word mix in each topic
- Choose the number of topics you think there are in your corpus
  Example: K = 2

gensim does this part for you

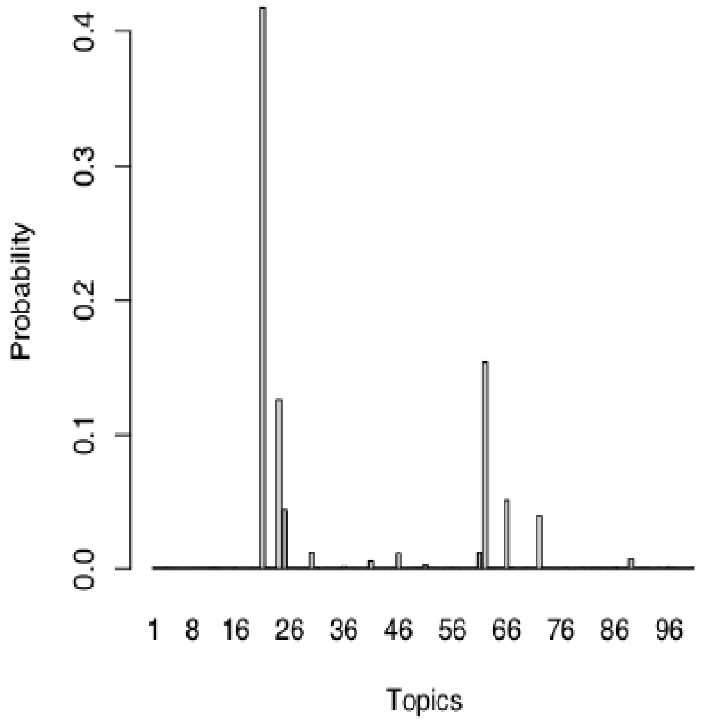
- Go through multiple iterations of this. Eventually, the topics will start making sense - interpret them.

Latent Dirichlet Allocation

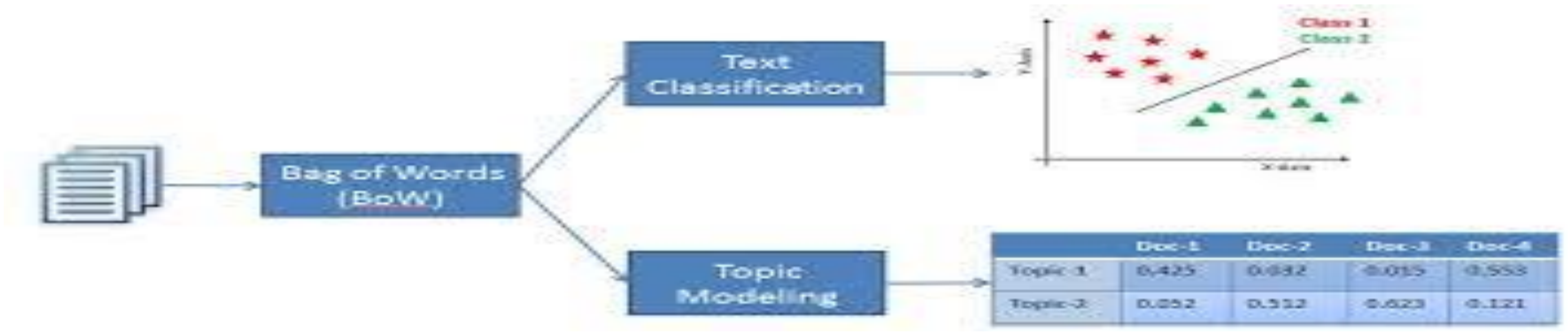
- Input: **Document-Term Matrix**, **number of topics**, **number of iterations**
- Gensim will go through the process of finding the best word distribution for each topic and best topic distribution for each document.
- Output: **The top words in each topic.** It is your job as a human to interpret this and see if the results makes sense. If not, try altering the parameters - terms in the document-term matrix, number of topics, number of iterations, etc. **Stop when the topics make sense.**
- This is a probabilistic approach to topic modeling. There are also matrix factorization techniques for topic modeling such as **Latent Semantic Indexing (LSI)** and **Non-Negative Matrix Factorization (NMF).**

Real inference with LDA

A 100-topic LDA model was fitted to **17,000 articles from the *Science* journal.**
At right are **the top 15 most frequent words** from the most frequent topics.
At left are the **inferred topic proportions** for the example article from previous slide.



"Genetics"	"Evolution"	"Disease"	"Computers"
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations



	Doc-1	Doc-2	Doc-3	Doc-4
Topic-1	0.425	0.032	0.015	0.553
Topic-2	0.252	0.512	0.623	0.121

Figure 1. Topic Modeling

General steps of text categorization are shown in Fig. 1.

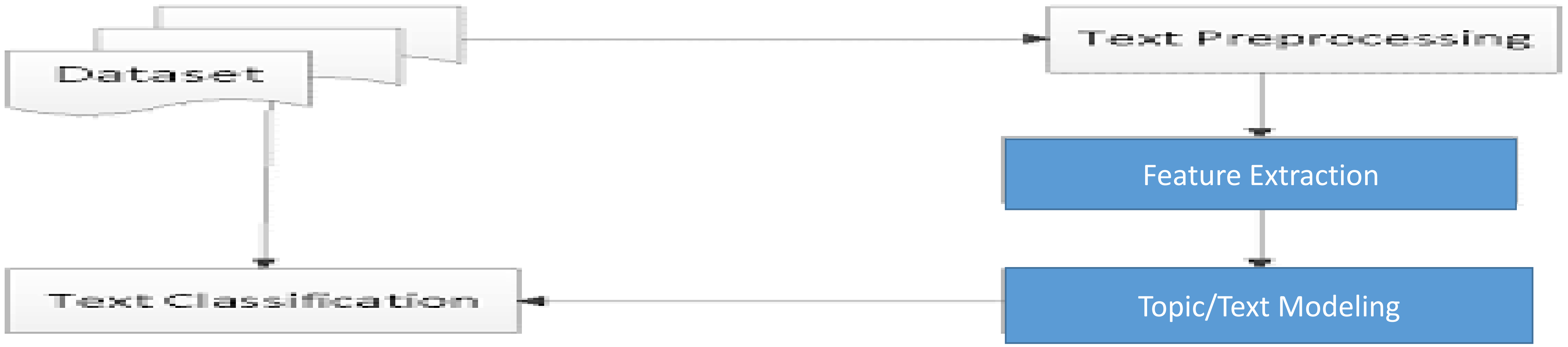


Figure 2. Text Modeling vs Text Classification

Word Embedding using Word2Vec

Word Embedding is a language modeling technique used for mapping words to vectors of real numbers. It represents words or phrases in vector space with several dimensions. Word embeddings can be generated using various methods like neural networks, co-occurrence matrix, probabilistic models, etc.

Word2Vec consists of models for generating word embedding. These models are shallow two layer neural networks having one input layer, one hidden layer and one output layer. Word2Vec utilizes two architectures :

1.CBOW (Continuous Bag of Words) : CBOW model predicts the current word given context words within specific window. The input layer contains the context words and the output layer contains the current word. The hidden layer contains the number of dimensions in which we want to represent current word present at the output layer.

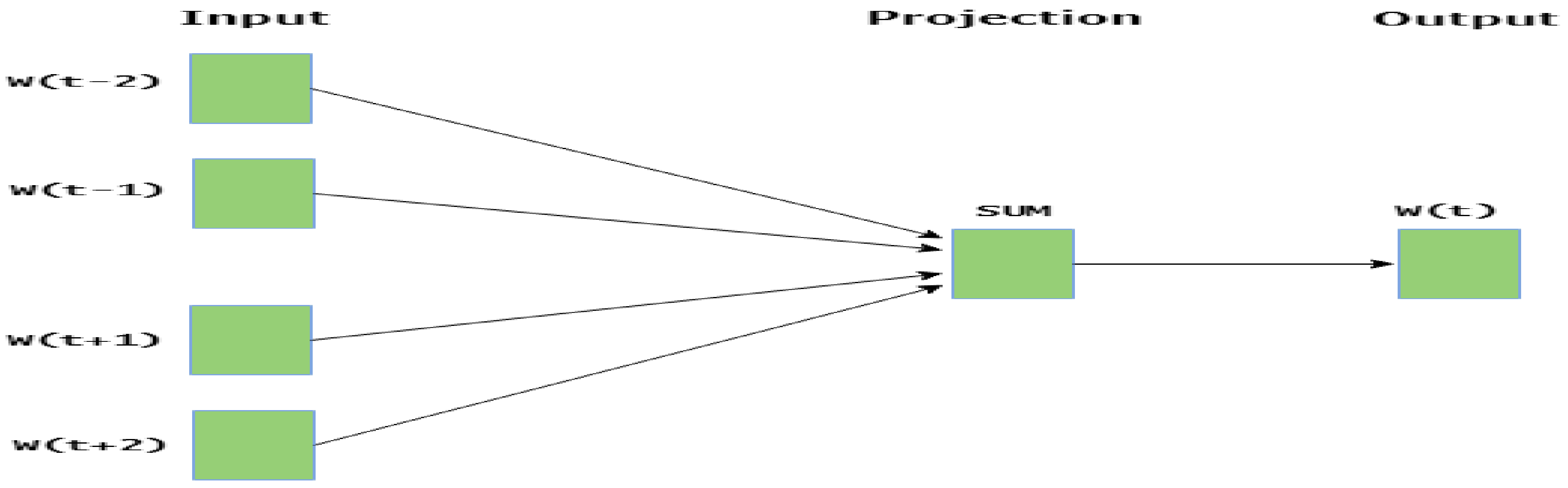


Figure 3 . CBOW (Continuous Bag of Words)

Skip Gram :

1 .Skip gram predicts the surrounding context words within specific window given current word.

2 .The input layer contains the current word and the output layer contains the context words.

3 .The hidden layer contains the number of dimensions in which we want to represent current word present at the input layer.

4.The basic idea of word embedding is words that occur in similar context tend to be closer to each other in vector space.

5.For generating word vectors in Python, modules needed are **nltk and gensim.**

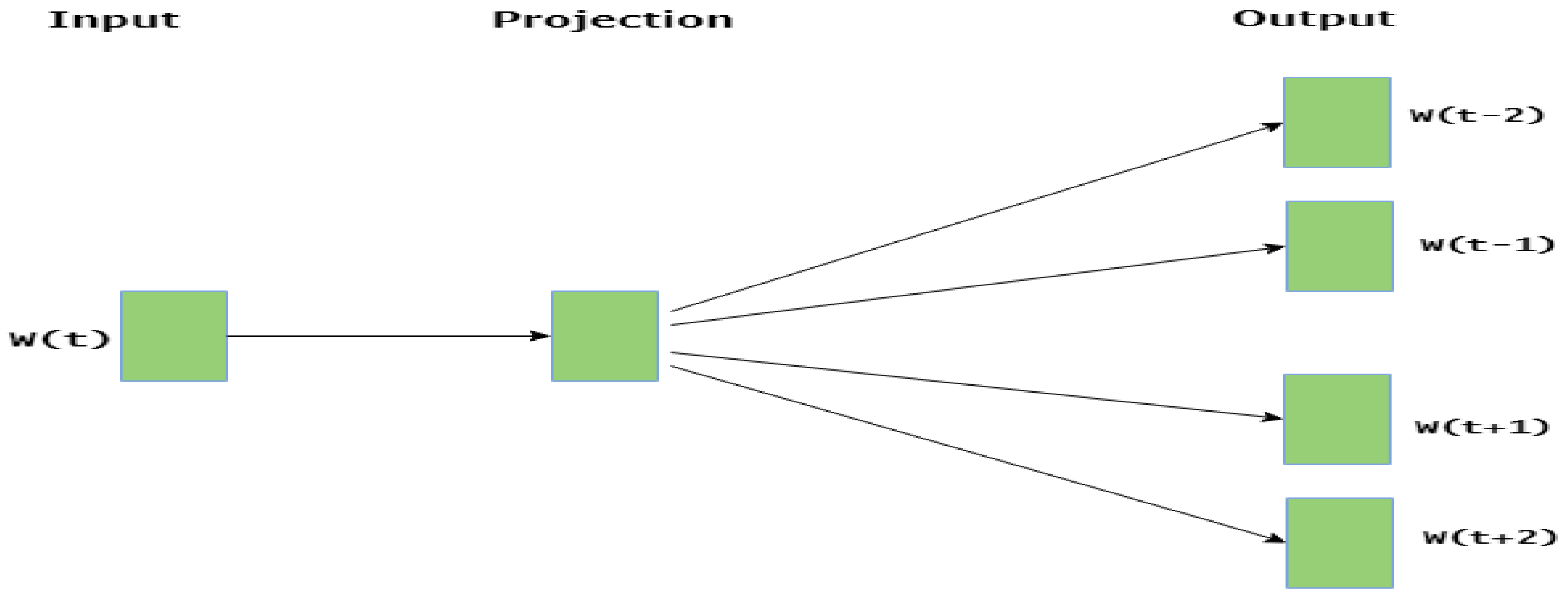


Figure : 4 Skip Gram

Topic modeling applications

- Topic-based text classification
- Classical text classification algorithms (e.g. perceptron, naive bayes, k-nearest neighbor, SVM, AdaBoost, etc.) are often assuming bag-of-words representation of input data.
- Topic modeling can be seen as a pre-processing step before applying supervised learning methods.
- Using topic-based representation it is possible to gain ~0,039 in precision and ~0,046 in F1 score [Cai, Hofmann, 2003]
- Collaborative filtering [Hofmann, 2004]
- Finding patterns in genetic data, images, and social networks

CBOW vs Skip Gram word2vec Model

Two varieties of word2vec, the Continuous Bag of Words (CBOW) model, and the Continuous Skip-Gram model.

The CBOW model learns word embedding's by predicting the current word based on its context.
Skip-gram learns word embedding's by predicting the context (surrounding words) of the current word.

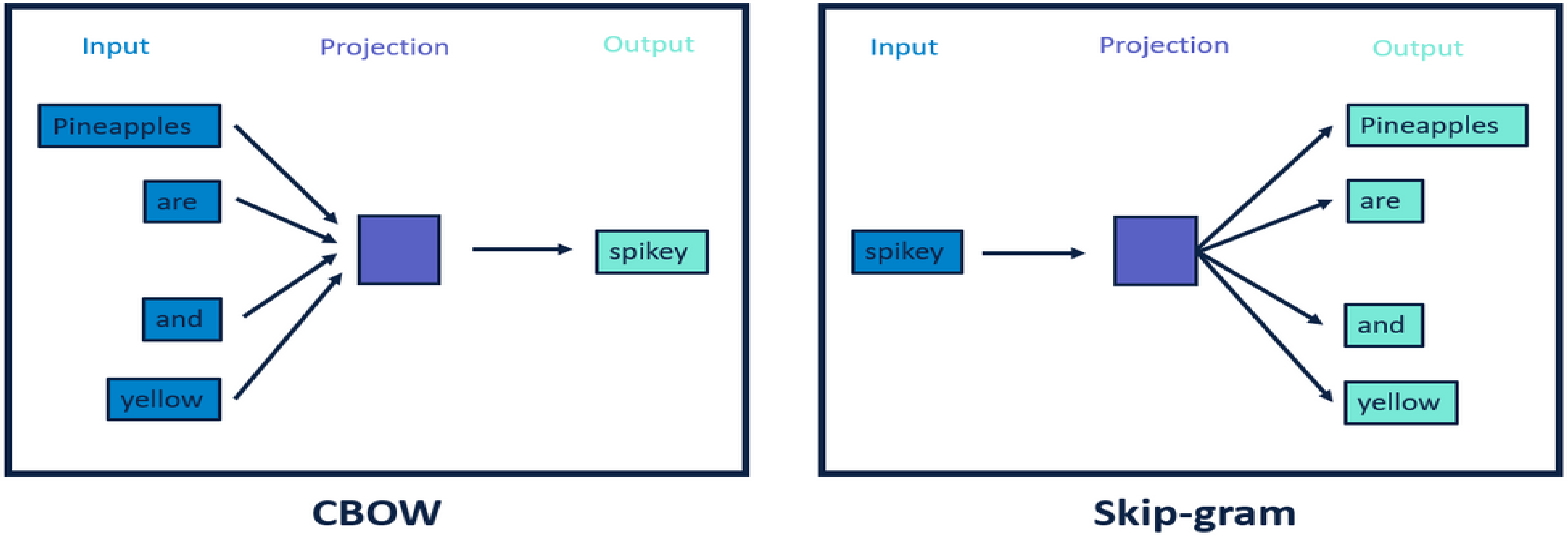


Figure 5 : CBOW vs SKIP GRAM MODEL