



WIKIPEDIA
The Free Encyclopedia

[Main page](#)
[Contents](#)
[Current events](#)
[Random article](#)
[About Wikipedia](#)
[Contact us](#)
[Donate](#)

[Contribute](#)

[Help](#)
[Community portal](#)
[Recent changes](#)
[Upload file](#)

[Tools](#)

[What links here](#)
[Related changes](#)
[Special pages](#)
[Permanent link](#)
[Page information](#)
[Cite this page](#)
[Wikidata item](#)

[Print/export](#)

[Download as PDF](#)
[Printable version](#)

[Languages](#)

[العربية](#)
[Español](#)
[Français](#)
[한국어](#)
[Bahasa Indonesia](#)
[Italiano](#)

Not logged in [Talk](#) [Contributions](#) [Create account](#) [Log in](#)

Article [Talk](#)

Read

[Edit](#)

[View history](#)



Bag-of-words model

From Wikipedia, the free encyclopedia

For Bag-of-words model in computer vision, see [Bag-of-words model in computer vision](#).

The **bag-of-words model** is a simplifying representation used in [natural language processing](#) and [information retrieval](#) (IR). 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](#). The bag-of-words model has also been [used for computer vision](#).^[1]

The bag-of-words model is commonly used in methods of [document classification](#) where the (frequency of) occurrence of each word is used as a [feature](#) for training a [classifier](#).^[2]

An early reference to "bag of words" in a linguistic context can be found in [Zellig Harris](#)'s 1954 article on *Distributional Structure*.^[3]

Contents [\[hide\]](#)

- [Example implementation](#)
- [Application](#)
- [n-gram model](#)
- [Python implementation](#)
- [Hashing trick](#)
- [Example usage: spam filtering](#)
- [See also](#)
- [Notes](#)
- [References](#)

Example implementation [\[edit \]](#)

The following models a text document using bag-of-words. Here are two simple text documents:

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

(2) Mary also likes to watch football games.

Based on these two text documents, a list is constructed as follows for each document:

```
"John", "likes", "to", "watch", "movies", "Mary", "likes", "movies", "too"
"Mary", "also", "likes", "to", "watch", "football", "games"
```

Representing each bag-of-words as a [JSON object](#), and attributing to the respective [JavaScript](#) variable:

```
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 };
```

Each key is the word, and each value is the number of occurrences of that word in the given text document.

The order of elements is free, so, for example `{"too":1, "Mary":1, "movies":2, "John":1, "watch":1, "likes":2, "to":1}` is also equivalent to *BoW1*. It is also what we expect from a strict *JSON object* representation.

Note: if another document is like a union of these two,

```
(3) John likes to watch movies. Mary likes movies too. Mary also likes to watch football games.
```

its JavaScript representation will be:

```
BoW3 = { "John":1, "likes":3, "to":2, "watch":2, "movies":2, "Mary":2, "too":1, "also":1, "football":1, "games":1 };
```

So, as we see in the [bag algebra](#), the "union" of two documents in the bags-of-words representation is, formally, the [disjoint union](#), summing the multiplicities of each element.

$$BoW3 = BoW1 \uplus BoW2.$$

Application [\[edit \]](#)

In practice, the Bag-of-words model is mainly used as a tool of feature generation. After transforming the text into a "bag of words", we can calculate various measures to characterize the text. The most common type of characteristics, or features calculated from the Bag-of-words model is term frequency, namely, the number of times a term appears in the text. For the example above, we can construct the following two lists to record the term frequencies of all the distinct words (BoW1 and BoW2 ordered as in BoW3):

```
(1) [1, 2, 1, 1, 2, 1, 1, 0, 0, 0]
(2) [0, 1, 1, 1, 0, 1, 0, 1, 1, 1]
```

Each entry of the lists refers to the count of the corresponding entry in the list (this is also the histogram representation). For example, in the first list (which represents document 1), the first two entries are "1,2":

- The first entry corresponds to the word "John" which is the first word in the list, and its value is "1" because "John" appears in the first document once.
- The second entry corresponds to the word "likes", which is the second word in the list, and its value is "2" because "likes" appears in the first document twice.

This list (or vector) representation does not preserve the order of the words in the original sentences. This is just the main feature of the Bag-of-words model. This kind of representation has several successful applications, such as [email filtering](#).^[1]

However, term frequencies are not necessarily the best representation for the text. Common words like "the", "a", "to" are almost always the terms with highest frequency in the text. Thus, having a high raw count does not necessarily mean that the corresponding word is more important. To address this problem, one of the most popular ways to "normalize" the term frequencies is to weight a term by the inverse of document frequency, or [tf-idf](#). Additionally, for the specific purpose of classification, [supervised](#) alternatives have been developed to account for the class label of a document.^[4] Lastly, binary (presence/absence or 1/0) weighting is used in place of frequencies for some problems (e.g., this option is implemented in the [WEKA](#) machine learning software system).

[n-gram model](#) [\[edit \]](#)

The Bag-of-words model is an orderless document representation — only the counts of words matter. For instance, in the above example "John likes to watch movies. Mary likes movies too", the bag-of-words representation will not reveal that the verb "likes" always follows a person's name in this text. As an alternative, the [n-gram](#) model can store this spatial information. Applying to the same example above, a **bigram** model will parse the text into the following units and store the term frequency of each unit as before.

```
[
    "John likes",
    "likes to",
    "to watch",
    "watch movies",
    "Mary likes",
    "likes movies",
    "movies too",
]
```

Conceptually, we can view bag-of-word model as a special case of the n-gram model, with $n=1$. For $n>1$ the model is named [w-shingling](#) (where w is equivalent to n denoting the number of grouped words). See [language model](#) for a more detailed discussion.

[Python implementation](#) [\[edit \]](#)

```
from keras.preprocessing.text import Tokenizer

sentence = ["John likes to watch movies. Mary likes movies too."]

def print_bow(sentence: str) -> None:
    tokenizer = Tokenizer()
    tokenizer.fit_on_texts(sentence)
    sequences = tokenizer.texts_to_sequences(sentence)
    word_index = tokenizer.word_index
    bow = {}
    for key in word_index:
        bow[key] = sequences[0].count(word_index[key])
    print(bow)
    print(f"Bag of word sentence 1 :\n{bow}")
    print(f'We found {len(word_index)} unique tokens.')

print_bow(sentence)
```

Hashing trick [\[edit\]](#)

A common alternative to using dictionaries is the [hashing trick](#), where words are mapped directly to indices with a hashing function.^[5] Thus, no memory is required to store a dictionary. Hash collisions are typically dealt via freed-up memory to increase the number of hash buckets. In practice, hashing simplifies the implementation of bag-of-words models and improves scalability.

Example usage: spam filtering [\[edit\]](#)

In [Bayesian spam filtering](#), an e-mail message is modeled as an unordered collection of words selected from one of two probability distributions: one representing [spam](#) and one representing legitimate e-mail ("ham"). Imagine there are two literal bags full of words. One bag is filled with words found in spam messages, and the other with words found in legitimate e-mail. While any given word is likely to be somewhere in both bags, the "spam" bag will contain spam-related words such as "stock", "Viagra", and "buy" significantly more frequently, while the "ham" bag will contain more words related to the user's friends or workplace.





To classify an e-mail message, the Bayesian spam filter assumes that the message is a pile of words that has been poured out randomly from one of the two bags, and uses [Bayesian probability](#) to determine which bag it is more likely to be in.

See also [\[edit\]](#)

- [Additive smoothing](#)
- [Bag-of-words model in computer vision](#)
- [Document classification](#)
- [Document-term matrix](#)

- [Feature extraction](#)
- [Hashing trick](#)
- [Machine learning](#)
- [MinHash](#)
- [n-gram](#)
- [Natural language processing](#)
- [Vector space model](#)
- [w-shingling](#)
- [tf-idf](#)

Notes [[edit](#)]

- [^] ^{[a](#)} ^{[b](#)} Sivic, Josef (April 2009). "Efficient visual search of videos cast as text retrieval"  (PDF). *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, VOL. 31, NO. 4. IEEE. pp. 591–605.
- [^] McTear et al 2016, p. 167.
- [^] Harris, Zellig (1954). "Distributional Structure". *Word*. **10** (2/3): 146–62. doi:10.1080/00437956.1954.11659520 . "And this stock of combinations of elements becomes a factor in the way later choices are made ... for language is not merely a bag of words but a tool with particular properties which have been fashioned in the course of its use"
- [^] Youngjoong Ko (2012). "A study of term weighting schemes using class information for text classification". *SIGIR'12. ACM*.
- [^] Weinberger, K. Q.; Dasgupta A.; Langford J.; Smola A.; Attenberg, J. (2009). "Feature hashing for large scale multitask learning". *Proceedings of the 26th Annual International Conference on Machine Learning*: 1113–1120. arXiv:0902.2206 . Bibcode:2009arXiv0902.2206W .

References [[edit](#)]

- McTear, Michael (et al) (2016). *The Conversational Interface*. Springer International Publishing.

V•T•E

Natural language processing

[show]

Categories: [Natural language processing](#) | [Machine learning](#)

This page was last edited on 11 May 2020, at 17:26 (UTC).

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. By using this site, you agree to the [Terms of Use](#) and [Privacy Policy](#). Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#) [Contact Wikipedia](#) [Developers](#) [Statistics](#) [Cookie statement](#) [Mobile view](#)