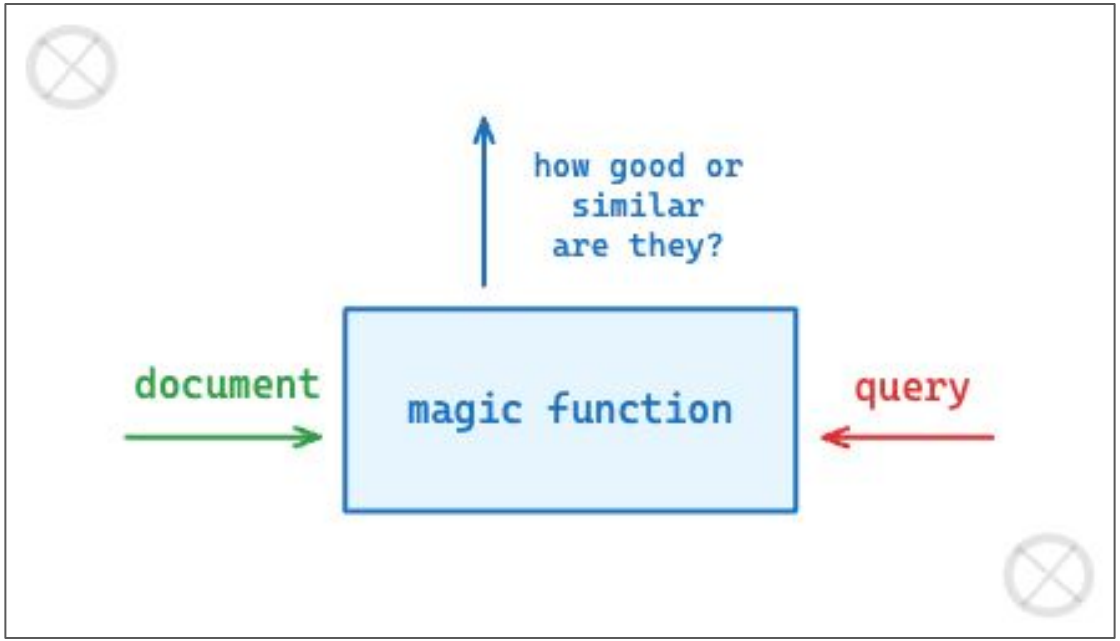
FOUNDERS AND CODERS

Machine Learning Institute

A gentle intro to information retrieval

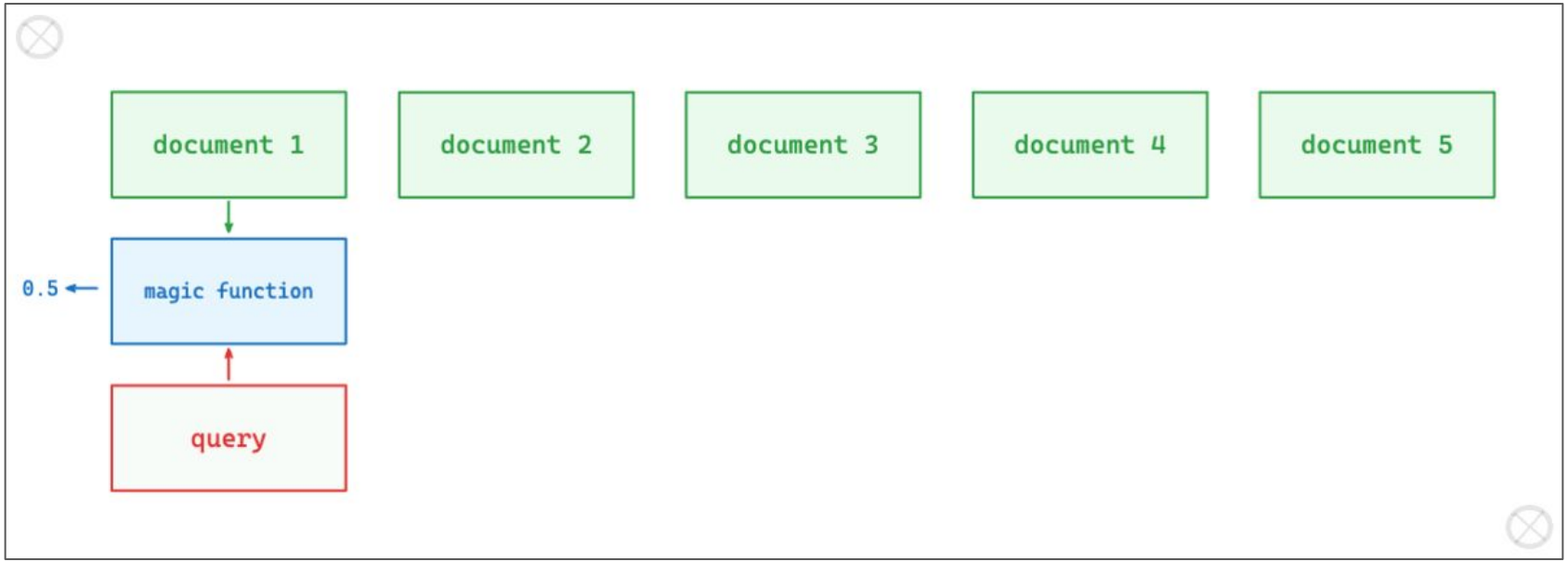
Learn To Search



FOUNDERS AND CODERS

Learn To Search & Rank

Machine Learning Institute



TF-IDF

Term Frequency

A: foo bar zoo zoo	foo	1/4	0.25
	bar	1/4	0.25
	zoo	2/4	0.50

$$\mathrm{TF}(t,d) = \frac{f_{t,d}}{\sum_{w \in \mathrm{V}} f_{w,d}}$$

$f_{t,d} = \mathrm{count}_d(t)$ simple count function :)

Inverse Document Frequency

A: foo bar zoo zoo; B: foo bar; C: zoo bar	foo	ln(3/2)	0.405
	bar	ln(3/3)	0.000
	zoo	ln(3/2)	0.405

$$\mathrm{IDF}(t,\mathcal{D}) = \log \frac{|\mathcal{D}|}{\mathrm{df}(t)}$$

$\mathrm{df}(t) = \sum_{d \in \mathcal{D}} \mathbf{1}_{\{t \in d\}}$ loop over all documents and add "1" if it contains the current term "t"

Final Score = TF * IDF

A: foo bar zoo zoo	foo	1/4 * ln(3/2)	0.1
	bar	1/4 * ln(3/3)	0.0
	zoo	2/4 * ln(3/2)	0.2
B: foo bar	foo	1/2 * ln(3/2)	0.2
	bar	1/2 * ln(3/3)	0.0
C: zoo bar	zoo	1/2 * ln(3/2)	0.2
	bar	1/2 * ln(3/3)	0.0

Inverted Index

Final Score = TF * IDF

A foo bar zoo zoo	foo	1/4 * ln(3/2)	0.1
	bar	1/4 * ln(3/3)	0.0
	zoo	2/4 * ln(3/2)	0.2
B foo bar	foo	1/2 * ln(3/2)	0.2
	bar	1/2 * ln(3/3)	0.0
C zoo bar	zoo	1/2 * ln(3/2)	0.2
	bar	1/2 * ln(3/3)	0.0

Inverted Index

foo	(B, 0.2), (A, 0.1)
bar	(A, 0.0), (B, 0.0), (C, 0.0)
zoo	(A, 0.2), (C, 0.2)

Is it good enough?

On the web, this strategy often returns very short documents that are the query plus a few words. For example, we have seen a major search engine return a page containing only "Bill Clinton Sucks" and picture from a "Bill Clinton" query. Some argue that on the web, users should specify more accurately what they want and add more words to their query. We disagree vehemently with this position. If a user issues a query like "Bill Clinton" they should get reasonable results since there is a enormous amount of high quality information available on this topic. Given examples like these, we believe that the standard information retrieval work needs to be extended to deal effectively with the web.

FOUNDERS AND CODERS

1998

Machine Learning Institute

The anatomy of a large-scale hypertextual Web search engine [1]

Sergey Brin [2], Lawrence Page [*,2]

Computer Science Department, Stanford University, Stanford, CA 94305, USA

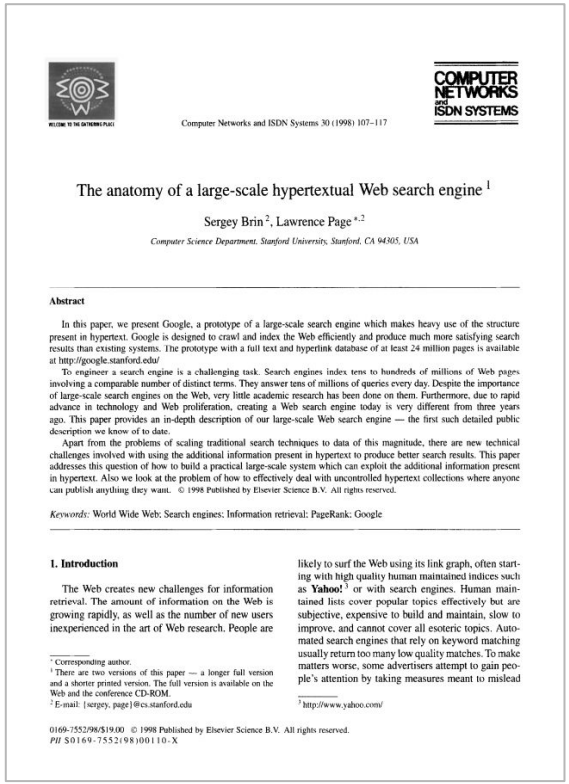
Abstract

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at http://google.stanford.edu/

To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of Web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the Web, very little academic research has been done on them. Furthermore, due to rapid advance in technology and Web proliferation, creating a Web search engine today is very different from three years ago. This paper provides an in-depth description of our large-scale Web search engine — the first such detailed public description we know of to date.

Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. This paper addresses this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

Keywords: World Wide Web; Search engines; Information retrieval; PageRank; Google



Computer Networks and ISDN Systems 30 (1998) 107–117

COMPUTER NETWORKS and ISDN SYSTEMS

The anatomy of a large-scale hypertextual Web search engine [1]

Sergey Brin [2], Lawrence Page [*,2]

Computer Science Department, Stanford University, Stanford, CA 94305, USA

Abstract

In this paper, we present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Google is designed to crawl and index the Web efficiently and produce much more satisfying search results than existing systems. The prototype with a full text and hyperlink database of at least 24 million pages is available at http://google.stanford.edu/

To engineer a search engine is a challenging task. Search engines index tens to hundreds of millions of Web pages involving a comparable number of distinct terms. They answer tens of millions of queries every day. Despite the importance of large-scale search engines on the Web, very little academic research has been done on them. Furthermore, due to rapid advance in technology and Web proliferation, creating a Web search engine today is very different from three years ago. This paper provides an in-depth description of our large-scale Web search engine — the first such detailed public description we know of to date.

Apart from the problems of scaling traditional search techniques to data of this magnitude, there are new technical challenges involved with using the additional information present in hypertext to produce better search results. This paper addresses this question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Also we look at the problem of how to effectively deal with uncontrolled hypertext collections where anyone can publish anything they want.

Keywords: World Wide Web; Search engines; Information retrieval; PageRank; Google

1. Introduction

The Web creates new challenges for information retrieval. The amount of information on the Web is growing rapidly, as well as the number of new users inexperienced in the art of Web research. People are likely to surf the Web using its link graph, often starting with high quality human maintained indices such as **Yahoo!** [3] or with search engines. Human maintained lists cover popular topics effectively but are subjective, expensive to build and maintain, slow to improve, and cannot cover all esoteric topics. Automated search engines that rely on keyword matching usually return too many low quality matches. To make matters worse, some advertisers attempt to gain people's attention by taking measures meant to mislead

* Corresponding author.

[1] There are two versions of this paper — a longer full version and a shorter printed version. The full version is available on the Web and the conference CD-ROM.

[2] E-mail: {sergey, page}@cs.stanford.edu

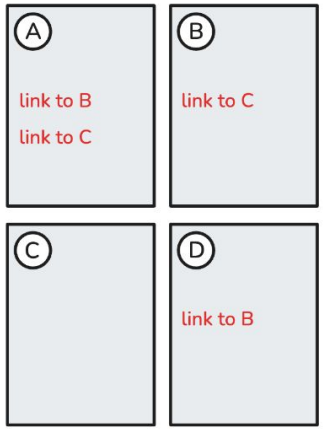
[3] http://www.yahoo.com/

0169-7552/98/$19.00
PII S0169-7552(98)00110-X

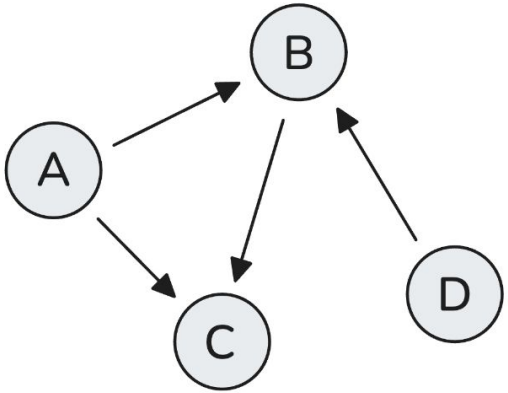
FOUNDERS AND CODERS

Basic Graph Theory

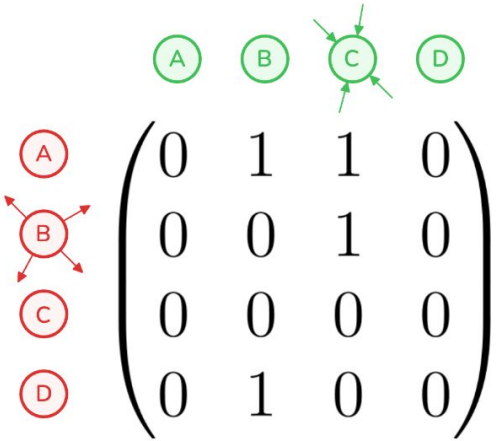
Machine Learning Institute



Documents



Graph



Adjacency Matrix

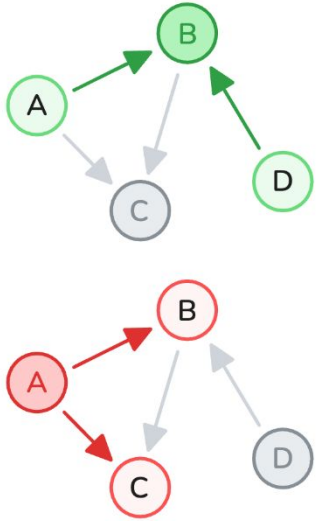
Point of views

inward just arrived here, what is the probability I came from a specific node

outward

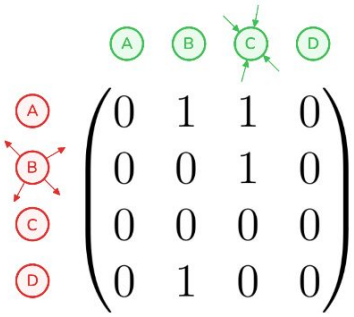
starting from this node what is the probability of ending in the next node

$$\begin{bmatrix} p_{a\to a} & p_{a\to b} & p_{a\to c} & p_{a\to d} \\ p_{b\to a} & p_{b\to b} & p_{b\to c} & p_{b\to d} \\ p_{c\to a} & p_{c\to b} & p_{c\to c} & p_{c\to d} \\ p_{d\to a} & p_{d\to b} & p_{d\to c} & p_{d\to d} \end{bmatrix}$$



From Graph To Markov Chain

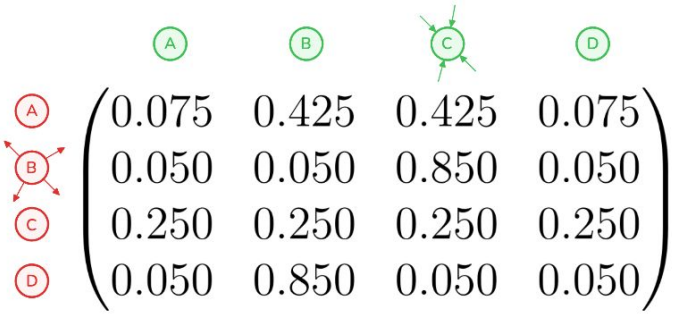
Adjacency Matrix



Teleportation alpha=0.15

$$p_{ij} = \begin{cases} \dfrac{1-\alpha}{k_i}, & \text{if } a_{ij} = 1 \text{ and } k_i > 0 \\ \dfrac{\alpha}{n-k_i}, & \text{if } a_{ij} = 0 \text{ and } k_i > 0 \\ \dfrac{1}{n}, & \text{if } k_i = 0 \end{cases}$$

Transition Matrix

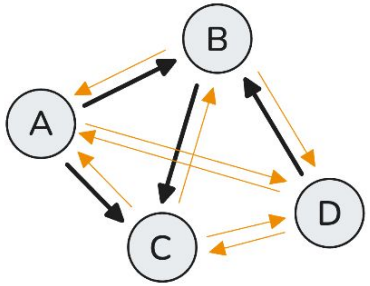


The Minions Walk

Transition Matrix

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0.075 & 0.425 & 0.425 & 0.075 \\ 0.050 & 0.050 & 0.850 & 0.050 \\ 0.250 & 0.250 & 0.250 & 0.250 \\ 0.050 & 0.850 & 0.050 & 0.050 \end{pmatrix} \end{array}$$

Fully Connected Graph



The Stationary Equation

$$\boldsymbol{\pi} = \boldsymbol{\pi} T$$

$$\boldsymbol{\pi} = \left[\pi_A,\ \pi_B,\ \pi_C,\ \pi_D \right]$$

$$\boldsymbol{\pi}^{(0)} = \left[0.25_A,\ 0.25_B,\ 0.25_C,\ 0.25_D \right]$$

FOUNDERS AND CODERS

Random walks & Markov chains

Machine Learning Institute

- Andrey Markov introduced them in 1906 (~120 years ago)
- Random walks are a core model in stochastic processes
- Model everything from: weather, games & stock markets
- PageRank didn’t invent anything new mathematically
- Teleportation makes PageRank’s random walk ergodic

Can we do better than TF-IDF?

FOUNDERS AND CODERS

Distribution hypothesis

Machine Learning Institute

What is the meaning of bardiwac?

- He poured her a glass of bardiwac.
- Beef tastes better when you drink bardiwac with it.
- Nigel felt dizzy after too much bardiwac.
- The bardiwac grapes grow well in warm sunshine.
- I ate bread and cheese with this good bardiwac.
- They served dark red bardiwac and a light, sweet white wine.

word2vec - recipe

1

Treat the entire corpus as one long string. Slide a window of five tokens across it; for each window, mask the middle token and use the remaining four to predict the "?"

For dinner we served dark red bardiwac with steak and mushrooms.

For dinner we served dark red bardiwac with steak and mushrooms.

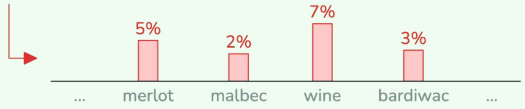
dark red ? with steak

? = f(dark, red, with, steak)

2

Because any given context can sensibly be completed by several different words, we don't force the model to output one "correct" token. Instead, we teach it to return a probability distribution over the whole vocabulary

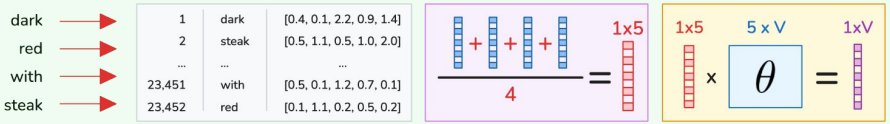
$$p_\theta(? \mid \text{"dark"}, \text{"red"}, \text{"with"}, \text{"steak"})$$

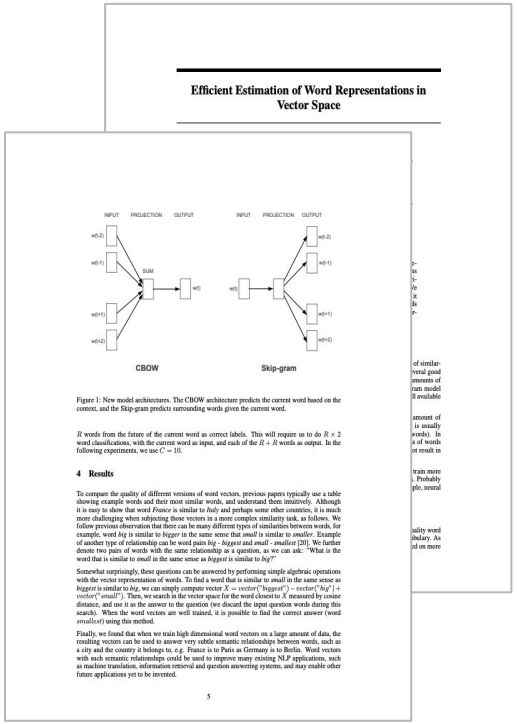


3

CBOW

Continuous Bag Of Words





Efficient Estimation of Word Representations in Vector Space

CBOW Skip-gram

Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

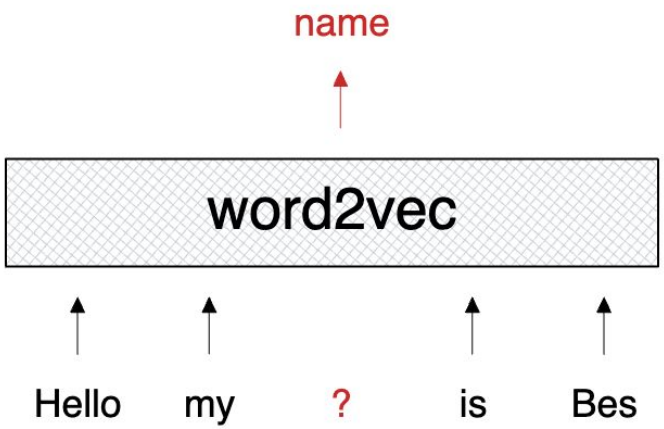
4 Results

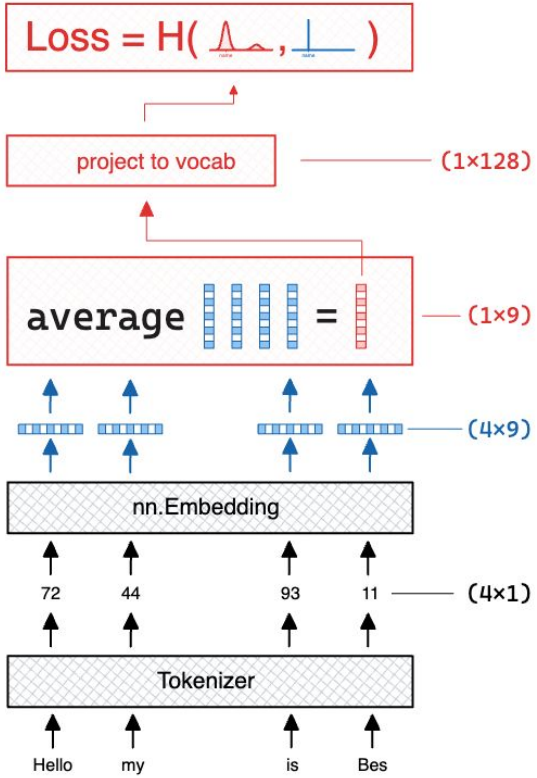
Mikolov et al. 2013

FOUNDERS AND CODERS

word2vec - cbow

Machine Learning Institute



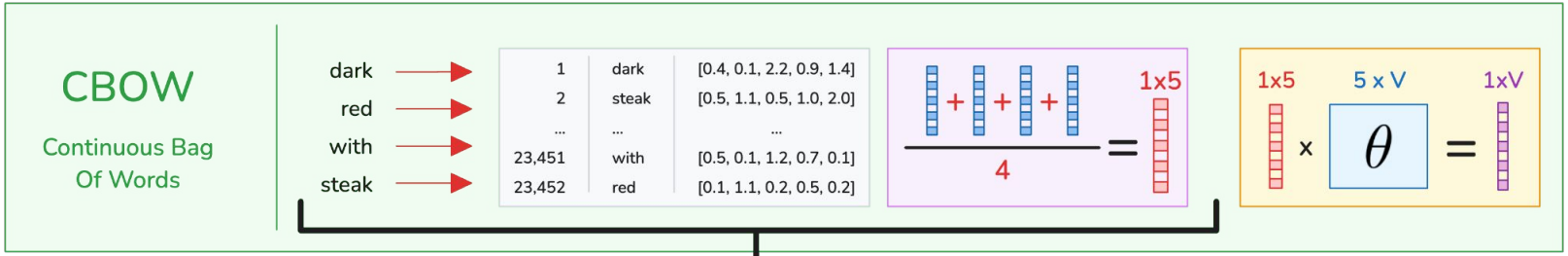



```python
#
#
#
import torch
import torch.nn.functional as F

#
#
#
vocab = {
  "Hello": 72,
  "my": 44,
  "name": 21,
  "is": 93,
  "Bes": 11
}

#
#
#
sentence = ["Hello", "my", "is", "Bes"]

#
#
#
class CBOW(torch.nn.Module):  ⇥ tab
  def __init__(self):
      super(CBOW, self).__init__()
      self.emb = torch.nn.Embedding(128, 9)
      self.linear = torch.nn.Linear(9, 128)

  def forward(self, inputs):
      embs = self.emb(inputs)
      embs = embs.mean(dim=1)
      out = self.linear(embs)
      probs = F.log_softmax(out, dim=1)
      return probs

```

FOUNDERS AND CODERS

Notice something…

Machine Learning Institute



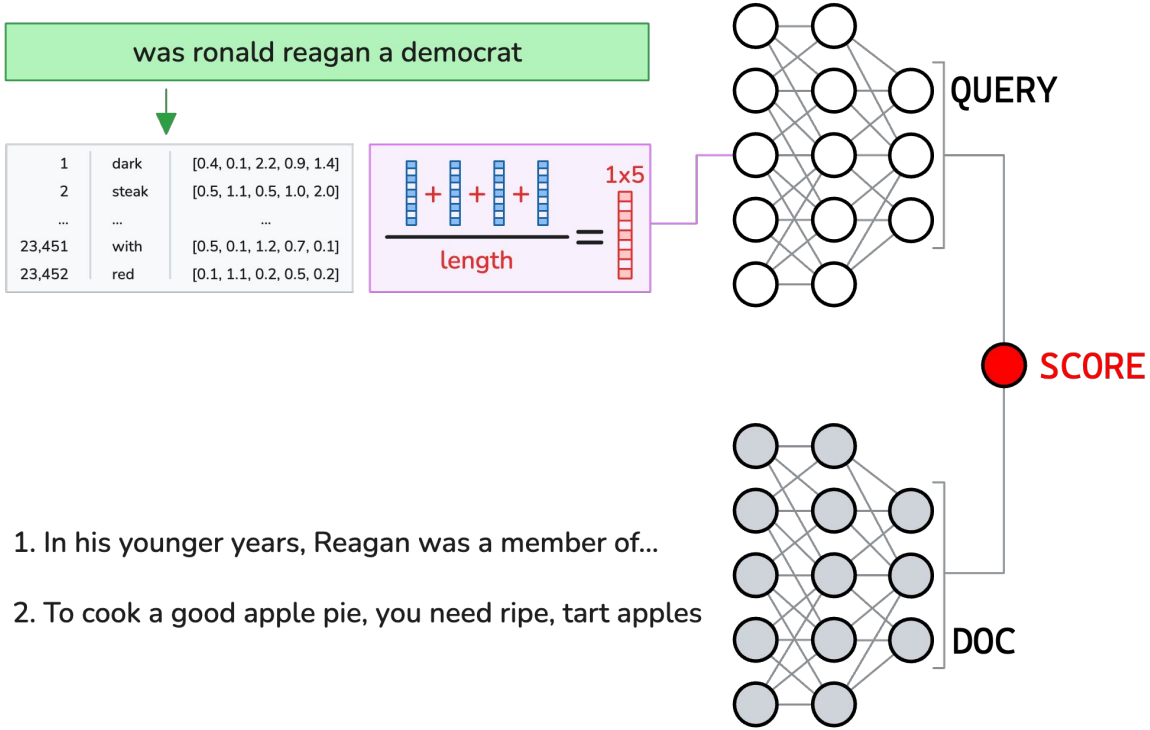
This is an encoder model

Given any sentence, the model will produce a vector representation.

FOUNDERS AND CODERS

Search Systems

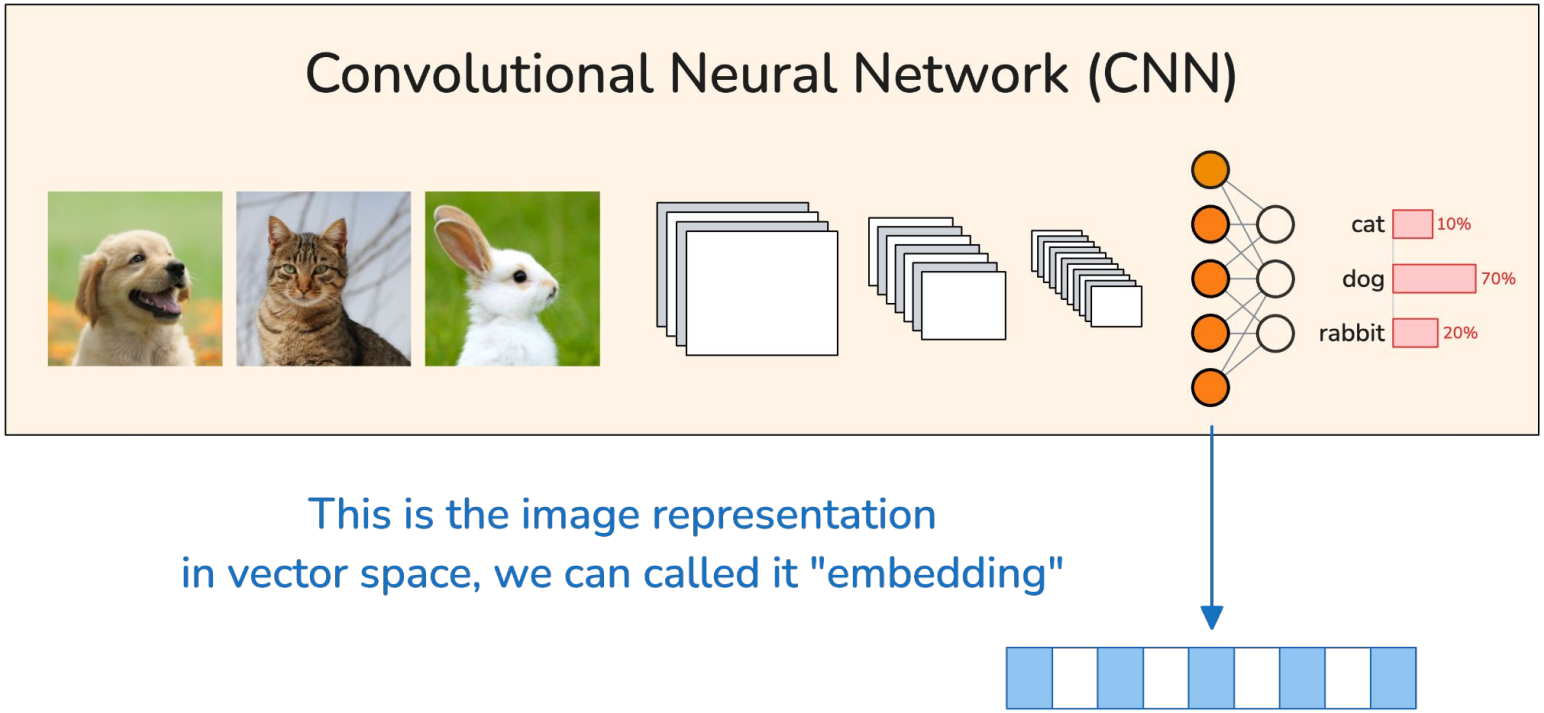
Machine Learning Institute



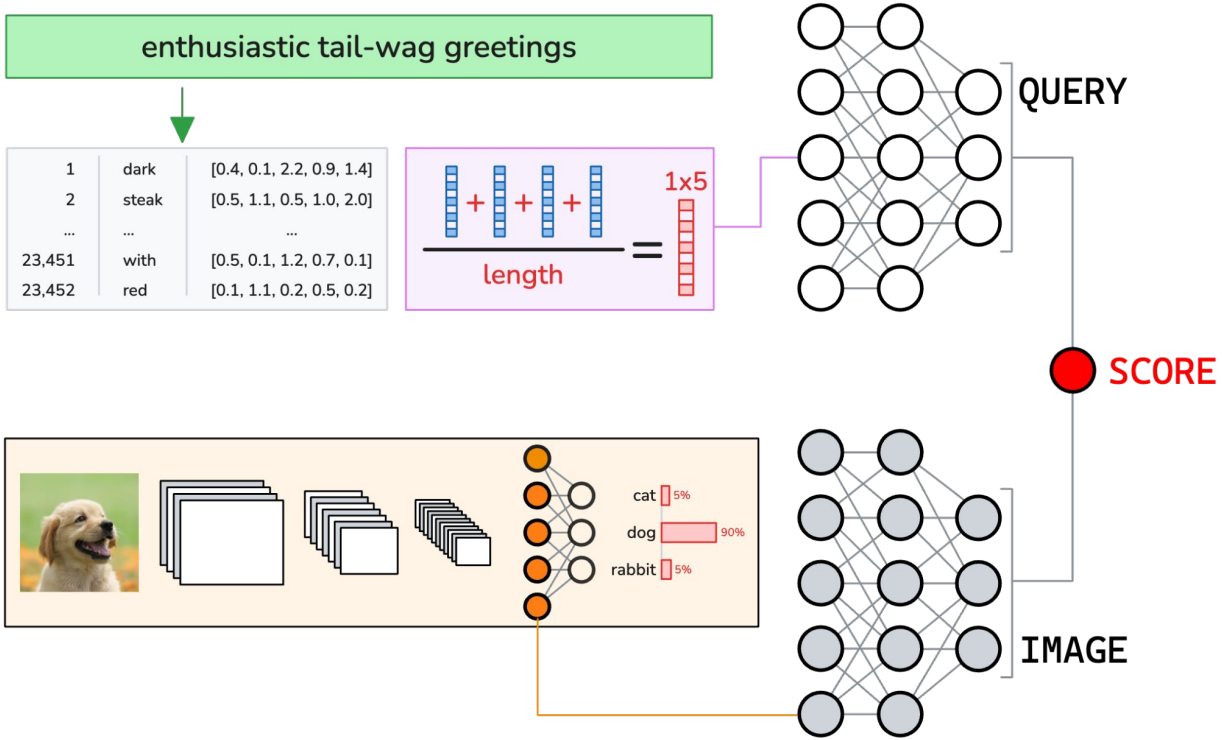
1. In his younger years, Reagan was a member of...

2. To cook a good apple pie, you need ripe, tart apples

Image representation



Multimodal Search Systems



FOUNDERS AND CODERS

Recap

Machine Learning Institute

We covered tons!

- classic TF-IDF
- PageRank algorithm and the birth of Google
- Graphs and Markov Chains close connection
- multimodal search systems

FOUNDERS AND CODERS

Machine Learning Institute

Thank you!