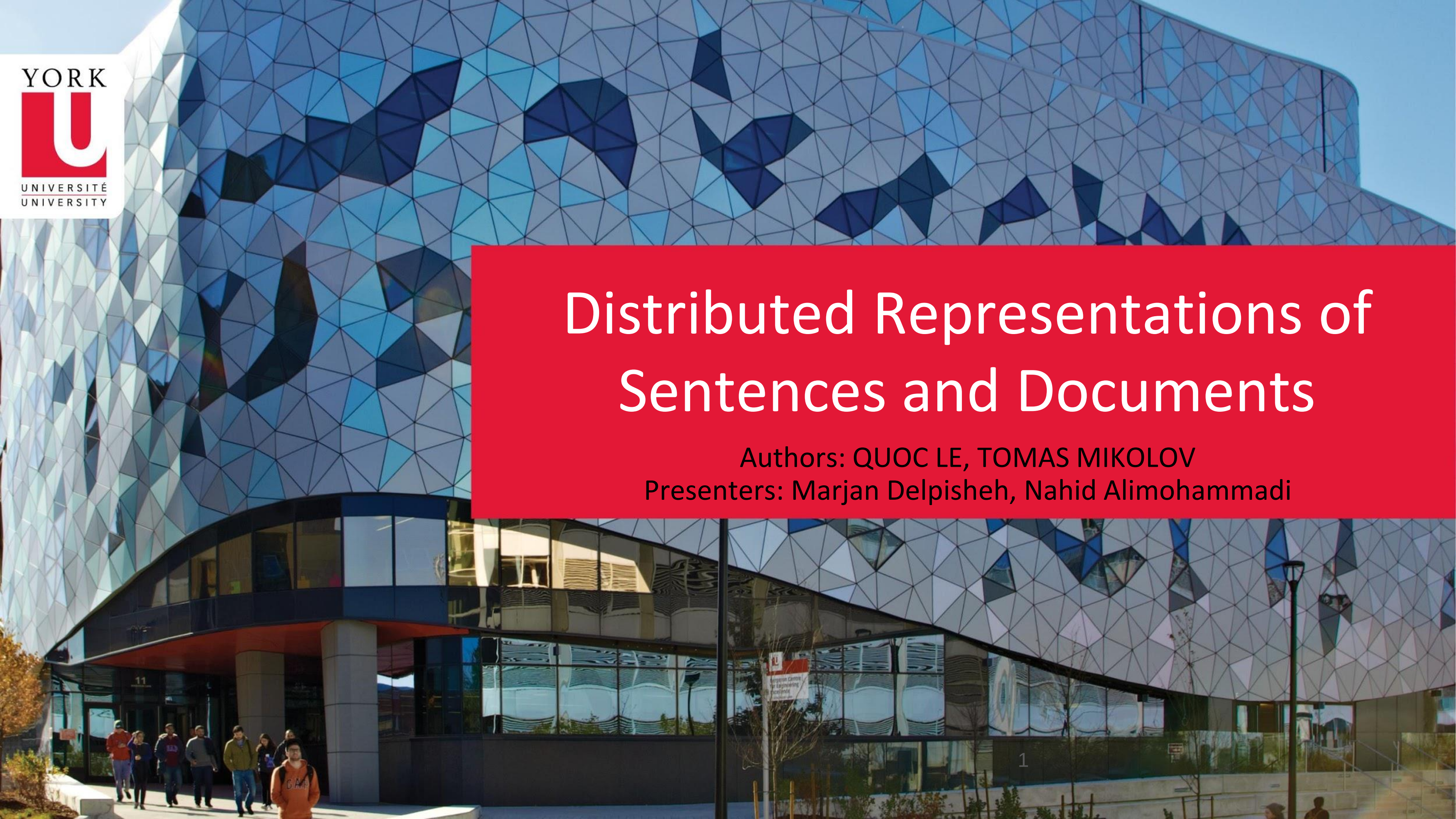


Distributed Representations of Sentences and Documents

Authors: QUOC LE, TOMAS MIKOLOV
Presenters: Marjan Delpisheh, Nahid Alimohammadi

Outline

- Objective of the paper
- Related works
- Algorithms
- Limitations and advantages
- Experiments
- Recap

Objective

- Text classification and clustering play an important role in many applications, e.g., document retrieval, web search, spam filtering
- Machine Learning algorithm require the text input to be represented as a fixed length vector

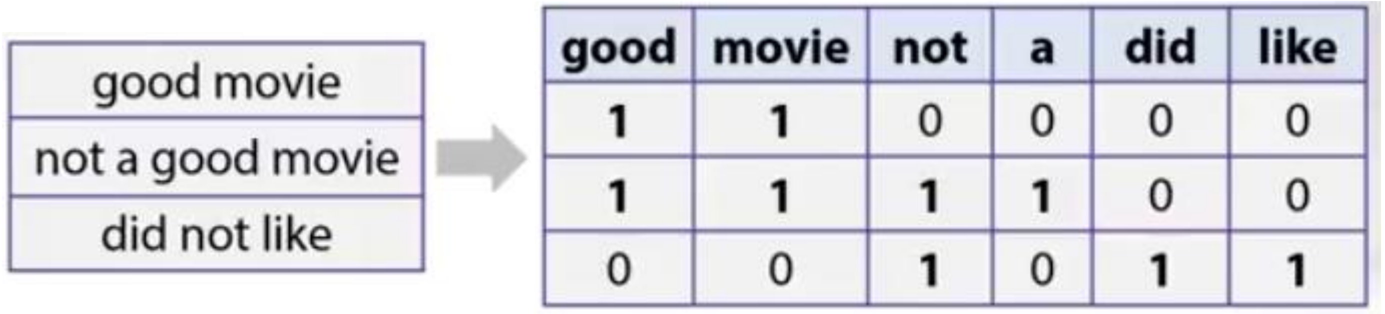
- Common vector representation
  - *bag-of-words*
  - *bag-of-n-grams*

bag-of-words

- A sentence or a document is represented as the bag of its words

BoW = {“good":2,“movie":2,“not":2,“a":1,“did":1,“like":1};

Text vectorization:



good movie
not a good movie
did not like

good	movie	not	a	did	like
1	1	0	0	0	0
1	1	1	1	0	0
0	0	1	0	1	1

words are equally distant!!

A bag-of-n-grams model

- Represents a sentence or a document as an unordered collection of its n-grams

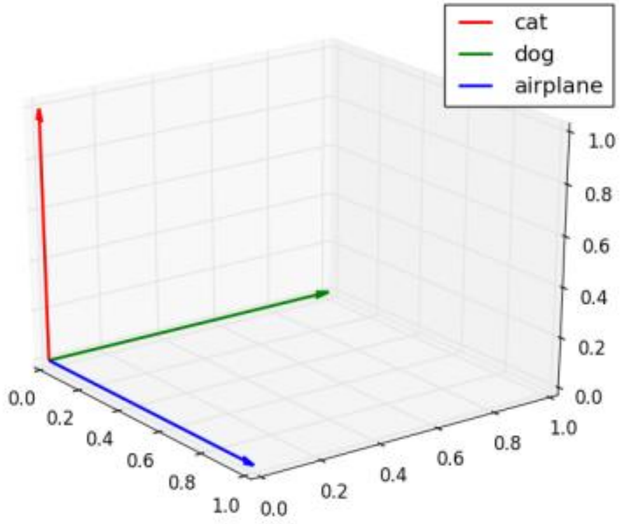
2-gram	frequency
Good movie	2
Not a	1
A good	1
Did not	1
Not like	1

Disadvantages of bag-of-words

- Lose the ordering of the words
- Ignore semantic of the words
- Suffers from sparsity and high dimensionality

Word Representations: Sparse

- Each word is represented by a one-hot representation.
- The dimension of the symbolic representation for each word is equal to the size of the vocabulary V.



Shortcomings of Sparse Representations

- There is no notion of similarity between words

  V = (cat, dog, airplane)

  V_{cat}= *(0, 0, 1)*

  V_{dog}= *(0, 1, 0)*

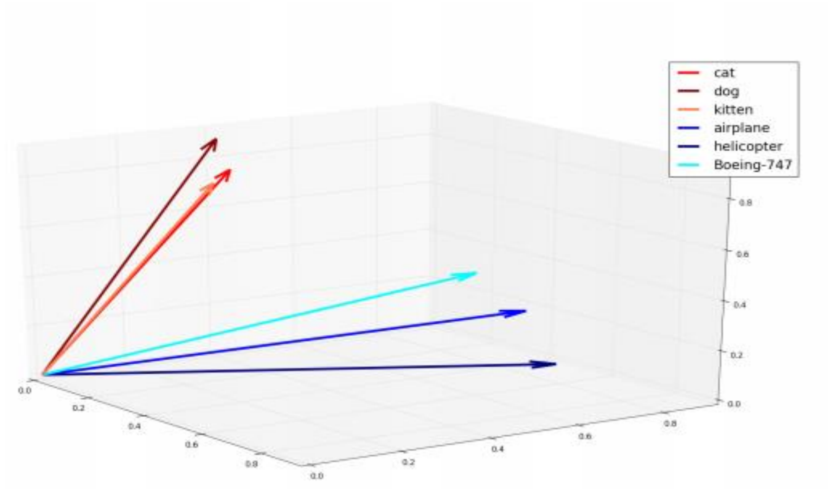
  $V_{airplane}$= *(1, 0, 0)*

  sim(cat, airplane) = sim(dog, cat) = sim(dog, airplane)

- The size of the dictionary matrix D

Word Representations: Dense

- Each word is represented by a dense vector, a point in a vector space
- The dimension of the semantic representation d is usually much smaller than the size of the vocabulary (d << V)



Word and Document Embedding

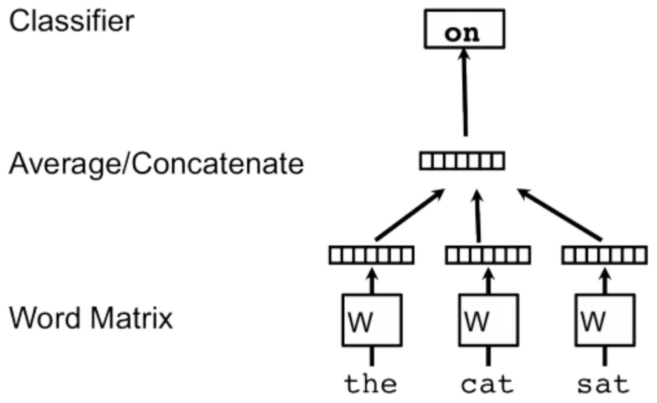
- Learning word vectors
  - *the cat sat on ------- . mat*
- Learning paragraph vectors
  - topic of the document = ‘technology’’
    - *Catch the …… .*   *Exception*
  - topic of the document = *‘’sports’’*
    - *Catch the …… .*   *Ball*

Learning Vector Representation of Words

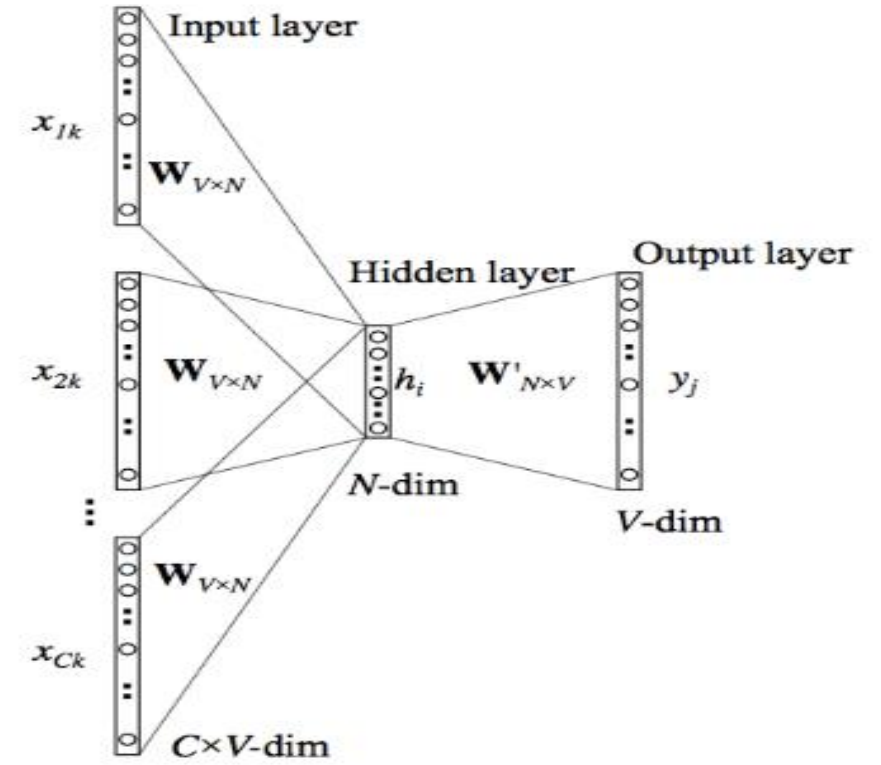
- Unsupervised algorithm
- Learns fixed-length feature representation of words from variable-length pieces of texts
- Trained to be useful for predicting words in a context
- This algorithm represents each word by a dense vector

Learning Vector Representation of Words(CBOW)

- **Task**: predict a word given the other words in a context
- Every word is mapped to a unique vector, represented by a column in a matrix W.
- The concatenation or sum of the vectors is then used as features for prediction of the next word in a sentence.



Learning Vector Representation of Words(CBOW)



Learning Vector Representation of Words

Given a sequence of training words : $W_1, W_2, W_3, \dots, W_T$

Objective: maximize the average log probability

$$\frac{1}{T}\sum_{t=k}^{T-k} \log p(w_t | w_{t-k}, \dots, w_{t+k})$$

Learning Vector Representation of Words

- The prediction task is typically done via a multiclass classifier, such as softmax

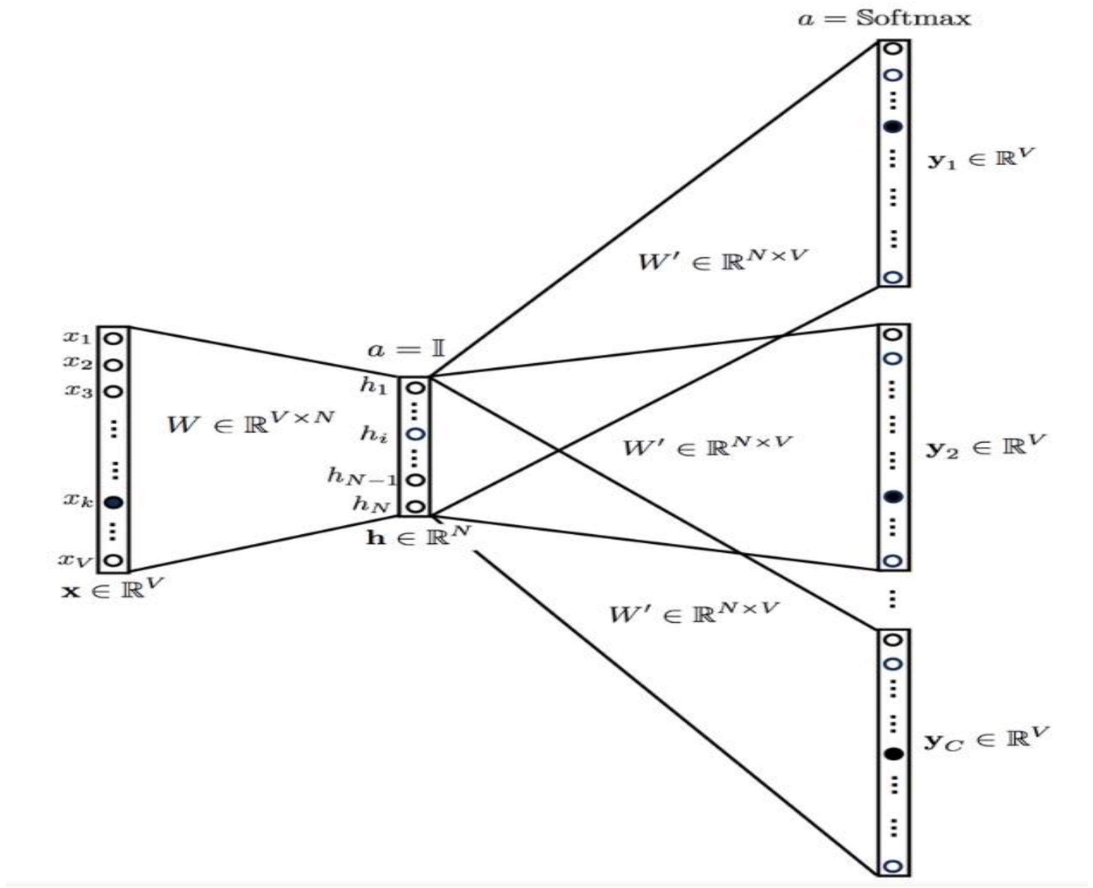
$$p(w_t | w_{t-k}, \ldots, w_{t+k}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}}$$

- Each of y_i is un-normalized log-probability for each output word i, computed as

$$y = b + Uh(w_{t-k}, \ldots, w_{t+k}; W)$$

- U, b are the softmax parameters. h is constructed by a concatenation or average of word vectors extracted from W.

Learning Vector Representation of Words(Skipgram)


$a = \text{Softmax}$
$\mathbf{y}_1 \in \mathbb{R}^V$
$W' \in \mathbb{R}^{N \times V}$
x_1
x_2
x_3
x_k
x_V
$\mathbf{x} \in \mathbb{R}^V$
$W \in \mathbb{R}^{V \times N}$
$a = \mathbb{I}$
h_1
h_i
h_{N-1}
h_N
$\mathbf{h} \in \mathbb{R}^N$
$W' \in \mathbb{R}^{N \times V}$
$\mathbf{y}_2 \in \mathbb{R}^V$
$W' \in \mathbb{R}^{N \times V}$
$\mathbf{y}_C \in \mathbb{R}^V$

Paragraph Vector – related work

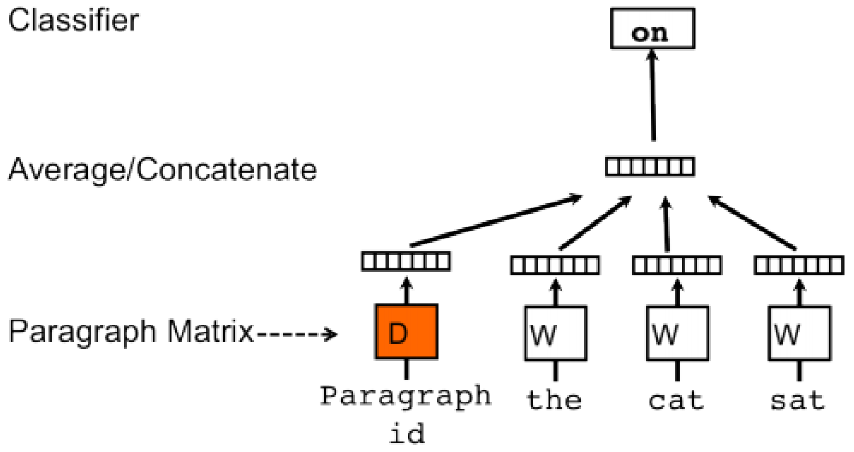
- Extending the models to go beyond word level to achieve phrase-level or sentence-level representations
  - A simple approach is using a weighted average of all the words in the document.
    - Weakness: loses the word order in the same way as the standard bag-of-words models do
  - A more sophisticated approach is combining the word vectors in an order given by a parse tree of a sentence, using matrix-vector operations (Socher et al., 2011b)
    - Weakness: work for only sentences because it relies on parsing

Paragraph Vector: A distributed memory model(PV-DM)

- Unsupervised algorithm
- Learns fixed-length feature representation from variable-length pieces of texts (e.g. sentences, paragraphs and documents)
- This algorithm represents each document by a dense vector
- The paragraph vectors are also asked to contribute to the prediction task of the next word given many contexts sampled from the paragraph.

Paragraph Vector: A distributed memory model(PV-DM)

- It acts as a memory that remembers what is missing from the current context – or the topic of the paragraph.



Paragraph Vector: A distributed memory model(PV-DM)

- The paragraph vector is shared across all contexts generated from the same paragraph but not across paragraphs.

- The word vector matrix W , however, is shared across paragraphs.

  (i.e. the vector for “powerful” is the same for all paragraphs)

Two key stages of this algorithm

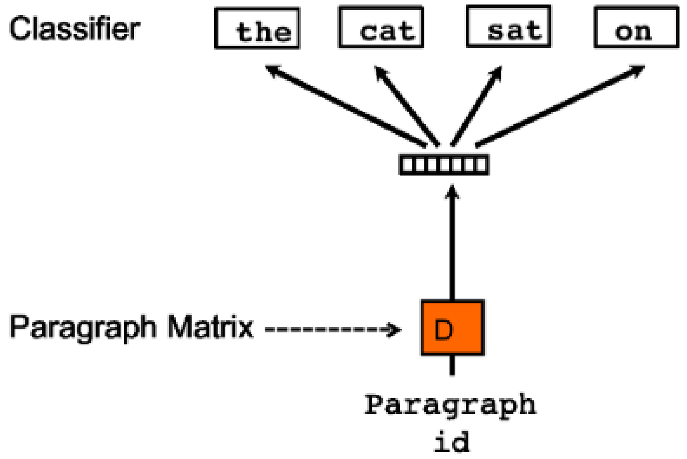
- training to get word vectors W, softmax weights U, b and paragraph vectors D on already seen paragraphs.
- "the inference stage" to get paragraph vectors D for new paragraphs (never seen before) by adding more columns in D

- After being trained feed these features directly to machine learning techniques

Paragraph Vector without word ordering: Distributed bag of words(PV-DBOW)

- Another way is to ignore the context words in the input, but force the model to predict words randomly sampled from the paragraph in the output

- At each iteration of stochastic gradient descent, we sample a text window, then sample a random word from the text window and form a classification task given the Paragraph Vector



Advantages of paragraph vectors

- They are learned from unlabeled data

- Paragraph vectors also address some of the key weaknesses of bag-of-words models
  - the semantics of the words
  - They take into consideration the word order

Limitations of paragraph vectors

- Sometimes information captured in the paragraph vectors is unclear and difficult to interpret

- Quality of the vectors is also highly dependent on the quality of the word vectors

Experiments

- Each paragraph vector is supposed as a combination of two vectors: one learned by PV-DM and one learned by PV- DBOW

- PV-DM alone usually works well for most tasks, but its combination with PV-DBOW is usually more consistent

- Experiments show benchmark of Paragraph Vector on two text understanding problems that require fixed-length vector representations of paragraphs
  - sentiment analysis
  - information retrieval

Sentiment Analysis with the Stanford Sentiment Treebank Dataset

- This dataset has 11,855 sentences taken from the movie review site **Rotten Tomatoes**
- The dataset consists of three sets: 8,544 sentences for **training**, 2,210 sentences for **test** and 1,101 sentences for **validation**
- Every sentence and its sub-phrases in the dataset has a label. The labels are generated by human annotators using Amazon Mechanical Turk
    - a 5-way fine-grained classification {Very Negative, Negative, Neutral, Positive, Very Positive}
    - a 2-way coarse-grained classification {Negative, Positive}
- There are 239,232 labeled phrases in the dataset

Sentiment Analysis with the Stanford Sentiment Treebank Dataset(Experimental protocols)

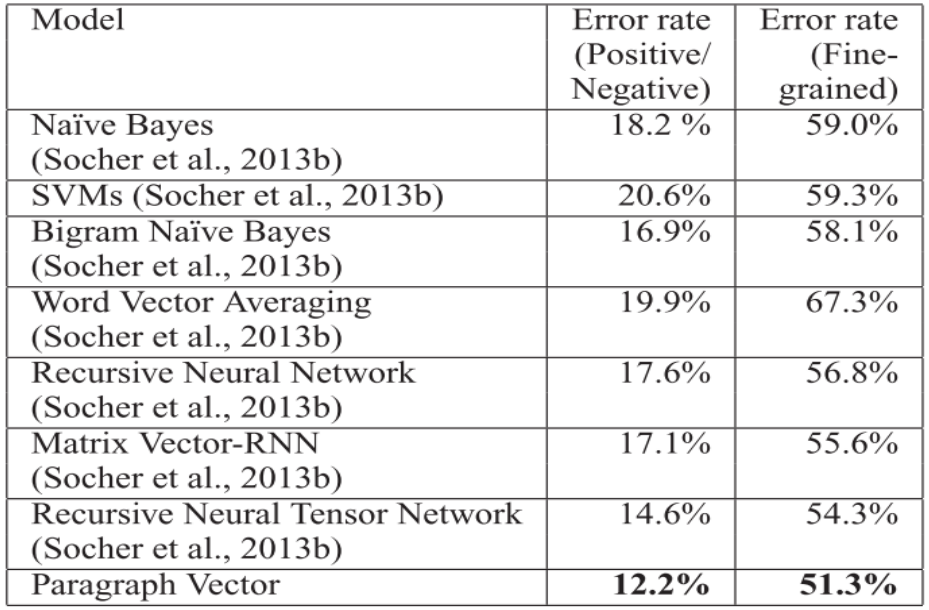
- Vector representations are learned and then fed to a logistic regression model to learn a predictor of the movie rating.

- At test time, the vector representation for each word is frozen, and representations for the sentences are learnt using gradient descent and fed to the logistic regression to predict the movie rating.

- The optimal window size is 8.

Sentiment Analysis with the Stanford Sentiment Treebank Dataset(Results)

Table 1. The performance of our method compared to other approaches on the Stanford Sentiment Treebank dataset. The error rates of other methods are reported in (Socher et al., 2013b).



Model	Error rate (Positive/ Negative)	Error rate (Fine-grained)
Naïve Bayes (Socher et al., 2013b)	18.2 %	59.0%
SVMs (Socher et al., 2013b)	20.6%	59.3%
Bigram Naïve Bayes (Socher et al., 2013b)	16.9%	58.1%
Word Vector Averaging (Socher et al., 2013b)	19.9%	67.3%
Recursive Neural Network (Socher et al., 2013b)	17.6%	56.8%
Matrix Vector-RNN (Socher et al., 2013b)	17.1%	55.6%
Recursive Neural Tensor Network (Socher et al., 2013b)	14.6%	54.3%
Paragraph Vector	**12.2%**	**51.3%**

Sentiment Analysis with IMDB dataset

- The dataset consists of 100,000 movie reviews taken from **IMDB**. The 100,000 movie reviews are divided into three datasets:
  - 25,000 labeled **training** instances, 25,000 labeled **test** instances and 50,000 unlabeled training instances.

- There are two types of labels: **Positive and Negative**. These labels are balanced in both the training and the test set.

Beyond One Sentence: Sentiment Analysis with IMDB dataset(Experimental protocols)

- Word vectors and paragraph vectors are learnt using training documents.
- The paragraph vectors for the labeled instances of training data are then fed through a neural network to learn to predict the sentiment.
- At test time, given a test review, the rest of the network is frozen and paragraph vectors are learnt for the test reviews by gradient descent and fed to the neural network to predict the sentiment of the reviews.
- The optimal window size is 10 words

Beyond One Sentence: Sentiment Analysis with IMDB dataset(results)

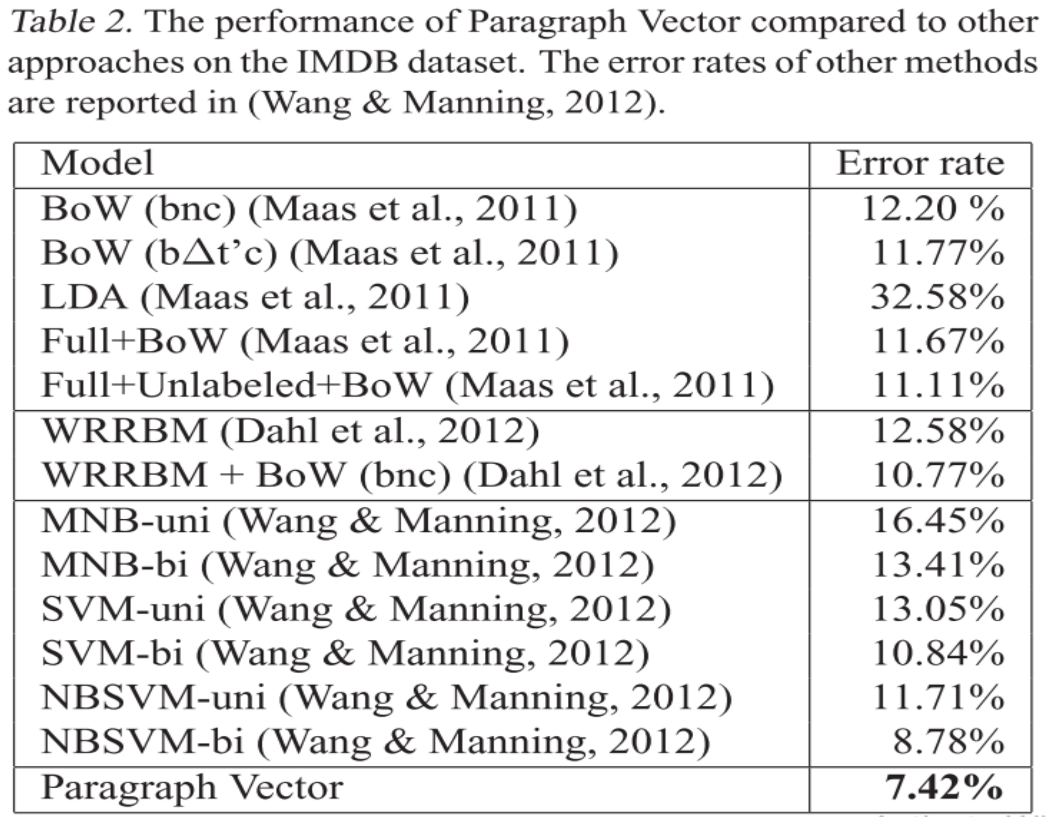


Table 2. The performance of Paragraph Vector compared to other approaches on the IMDB dataset. The error rates of other methods are reported in (Wang & Manning, 2012).

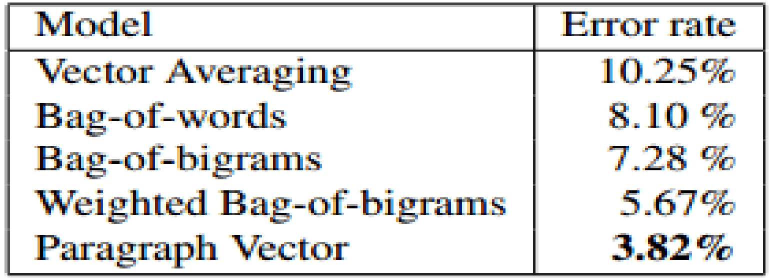
Model	Error rate
BoW (bnc) (Maas et al., 2011)	12.20 %
BoW (bΔt'c) (Maas et al., 2011)	11.77%
LDA (Maas et al., 2011)	32.58%
Full+BoW (Maas et al., 2011)	11.67%
Full+Unlabeled+BoW (Maas et al., 2011)	11.11%
WRRBM (Dahl et al., 2012)	12.58%
WRRBM + BoW (bnc) (Dahl et al., 2012)	10.77%
MNB-uni (Wang & Manning, 2012)	16.45%
MNB-bi (Wang & Manning, 2012)	13.41%
SVM-uni (Wang & Manning, 2012)	13.05%
SVM-bi (Wang & Manning, 2012)	10.84%
NBSVM-uni (Wang & Manning, 2012)	11.71%
NBSVM-bi (Wang & Manning, 2012)	8.78%
Paragraph Vector	**7.42%**

Information Retrieval with Paragraph Vectors

- Requires fixed-length representations of paragraphs
- A dataset of paragraphs the first 10 results returned by a search engine given each of 1,000,000 most popular queries
- Summarizes the content of a web page and how a web page matches the query

Information Retrieval with Paragraph Vectors

- A triplet of paragraphs
  - Two paragraphs are results of the same query
  - One paragraph is a the result of a different query
- Goal is to identify which of the three paragraphs are results of the same query



Model	Error rate
Vector Averaging	10.25%
Bag-of-words	8.10 %
Bag-of-bigrams	7.28 %
Weighted Bag-of-bigrams	5.67%
Paragraph Vector	**3.82%**

Recap

- Paragraph Vector, an unsupervised learning algorithm that learns vector representations for variable- length pieces of texts such as sentences and documents

- This algorithm overcomes many weaknesses of bag-of-words models

Resources

- https://www.eecs.yorku.ca/course_archive/2016-17/W/6412/reading/DistributedRepresentationsofSentencesandDocuments.pdf
- https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa
- https://www.fer.unizg.hr/_download/repository/TAR-07-WENN.pdf

Thank You!