Distributed Representations of Sentences and Documents

Quoc Le, Tomas Mikolov(2014)

박성진

References

- PR-76 논문리뷰(곽근봉)

  https://youtu.be/NxKpgY6sWOQ

- Doc2Vec tutorial

  https://github.com/RaRe-Technologies/gensim/blob/develop/docs/notebooks/doc2vec-lee.ipynb

Abstract

- In this paper, we propose Paragraph Vector, an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts, such as sentences, paragraphs, and documents.

- Our algorithm represents each document by a dense vector which is trained to predict words in the document.

- PV-DM, PV-DBOW
  (paragraph Vector– Distributed Memory , Bag of words)

1. Introduction – Problems of BOW(Bag of Words)

- The word order is lost, and thus different sentences can have exactly the same representation

- very little sense about the semantics of the words or more formally the distances between the words

- This means that words “powerful”, “strong” and “Paris” are equally distant despite the fact that semantically, “powerful” should be closer to “strong” than “Paris”

1. Introduction – Solution

- we propose Paragraph Vector, an unsupervised framework that learns continuous distributed vector representations for pieces of texts.

- the vector representation is trained to be useful for predicting words in a paragraph.

- Our technique is inspired by W2V

2. **Algorithms** - Learning Vector Representation of Words

- W2V – predict next word

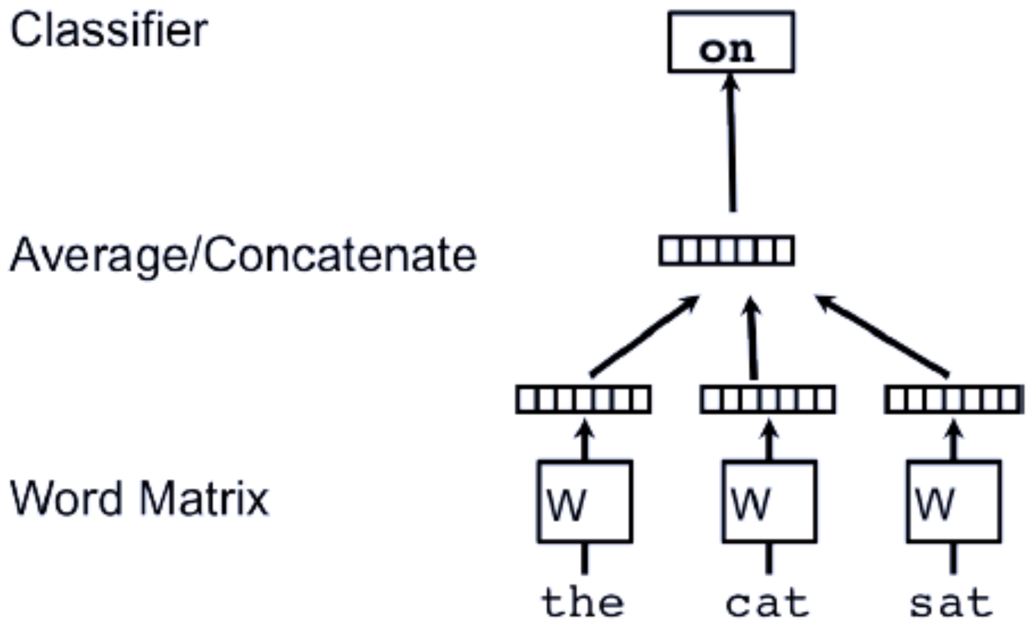


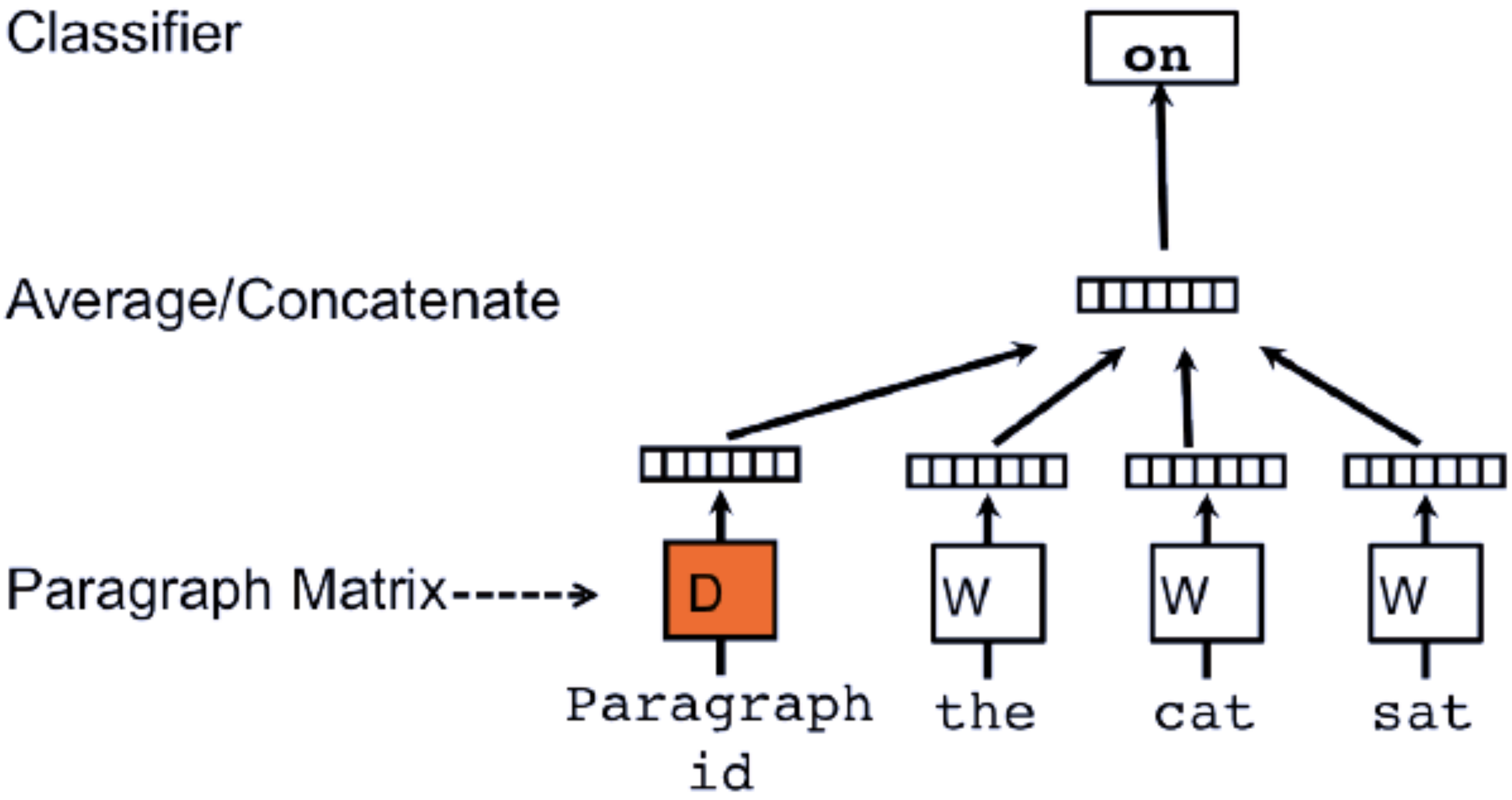
Figure 1. A framework for learning word vectors. Context of three words ("the," "cat," and "sat") is used to predict the fourth word ("on"). The input words are mapped to columns of the matrix *W* to predict the output word.

2. Algorithms - Paragraph Vector: A distributed memory model

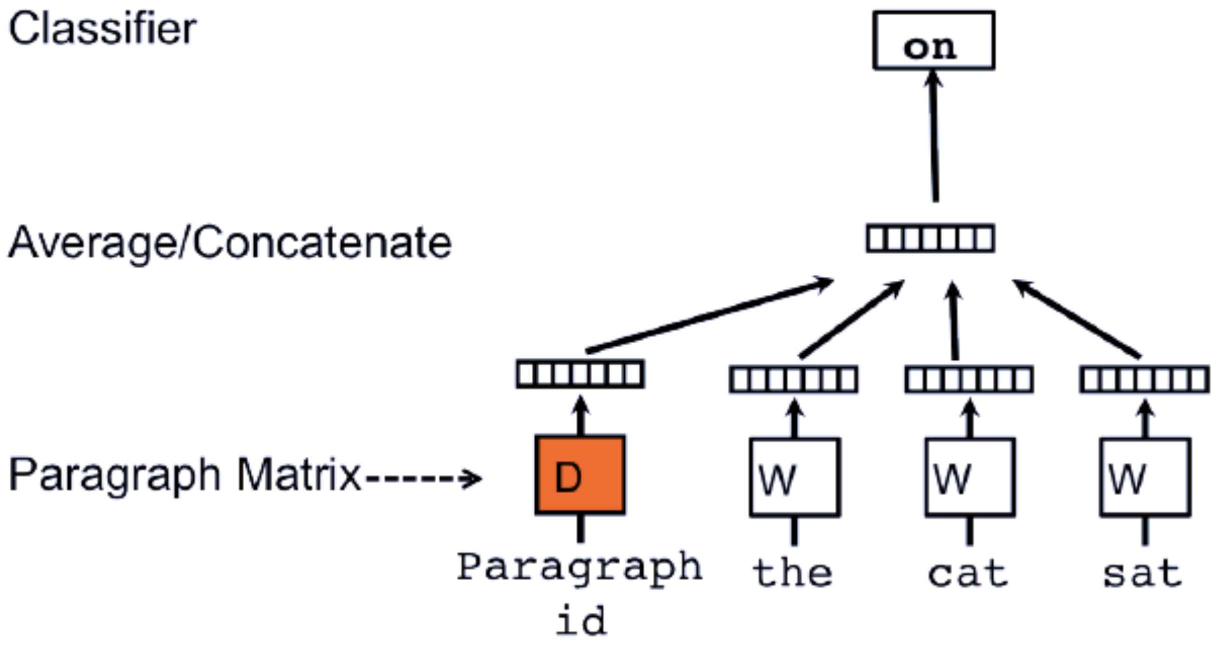
- Our approach for learning paragraph vectors is inspired by the methods for learning the word vectors.

- The paragraph vectors are also asked to contribute to the prediction task of the next word given many contexts sampled from the paragraph.

2. Algorithms PV-DM



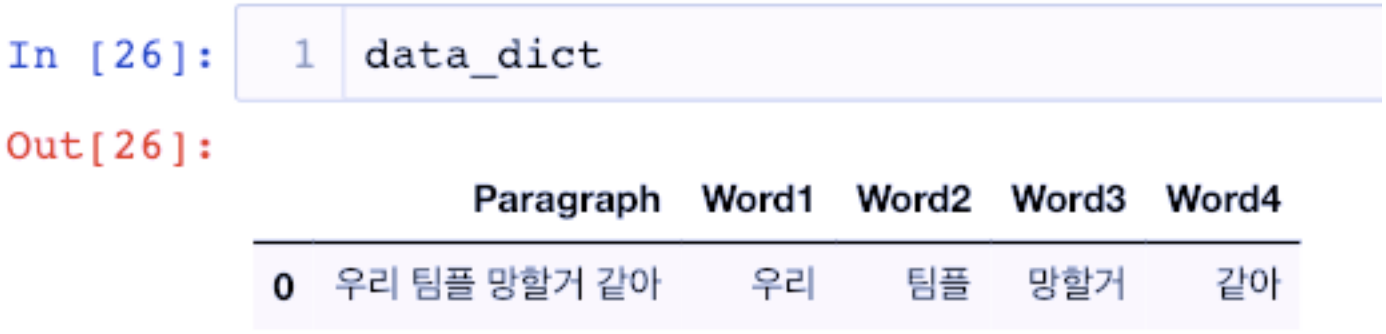
2. Algorithms PV-DM



- Paragraph is mapped to a unique vector, represented by a column in matrix D
- Word is also mapped to a unique vector, represented by a column in matrix W

2. Algorithms PV-DM

Paragraph: “우리 팀플 망할거 같아”

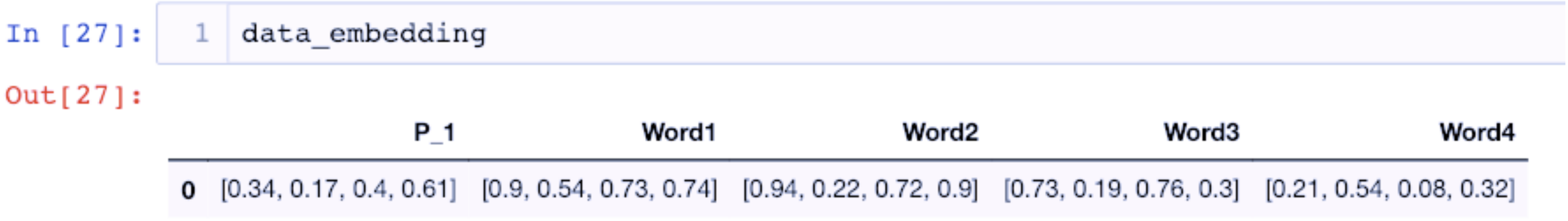


```
In [26]:  1  data_dict
```

Out[26]:

	Paragraph	Word1	Word2	Word3	Word4
0	우리 팀플 망할거 같아	우리	팀플	망할거	같아

- Paragraph, Word Dictionary



```
In [27]:  1  data_embedding
```

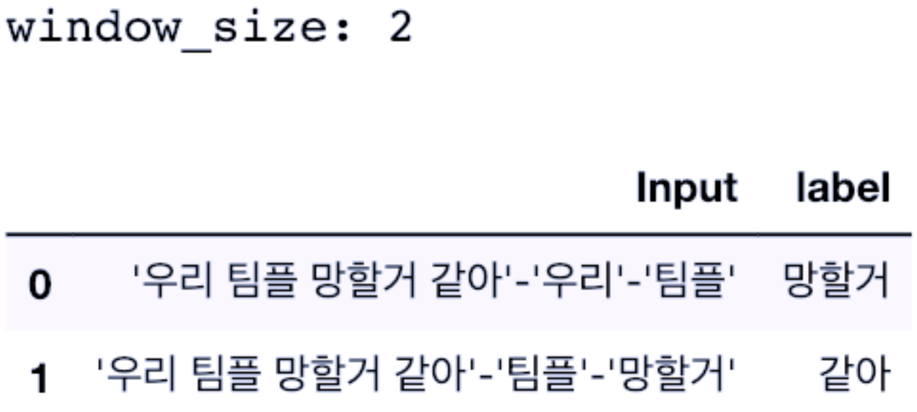
Out[27]:

	P_1	Word1	Word2	Word3	Word4
0	[0.34, 0.17, 0.4, 0.61]	[0.9, 0.54, 0.73, 0.74]	[0.94, 0.22, 0.72, 0.9]	[0.73, 0.19, 0.76, 0.3]	[0.21, 0.54, 0.08, 0.32]

- Paragraph, Word Embedding (random Initialization)

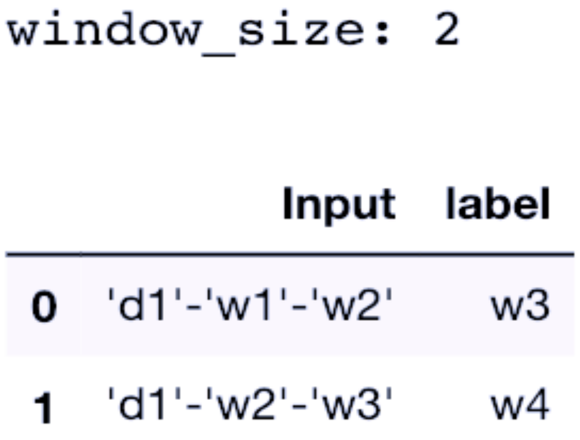
2. **Algorithms** PV-DM

Paragraph: “우리 팀플 망할거 같아”



```
window_size: 2
```

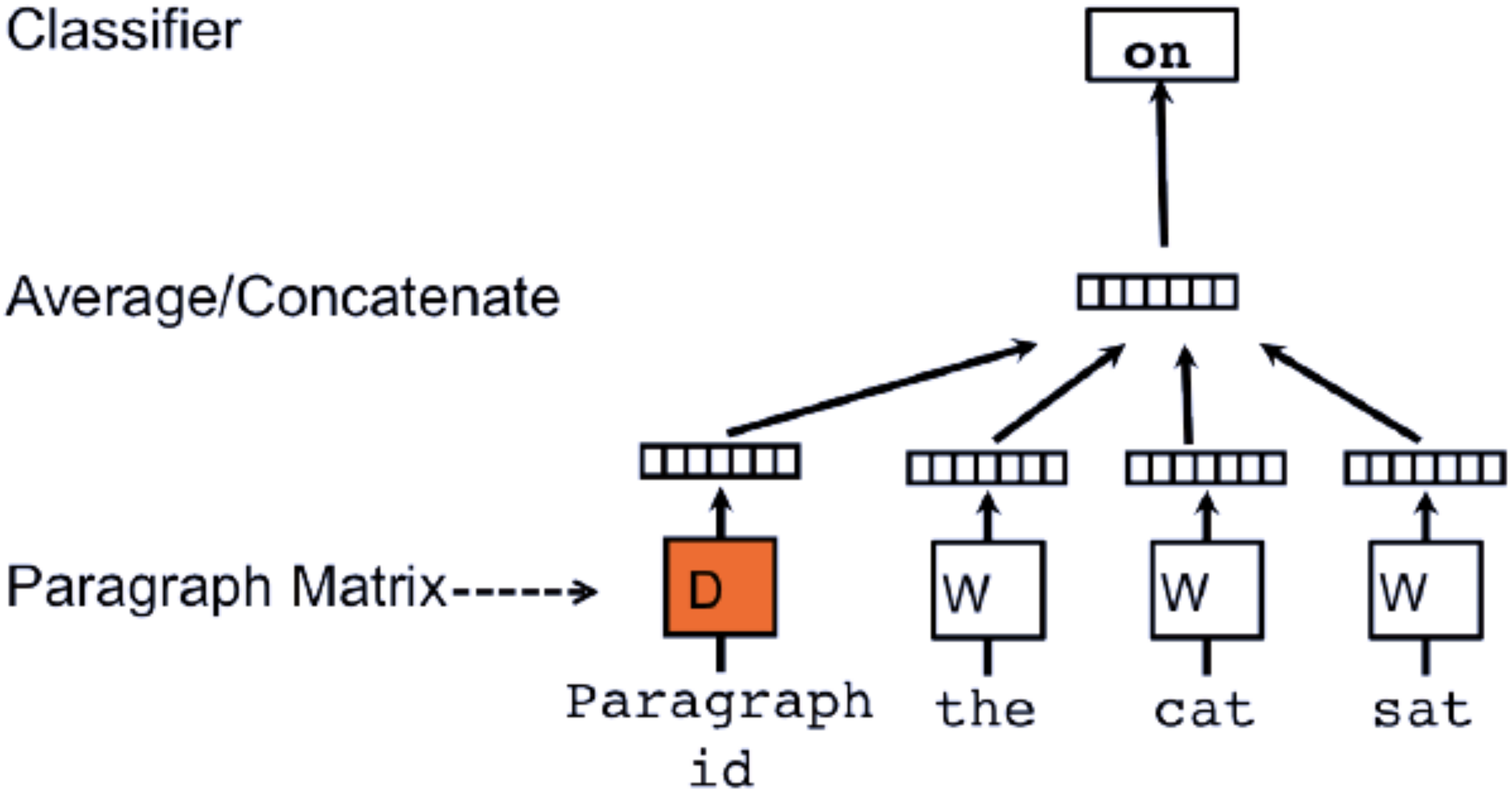
	Input	label
0	'우리 팀플 망할거 같아'-'우리'-'팀플'	망할거
1	'우리 팀플 망할거 같아'-'팀플'-'망할거'	같아



```
window_size: 2
```

	Input	label
0	'd1'-'w1'-'w2'	w3
1	'd1'-'w2'-'w3'	w4

2. Algorithms Training



2. Algorithms Training

- The additional paragraph token D

- Concatenation or average of this vector with a context of three words is used to predict the fourth word.

- The paragraph vector represents the missing information from the current context

- The paragraph vector can act as a memory of the topic of the paragraph.

2. Algorithms Training

- SGD -Backpropagation.

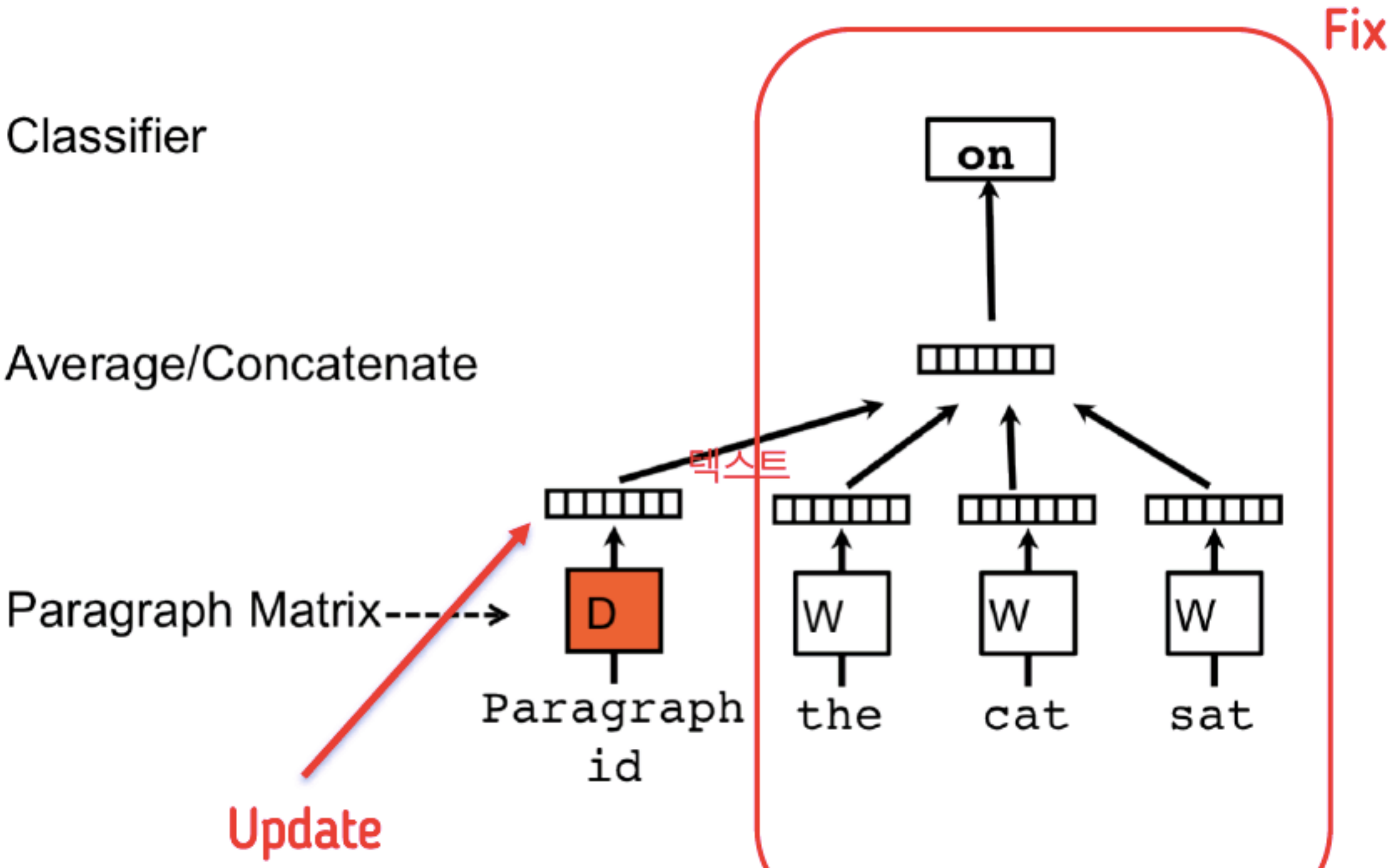
- At every step of stochastic gradient descent, one can sample a fixed-length context from a random paragraph, compute the error gradient from the network

- and use the gradient to update the parameters in our model.

2. Algorithms Training

- N paragraphs in the corpus, M words in the vocabulary
- Want to learn paragraph vectors such that each paragraph is mapped to p dimensions and each word is mapped to q dimensions : PV, WV
- Parameters: Np + Mq parameters (excluding the softmax parameters)
- After being trained, the paragraph vectors can be used as features for the paragraph

2. **Algorithms** Inference (predict)



2. Algorithms Inference (predict)

- At prediction time, one needs to perform an inference step to compute the paragraph vector for a new paragraph.

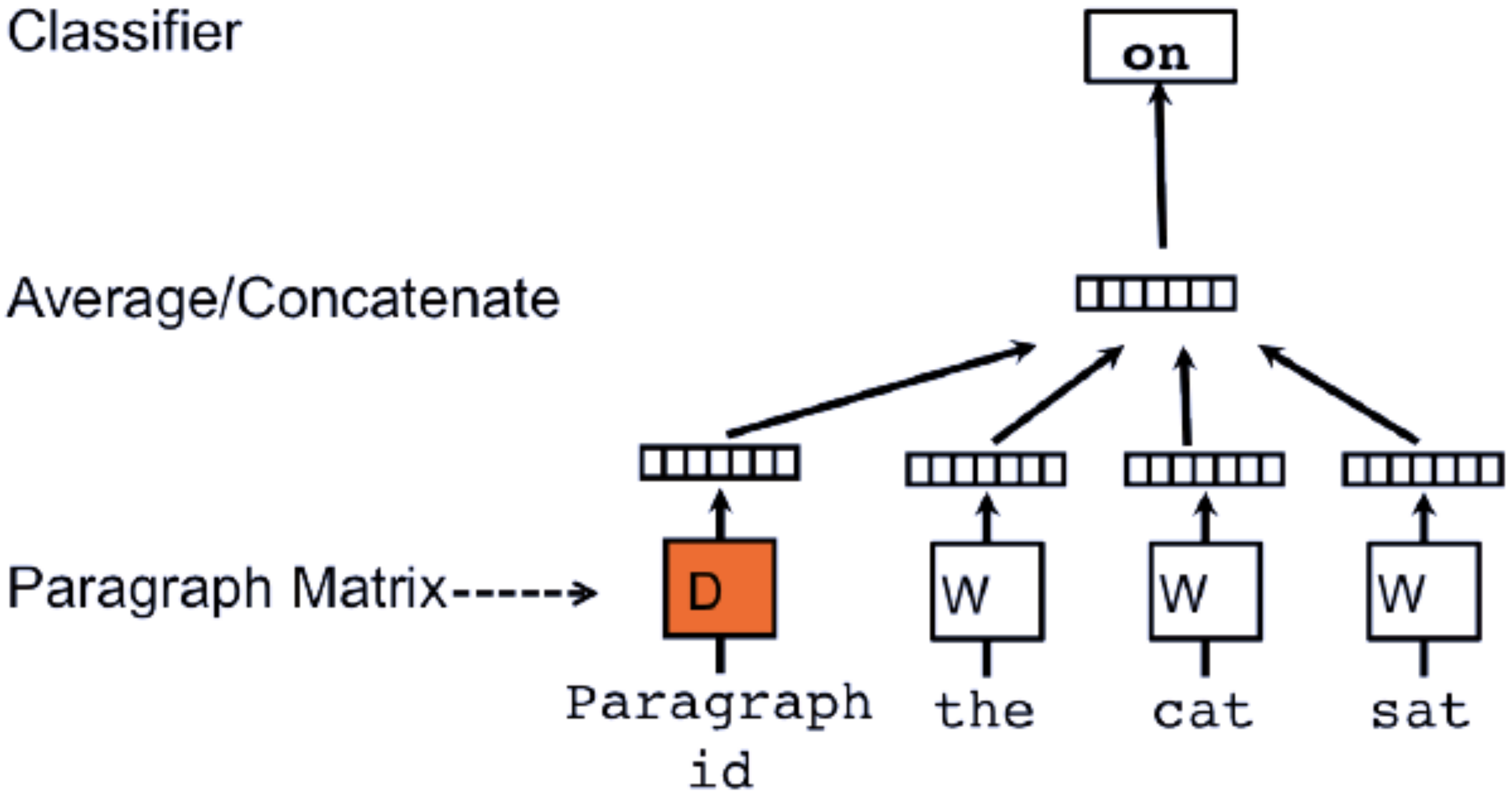
- This is also obtained by gradient descent. In this step, the parameters for the rest of the model, the word vectors W and the softmax weights, are fixed.
  (W = Mq , softmax weights)

2. **Algorithms –** Summary

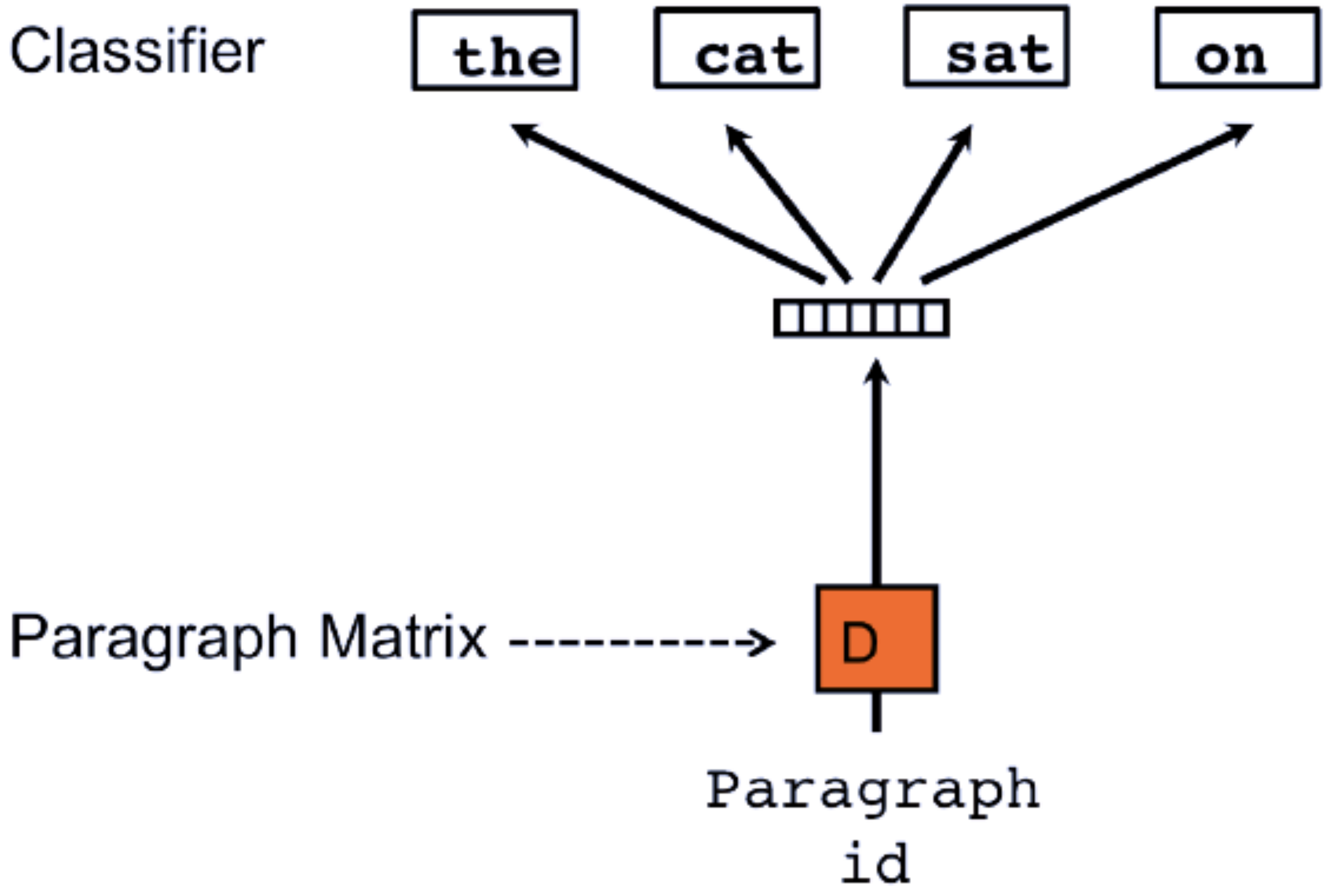
- 1) Training to get word vectors W, softmax weights U, b and paragraph vectors D on already seen paragraphs

- 2) “The inference stage” to get paragraph vectors D for new paragraphs (never seen before) by adding more columns in D and gradient descending on D while holding W, U, b fixed.

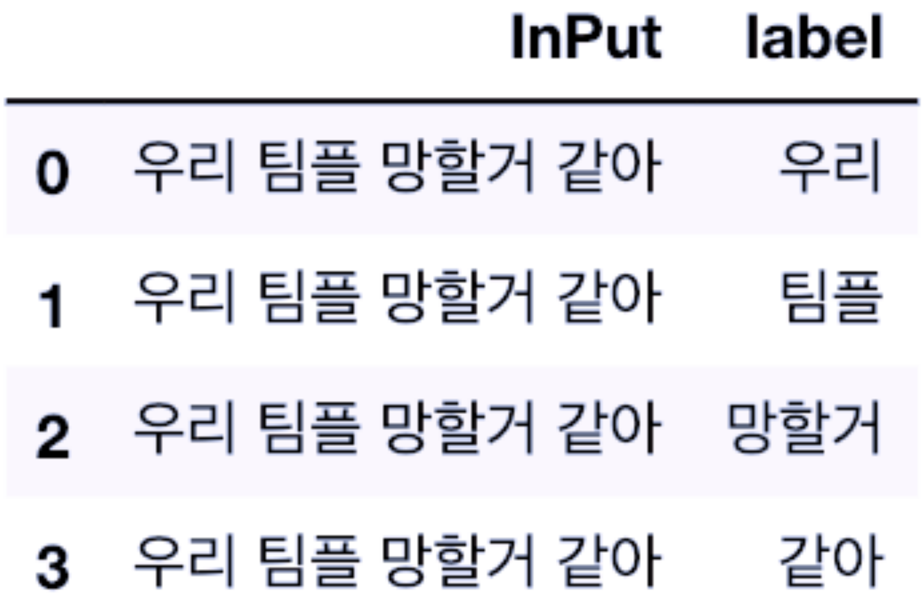
2. Algorithms PV-DM



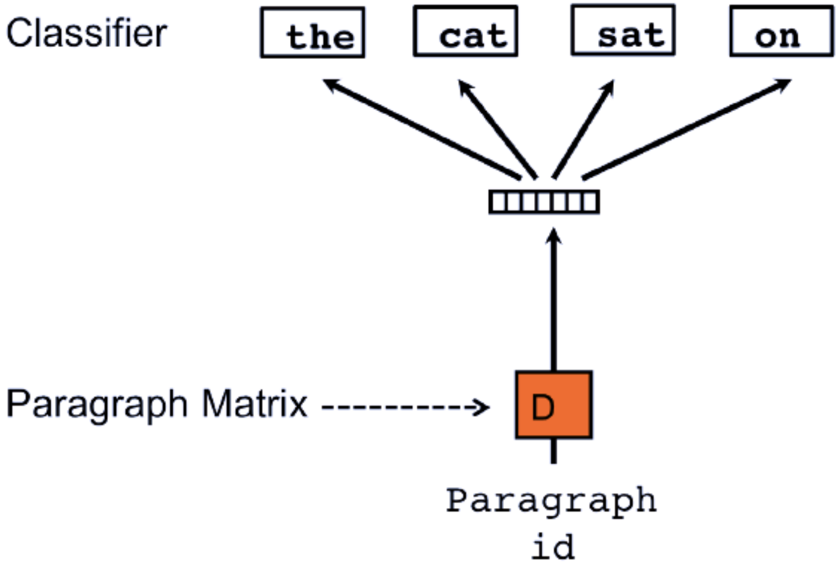
2. Algorithms PV-DBOW(like Skip_gram)



2. Algorithms PV-DBOW(like Skip_gram)



	InPut	label
0	우리 팀플 망할거 같아	우리
1	우리 팀플 망할거 같아	팀플
2	우리 팀플 망할거 같아	망할거
3	우리 팀플 망할거 같아	같아



2. Algorithms PV-DBOW (like SG)

- Ignore the context words in the input
- Predict words randomly sampled from the paragraph in the output.
- we sample a text window, then sample a random word from the text window and form a classification task given the Paragraph Vector.
- the paragraph vector is trained to predict the words in a small window.

3. Some further observations

- PV-DM is consistently better than PV-DBOW.
- Using concatenation in PV-DM is often better than sum.
- A good guess of window size in many applications is between 5 and 12.

감사합니다