Distributed Representations of Sentences and Documents

Quoc Le, Tomas Mikolov

Presented by Seonghyeon Kim

2018.04.26

Motivation

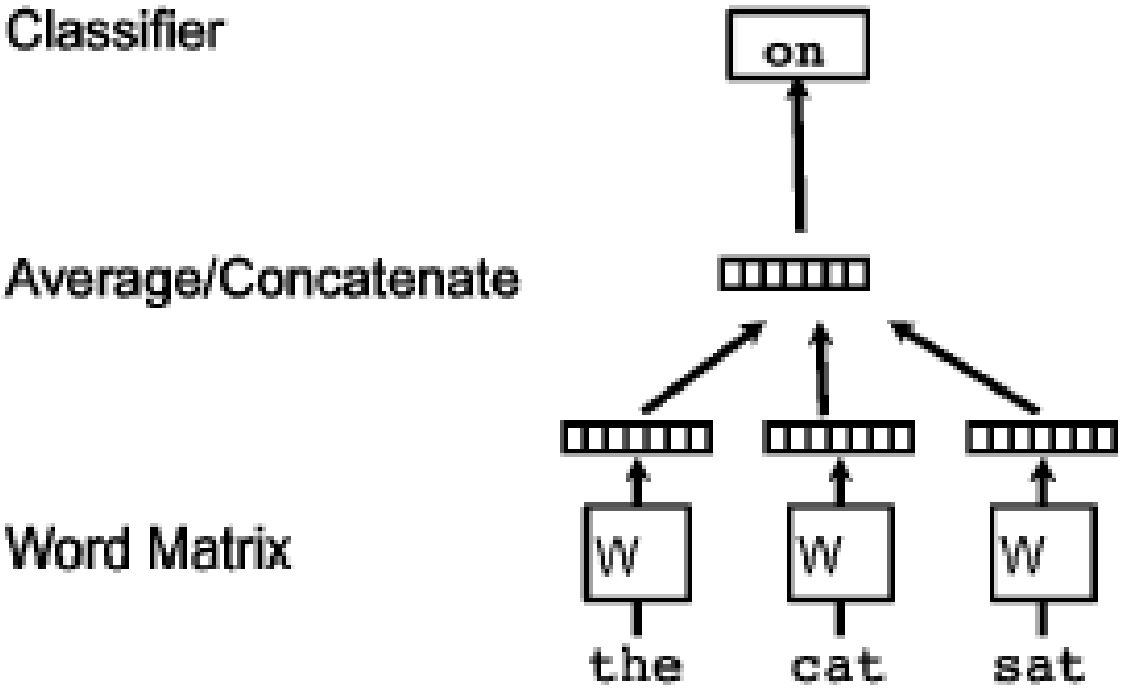


Figure 1. A framework for learning word vectors. Context of three words ("the," "cat," and "sat") is used to predict the fourth word ("on"). The input words are mapped to columns of the matrix W to predict the output word.

Motivation

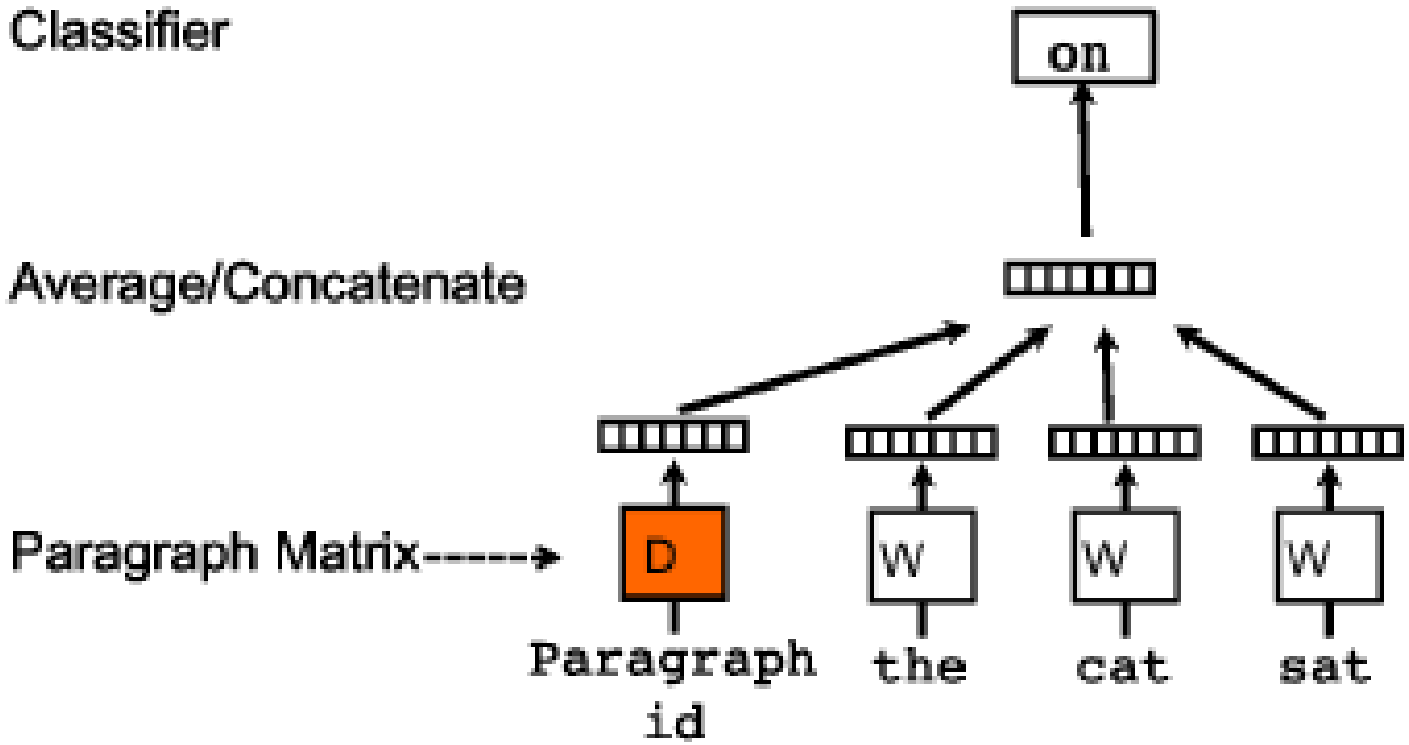
Learning Vector Representation of Words

$(x_1, \ldots, x_k, x_{k+1})$: word IDs of words in a window
$(w_1, \ldots, w_M)$: w_i is a vector corresponding to word of which ID is i.
b, U : softmax parameters
h : averaging(or concatenating) function

$$p(x_{k+1}|x_1, \ldots, x_k) = \frac{e^{z_{x_{k+1}}}}{\sum_{i=1}^{M} e^{z_i}}$$

$$z = b + Uh(w_{x_1}, \ldots, w_{x_k})$$

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



PV-DM

PV-DM

v : Paragraph ID

$(x_1, \ldots, x_k, x_{k+1})$: word IDs of words in a window

$(w_1, \ldots, w_M)$: w_i is a vector corresponding to word of which ID is i.

$(d_1, \ldots, d_N)$: d_i is a vector corresponding to ith paragraph.

b, U : softmax parameters

h : averaging(or concatenating) function

$$p(x_{k+1}|v, x_1, \ldots, x_k) = \frac{e^{z_{x_{k+1}}}}{\sum_{i=1}^{M} e^{z_i}}$$

$$z = b + Uh(w_{x_1}, \ldots, w_{x_k})$$

PV-DM

- They reflect the semantics of the words.
  (“powerful” is closer to “strong” than to “Paris.”)
- They take into consideration the word order, at least in a small context.
- Paragraph vector for a new paragraph can be obtained by gradient descent. At this point, W, b, U are fixed.

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

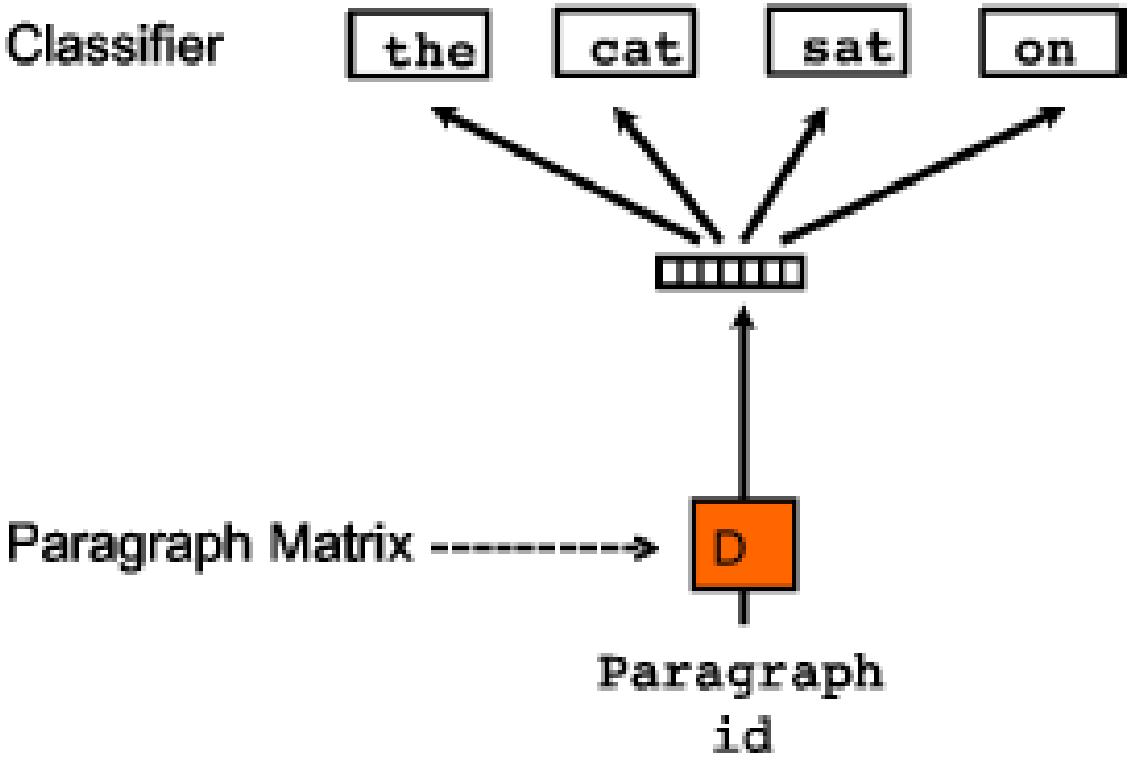


Figure 3. Distributed Bag of Words version of paragraph vectors. In this version, the paragraph vector is trained to predict the words in a small window.

PV-DBOW

- PV-DM alone usually works well for most tasks.
- But its combination with PV-DBOW is usually more consistent across many tasks.

Result

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%**