92586 Computational Linguistics

13. From document representations, towards sequences

Alberto Barrón-Cedeño

Alma Mater Studiorum-Università di Bologna
a.barron@unibo.it @_albarron_

30/04/2020

Previously

- Training and loading (existing) embeddings
- Visualisation

Table of Contents

Chapters 6 and 7 of Lane et al. (2019)

Doc2vec

Doc2vec

Objective Computing a vectorial representation of a document.

Same idea as with word2vec: a NN to predict words

Input

- *k* context words (optional)
- A unique ID of the sentence/paragraph/document

Output

- 1 target word

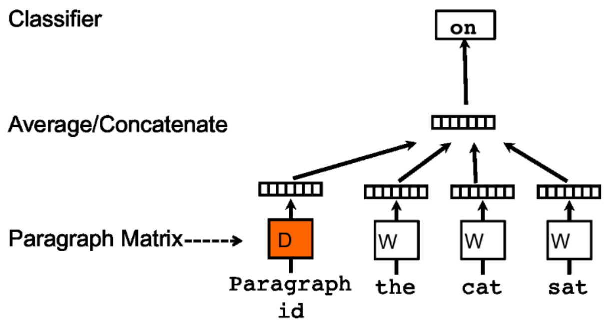
- The paragraph vector is unique among all documents
- The word vectors are shared among all documents
- The document vector is computed **on the fly**

Le and Mikolov (2014); (Lane et al., 2019, p. 215)

Doc2vec

Distributed Memory Model of Paragraph Vectors (PV-DM)

Derived from CBOW

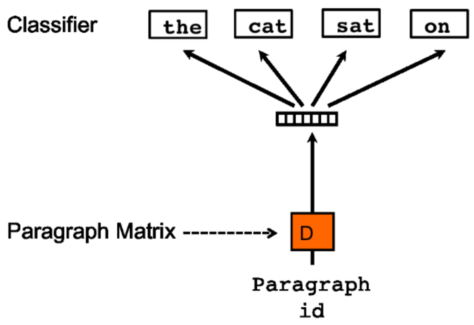


- Each column in the paragraph matrix is a vector representing one paragraph
- We can average or concatenate the word and paragraph vectors

Doc2vec

Distributed Bag of Words version of Paragraph Vector (PV-DBOW)

Similar to skip-gram



- Iteration: a text window and a random word from the text window are sampled, forming a classification task given the paragraph vector.
- No word vectors: faster + lower memory requirements

Let us see

Prologue to CNN and RNN

Prologue

- We have learned to build embedding spaces for words and texts
- We are considering the neighborhood of the words (the bag)
- We are not considering actual connections yet
- The downstream application is usually classification or regression

We will start heading towards text generation

Words have relations and influence each other

Word order

$$s_1 = \text{The dog chased the cat.}$$
$$s_2 = \text{The cat chased the dog.}$$

$$\begin{aligned} sim(tfidf(s_1), tfidf(s_2)) &= 1 \\ sim(wv(s_1), wv(s_2)) &= 1 \\ sim(dv(s_1), dv(s_2)) &= 1 \end{aligned}$$

But s_1 and s_2 are not the same!

Word proximity

$s =$ His mother, besides her son's willingness to amend the issue, decided to punish him

mother. . . decided | son. . . him

(Lane et al., 2019, p. 220)

Words have relations and influence each other

Spatial relation
Consider the position of words (∼written)

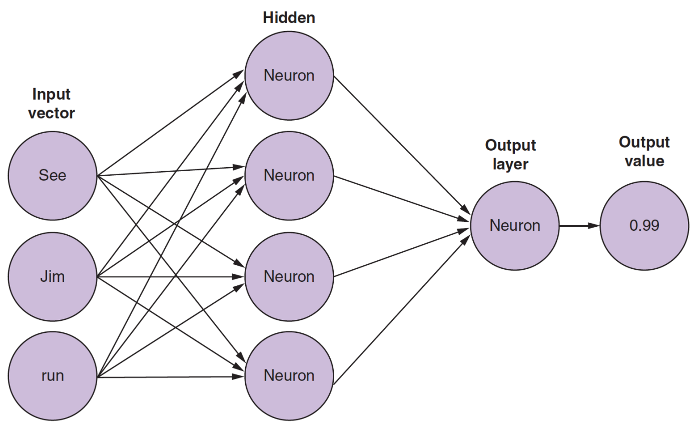
→ fixed-width window
convolutional neural networks

Temporal relation
Consider words as time series (∼spoken)

→ ongoing (unk) amount of time
recurrent neural networks

(Lane et al., 2019, p. 220)

Multiple Input Words



- Three tokens are passed at a time
- Two input alternatives
  - one-hot vector
  - pre-trained word vector

`See Jim run` $\neq$ `run See Jim` (!)

Back to Keras

Sequential()

- Python class
- Neural network abstraction
- Grants access to the basic Keras API

Sequential.compile()

- Builds the underlying weights
- Builds the and the interconnected relationships

Sequential.fit()

- Computes the training errors (loss)
- Applies backpropagation (weight adjustment)

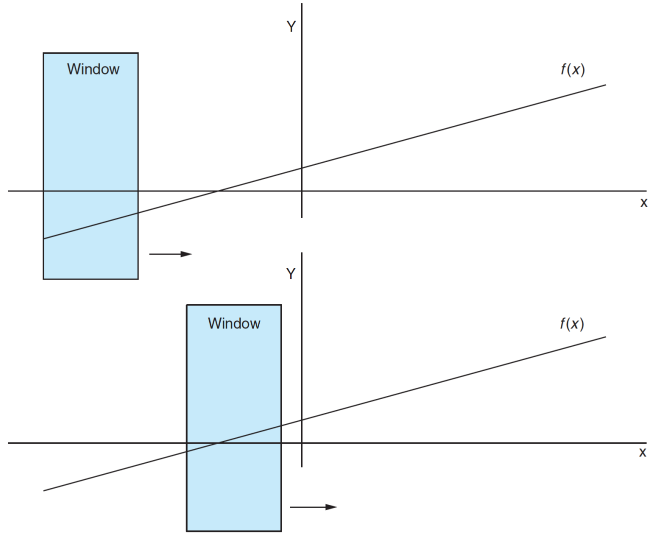
Some "cooking" hyperparameters

epochs	number of iterations over the data
batch_size	number of instances before adjusting
optmizer	function

CNN

Convolutional Neural Networks

Sliding —or convolving[1]— a window over the sample

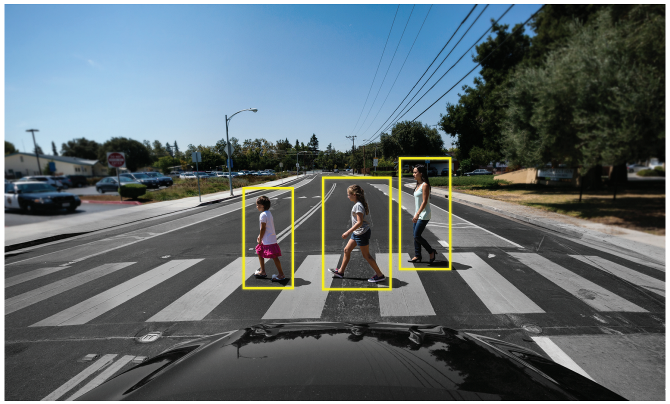


[1] To roll or wind together (Webster's)
(Lane et al., 2019, p. 222)

Convolutional Neural Networks

Back to the roots: image recognition

- Input: pixels of an image
- Output: the image contains x

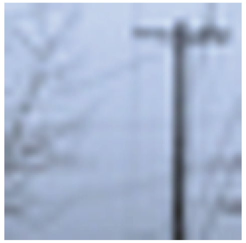


https://blogs.nvidia.com/wp-content/uploads/2019/04/ADAS-IMG_0052.jpg

Convolutional Neural Networks

When the input is an image

- B&W: [0,1] (with a smooth binariser)
- Grayscaled: [0, 255]
- Colour: R: [0, 255] G: [0, 255] B: [0, 255]



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
0	120	119	118	108	103	111	113	115	111	117	120	120	120	119	121	114	118	120	120	120	121	121	121	115	100	120	118	117
1	111	109	111	106	107	118	120	110	106	117	120	119	119	119	121	114	118	119	118	119	118	114	119	109	102	122	114	117
2	109	114	121	116	108	119	118	104	111	119	119	119	119	119	119	110	119	121	122	124	73	92	128	100	87	119	115	119
3	109	102	114	117	108	116	108	111	117	118	118	117	119	117	118	79	75	80	76	87	36	63	86	51	55	72	91	120
4	108	110	100	116	111	95	104	110	114	117	117	117	117	117	117	106	107	108	94	90	42	54	70	62	86	70	96	118
5	103	112	109	98	100	93	111	112	115	113	117	117	117	116	116	112	118	116	114	110	41	81	110	102	119	114	119	117
6	111	111	112	104	93	106	119	115	108	110	116	115	116	117	115	107	108	108	111	118	49	87	109	100	115	110	115	115
7	111	111	109	105	103	110	103	103	106	111	115	115	114	110	107	103	113	114	115	115	48	87	105	105	114	111	116	115
8	112	105	108	94	89	102	95	106	111	113	111	108	106	109	112	109	114	113	114	115	45	83	104	108	111	112	114	114
9	112	106	108	86	95	109	104	103	106	104	108	109	114	114	113	108	112	113	113	114	43	80	104	111	109	113	114	113
10	99	111	102	88	111	107	101	101	106	111	112	112	110	113	111	107	112	113	112	112	43	79	106	110	108	114	112	112
11	110	106	93	96	108	107	110	109	111	112	108	107	111	112	111	107	111	111	112	110	38	78	109	108	108	111	114	103
12	101	93	76	101	103	107	107	108	110	107	103	111	111	110	109	106	112	111	111	109	37	82	108	107	111	113	111	100
13	98	92	99	115	108	108	111	106	100	98	106	108	109	110	107	106	109	108	109	107	37	78	106	103	106	108	103	98
14	100	73	97	102	92	95	93	89	89	97	103	103	106	106	103	101	106	104	106	103	33	75	105	103	108	108	98	107
15	84	69	92	87	85	92	89	95	98	100	107	107	107	108	106	104	108	107	109	105	29	74	107	107	110	106	99	109
16	71	82	87	85	78	89	106	104	99	106	106	105	106	105	104	103	106	106	107	103	21	72	106	106	109	100	102	109
17	67	87	64	68	84	89	98	96	99	104	104	104	104	105	103	101	105	103	106	102	23	76	103	103	106	98	108	101
18	68	82	84	97	92	81	84	84	90	98	98	102	102	103	100	99	103	101	103	92	16	76	98	98	89	86	92	73
19	60	71	77	77	80	88	92	91	93	96	96	101	100	101	100	98	101	101	104	92	13	64	93	89	81	89	81	79
20	84	98	87	94	101	100	101	101	103	101	101	100	101	103	98	98	100	96	97	87	12	71	100	97	93	105	91	101
21	92	80	88	99	100	98	100	100	100	97	98	97	97	96	92	91	92	93	96	87	13	79	105	95	98	96	89	100
22	77	80	88	92	96	97	96	95	93	92	93	92	92	92	91	94	96	95	98	89	12	79	103	91	100	89	98	104
23	81	87	83	84	89	89	91	87	90	92	93	95	94	96	94	95	98	94	93	84	7	74	89	86	90	70	81	73
24	60	66	82	92	90	90	87	90	94	94	94	94	93	94	92	93	94	95	95	81	0	76	90	92	81	77	65	58
25	87	81	83	86	87	84	90	92	92	92	92	92	92	92	91	92	92	91	92	77	4	73	91	92	81	95	96	95
26	87	88	88	83	85	91	91	89	90	91	92	91	91	91	89	90	92	90	89	73	8	66	92	83	84	92	91	91
27	81	86	88	91	89	89	89	89	89	88	88	89	89	89	84	83	79	80	87	74	0	60	89	77	90	91	90	89

(Lane et al., 2019, p. 223)

Convolutional Neural Networks

When the input is an image

An image is just a bunch of numbers

- Appropriate as input for an NN
- But one single pixel has no real meaning

$\rightarrow$ **Sliding over fragments of the image**

The convolution defines a set of filters (aka kernels) to do just that

- Take "snapshots" of different areas of the image
- Process them, one at a time

Convolutional Neural Networks

Strides and filters

Stride

- The distance "traveled" when sliding
- Yet another parameter
- Never bigger than the size of the filter $\rightarrow$ overlapping areas

Sounds familiar? *n*-**grams!**

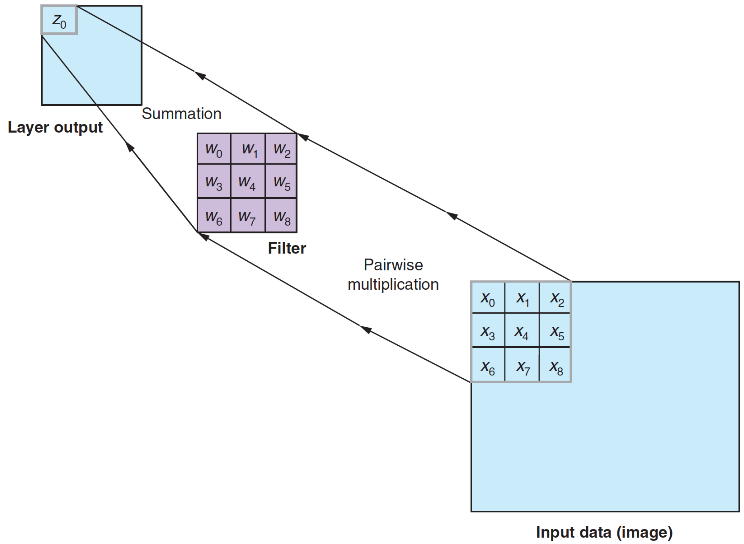
Filter

- $n \times m$ surfaces
- Typically $n = m = 3$ (often $n \neq m$)
- Includes a set of weights (fix for the whole image)
- Includes an activation function: usually ReLU

$$z = \max(sum(x * w), 0)$$

Convolutional Neural Networks

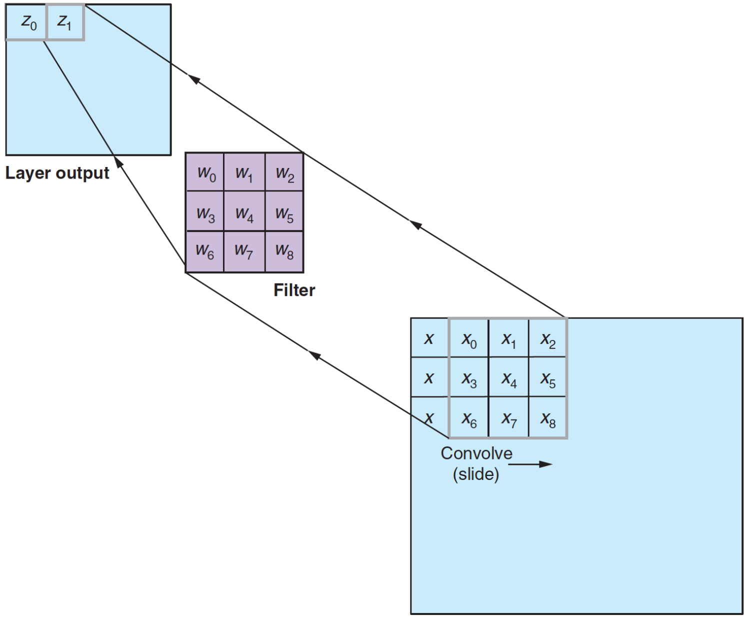
Convolutional step



(Lane et al., 2019, p. 225)

Convolutional Neural Networks

Convolution



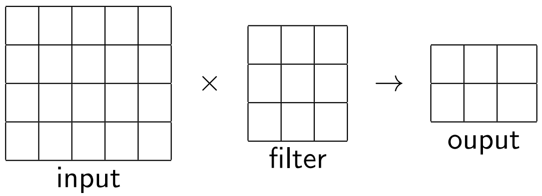
Convolutional Neural Networks

Producing multiple images

- k filters exist which carry out different operations
- Every filter will produce a new image, combination of source and filter

Convolutional Neural Networks

Padding



We are producing smaller images

"I don't care": Keras' argument `padding='valid'`
The edges of the image are undersampled

"I do care": Keras' padding argument `padding='same'`

In NLP **we care**

Convolutional Neural Networks

Pipeline

Input: an image, text

Output: a class, a real number

- Produce k new images through k filters
- Wire the filtered images to a feed-forward
- Proceed as usual

We can add multiple convolution layers

A full path of learning layers and abstractions

- Edges
- Shapes
- Colours
- **Concepts**

What is learned

- Good filters
- "Standard" weights

Convolutional Neural Networks

Keras premier

```
from keras.models import Sequential
from keras.layers import Conv1D

model = Sequential()

model.add(Conv1D(filters=16,
                 kernel_size=3,
                 padding='same',
                 activation='relu',
                 strides=1,
                 input_shape=(100, 300))
)
```


Next time

- CNNs for NLP

References

Lane, H., C. Howard, and H. Hapkem
2019. *Natural Language Processing in Action*. Shelter Island, NY: Manning Publication Co.

Le, Q. V. and T. Mikolov
2014. Distributed representations of sentences and documents.