Convolutional Neural Networks

Elisa Bassignana

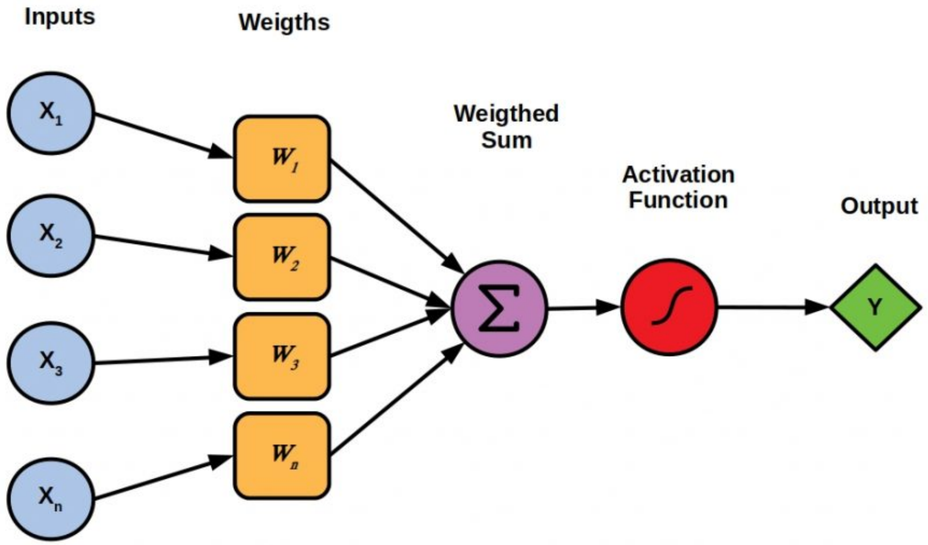
ITU - 22/02/2023

Plan of the day

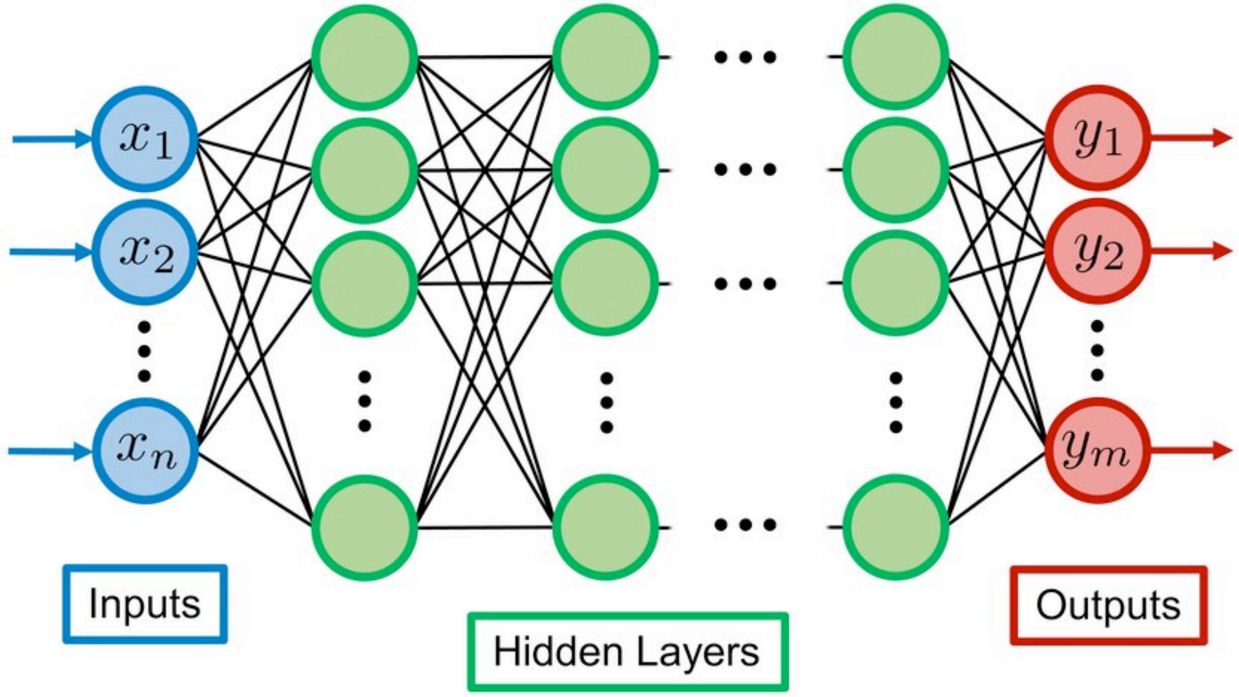
1. Recap of FFNN and their limitations
2. Introduction to Convolutional Neural Networks (CNNs)
3. CNNs for NLP
4. Case study with CNNs

Recap: Feed-Forward Neural Networks

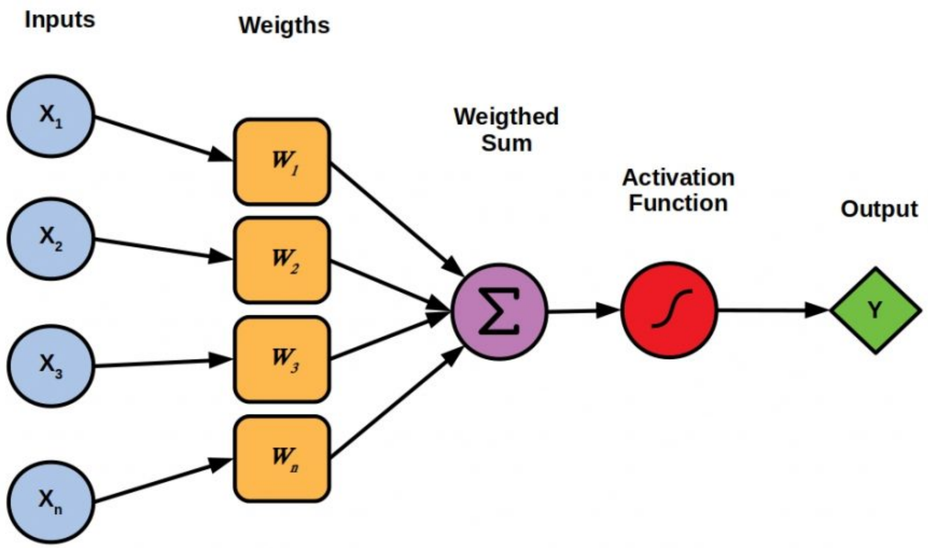
Neural unit

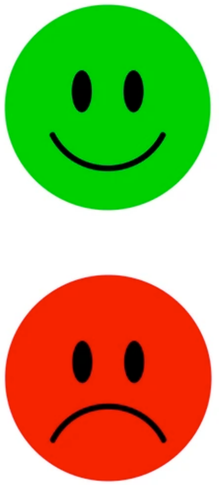


Recap: Feed-Forward Neural Networks



FFNN: Any problems?





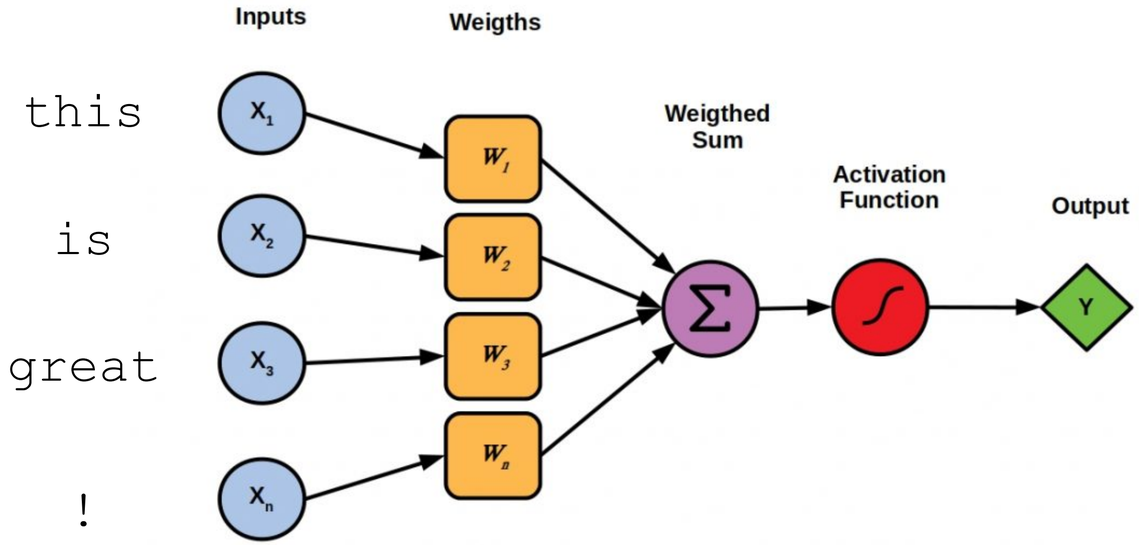
FFNN: Any problems?

this

is

great

!



FFNN: Any problems?

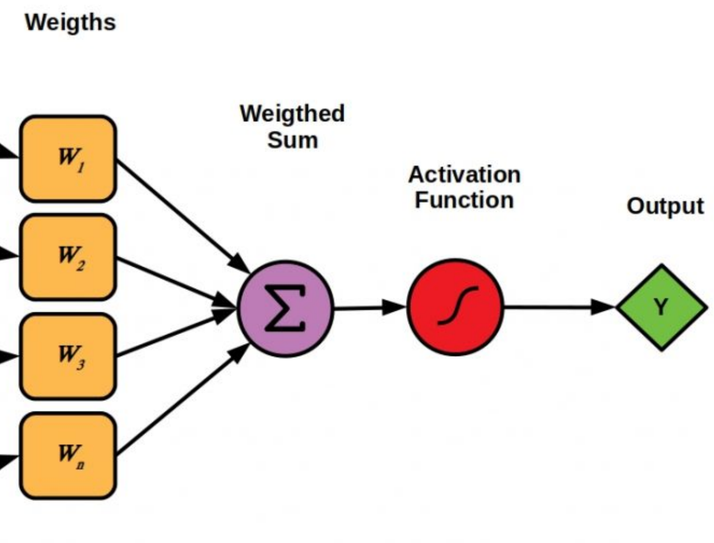
a

big

waste

of

time



FFNN: Any problems?

PROBLEM:
we need fixed input size!

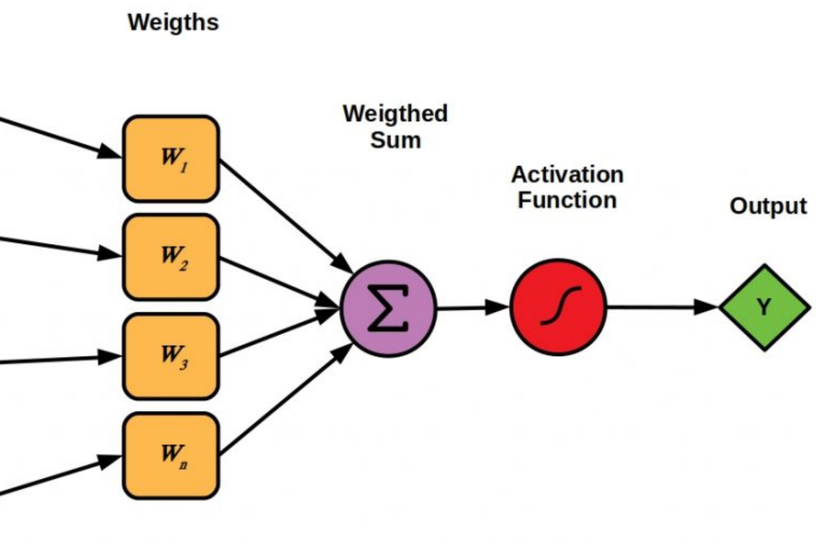
a

big

waste

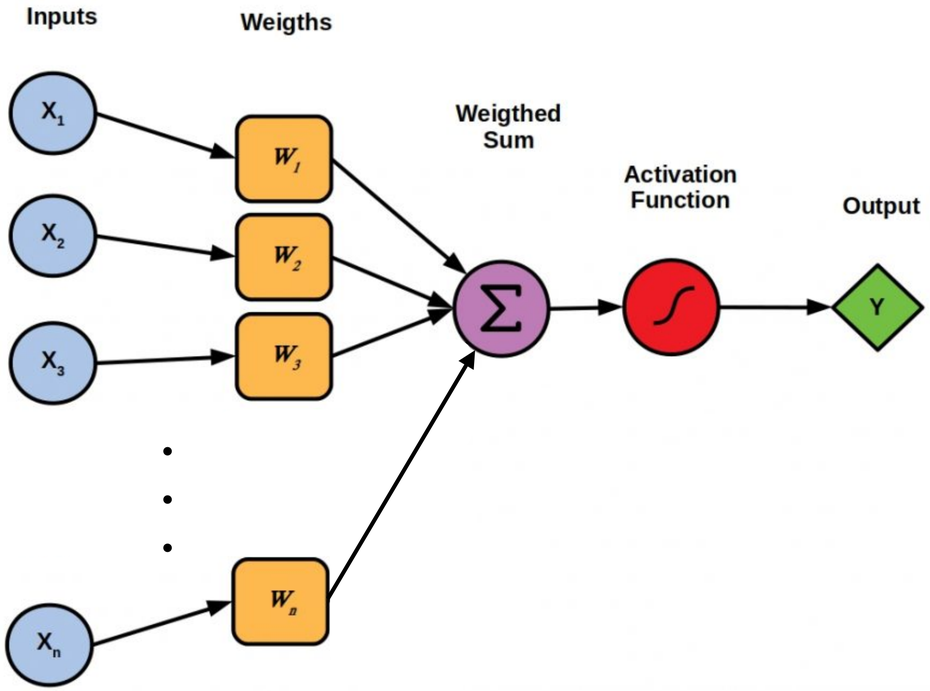
of

time



FFNN: What is still missing?

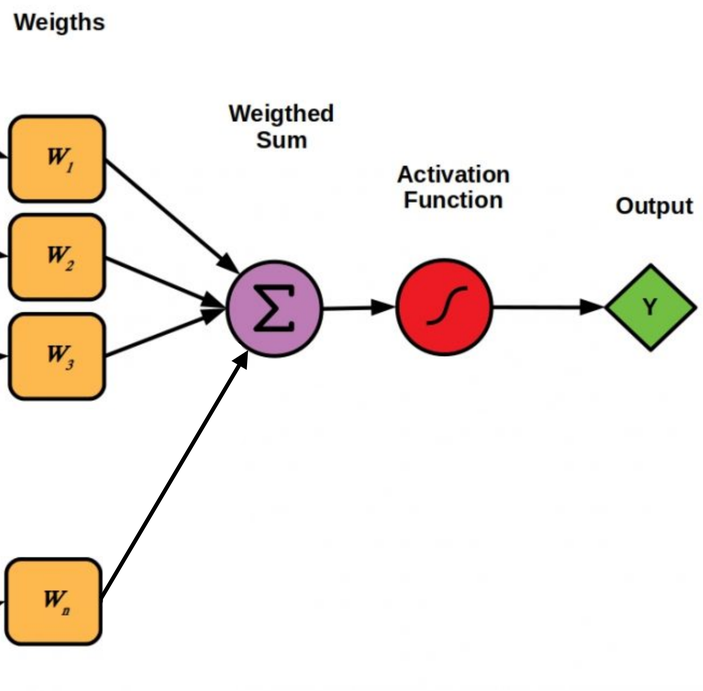
it
was
not
good
,
it
was
actually
quite
bad



FFNN: What is still missing?

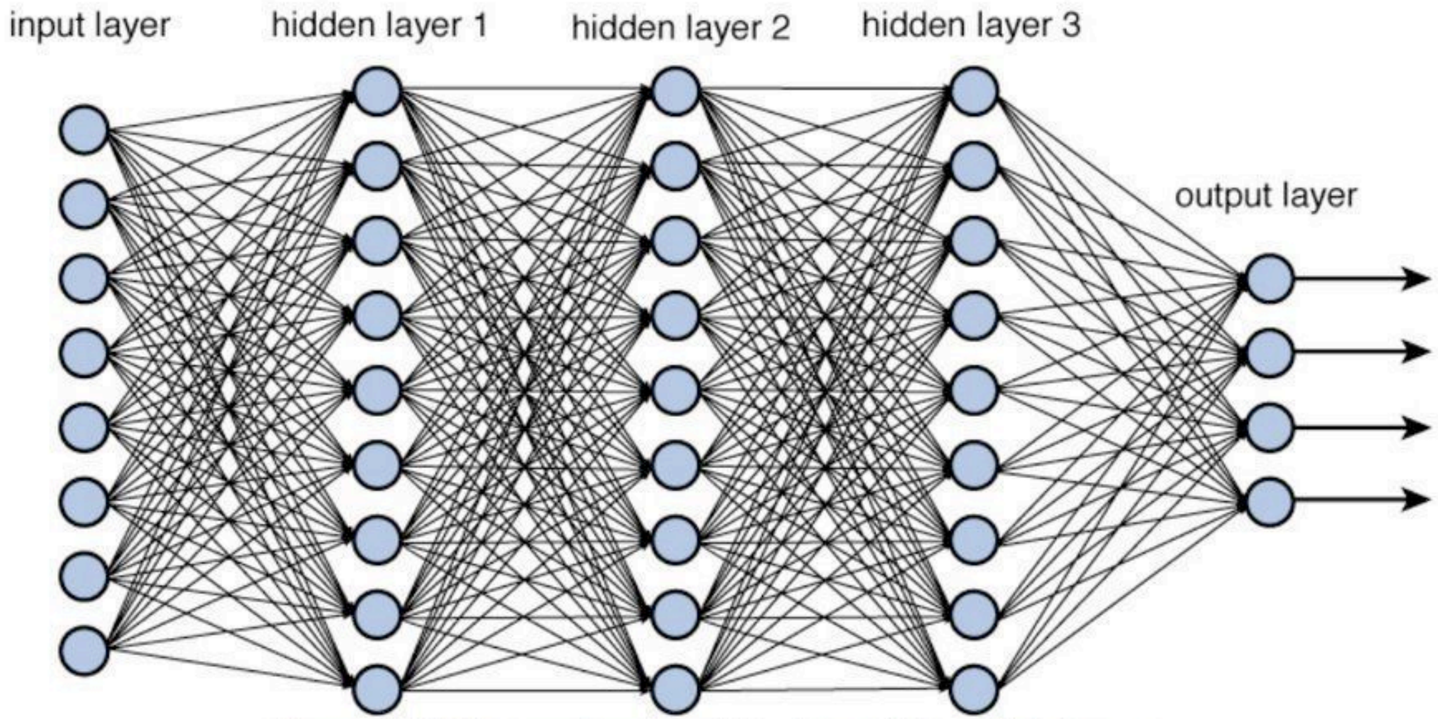
PROBLEM:
it does not consider the order!

it
was
not
good
,
it
was
actually
quite
bad



Moreover...

PROBLEM:
FFNN can became very complex:
many parameters to tune!



Any questions so far?

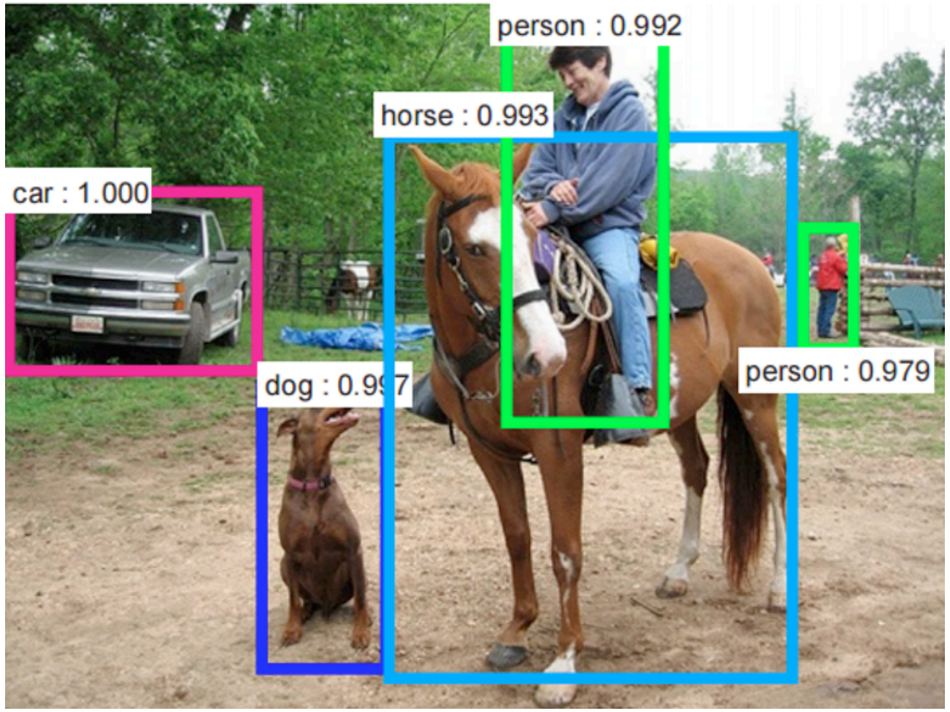
Convolutional Neural Networks (CNNs or convnets)

- From the computer vision field
- Work with 2-dimensional data (matrix/grid): images
- Can handle arbitrary-size input and reduce them down to a fixed size vector representation
- Core ideas:
  - *Local processing*
  - *Parameter sharing*

Intuition

- **Invariance** in the data

(we want to find an object regardless of its position in the image)



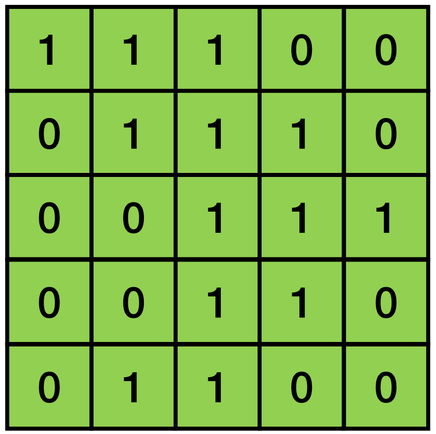
What are CNNs?

From a technical point of view there are 3 core concepts:

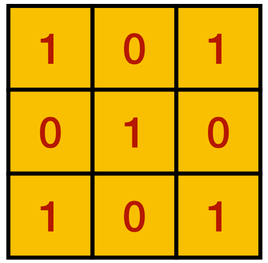
1. CNNs use **convolutions** over the input to compute the output
2. Each convolution applies different **filters**
3. Results of the convolutions are combined together via **pooling**

Filter (or kernel)

- Sliding window of learned weights



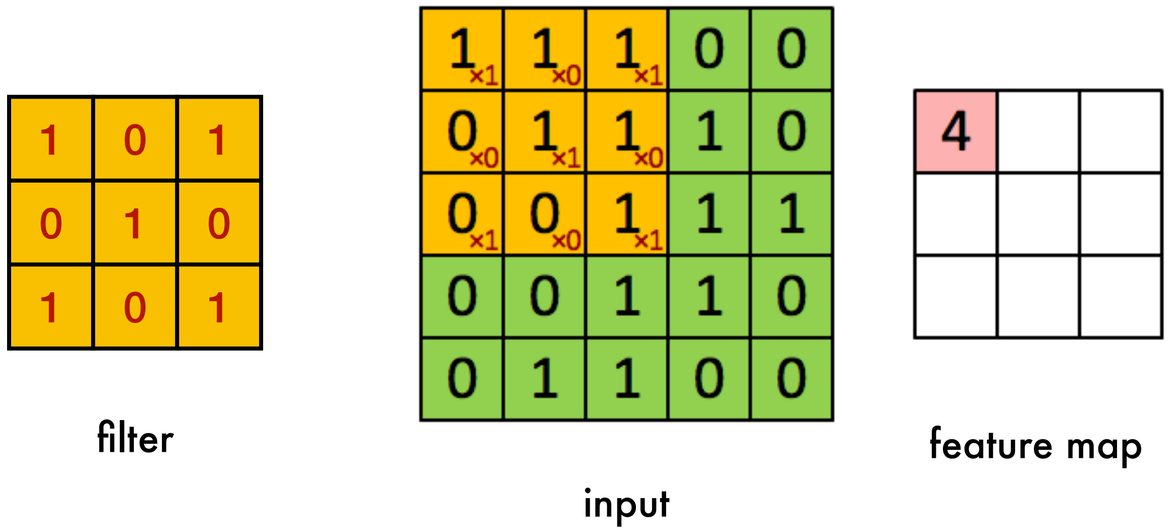
input



filter

Convolution

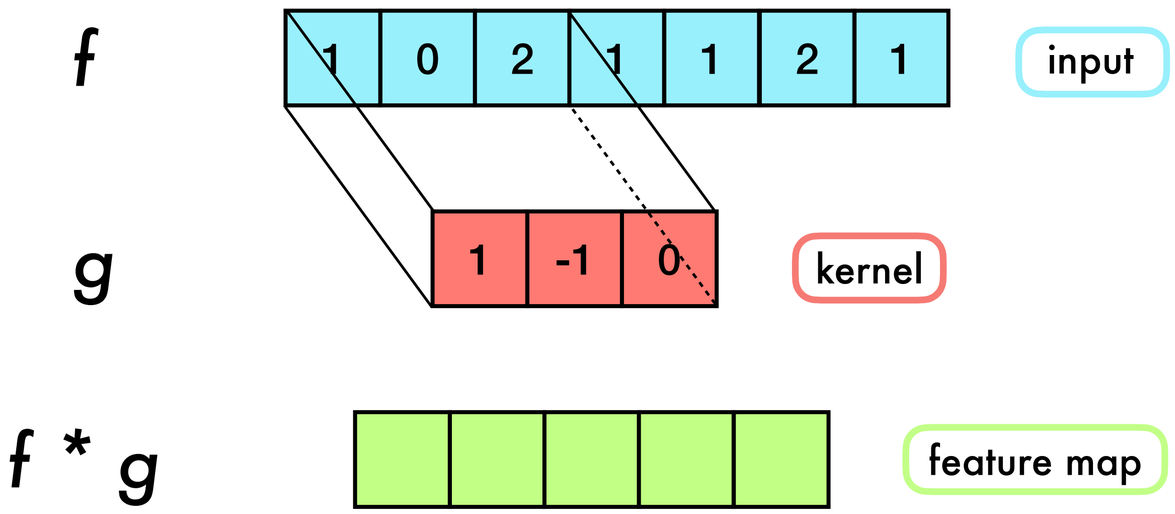
- Mathematical operation: **composition function**
- Sliding a filter over the input, generating an output (**feature map**)
- Parameters of the composition function are the same (shared)
- For each position of the filter, we multiply the overlapping values of the filter and of the image together and add up the results, to produce the output



Example

f is a 1-d input vector of length *n*

g is a 1-d filter (kernel) of length *m*



Example

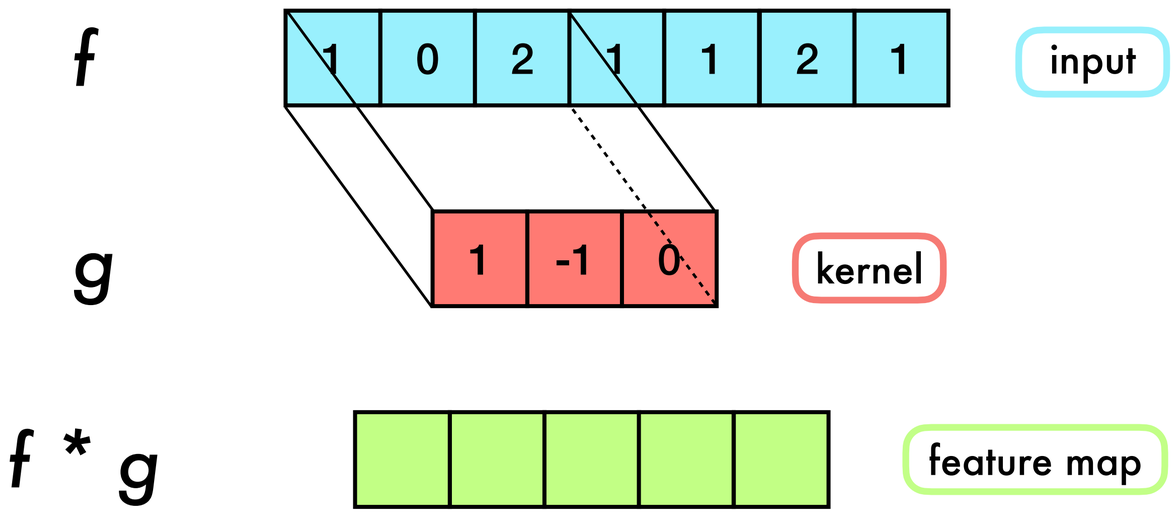
f is a 1-d input vector of length *n*

g is a 1-d filter (kernel) of length *m*

$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i + j - 1)$$

n = 7

m = 3



Example

f is a 1-d input vector of length *n*

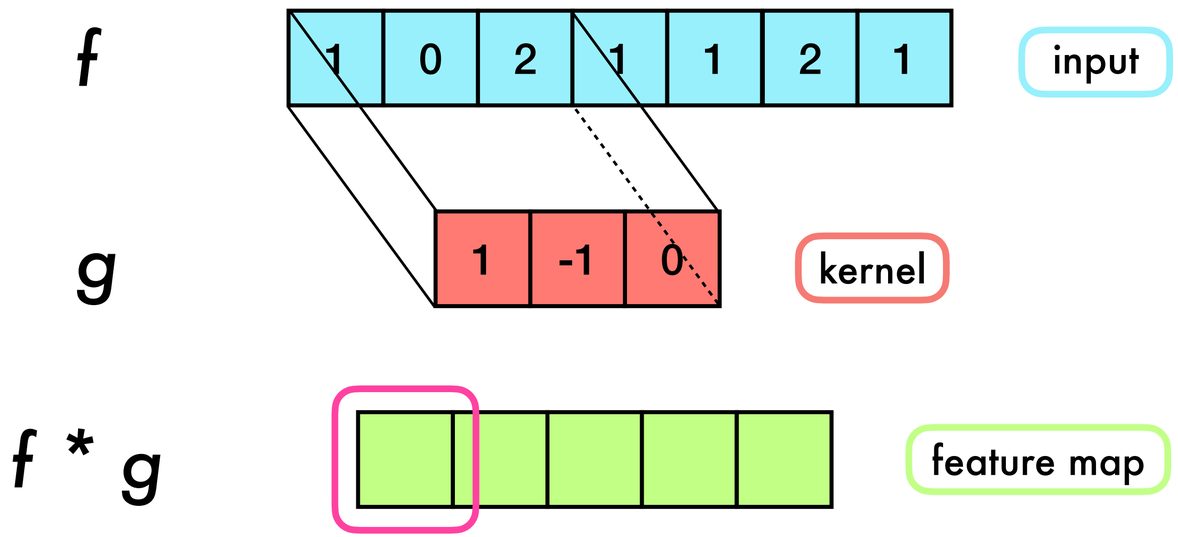
g is a 1-d filter (kernel) of length *m*

$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i + j - 1)$$

n = 7

m = 3

i = 1 2 3 4 5



Example

f is a 1-d input vector of length *n*

g is a 1-d filter (kernel) of length *m*

$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i + j - 1)$$

n = 7

m = 3

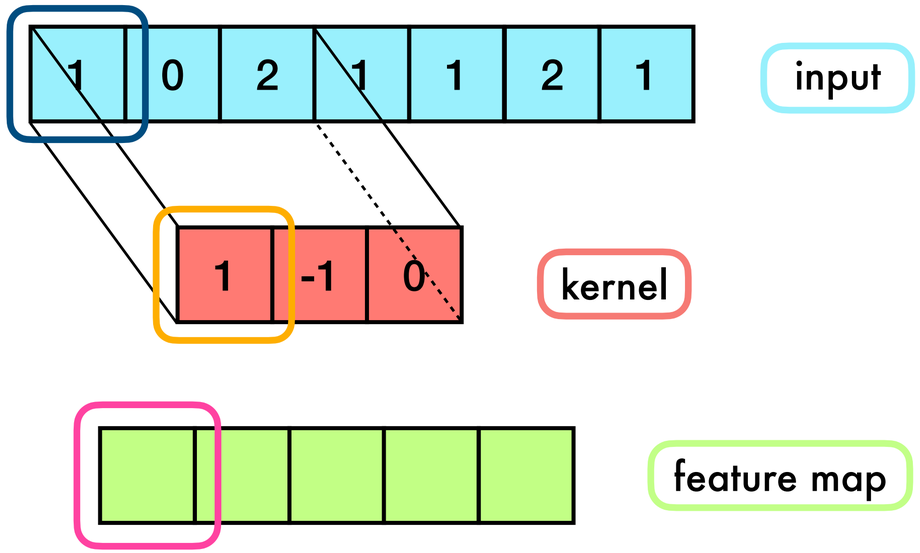
i = 1 2 3 4 5

j = 1 2 3

f

g

*f * g*



*1 * 1 = 1*

Example

f is a 1-d input vector of length *n*

g is a 1-d filter (kernel) of length *m*

$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i + j - 1)$$

n = 7

m = 3

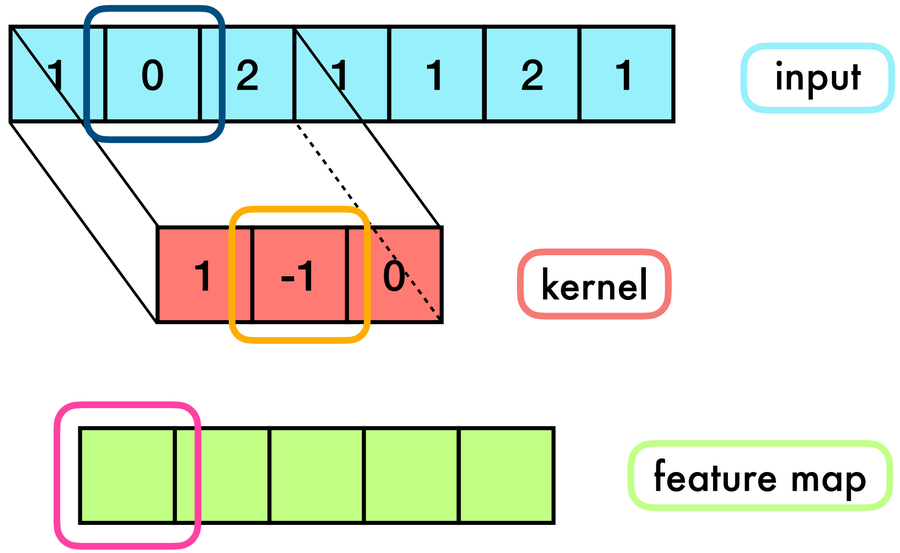
i = 1 2 3 4 5

j = 1 2 3

f

g

*f * g*



*1 * 1 = 1*

*-1 * 0 = 0*

Example

f is a 1-d input vector of length *n*

g is a 1-d filter (kernel) of length *m*

$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i + j - 1)$$

n = 7

m = 3

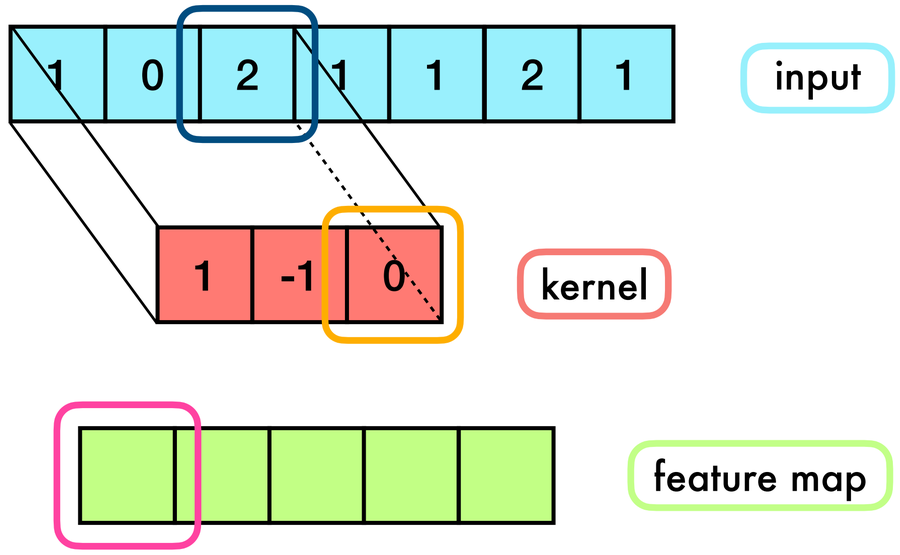
i = 1 2 3 4 5

j = 1 2 3

f

g

*f * g*



*1 * 1 = 1*

*-1 * 0 = 0*

*0 * 2 = 0*

Example

f is a 1-d input vector of length *n*

g is a 1-d filter (kernel) of length *m*

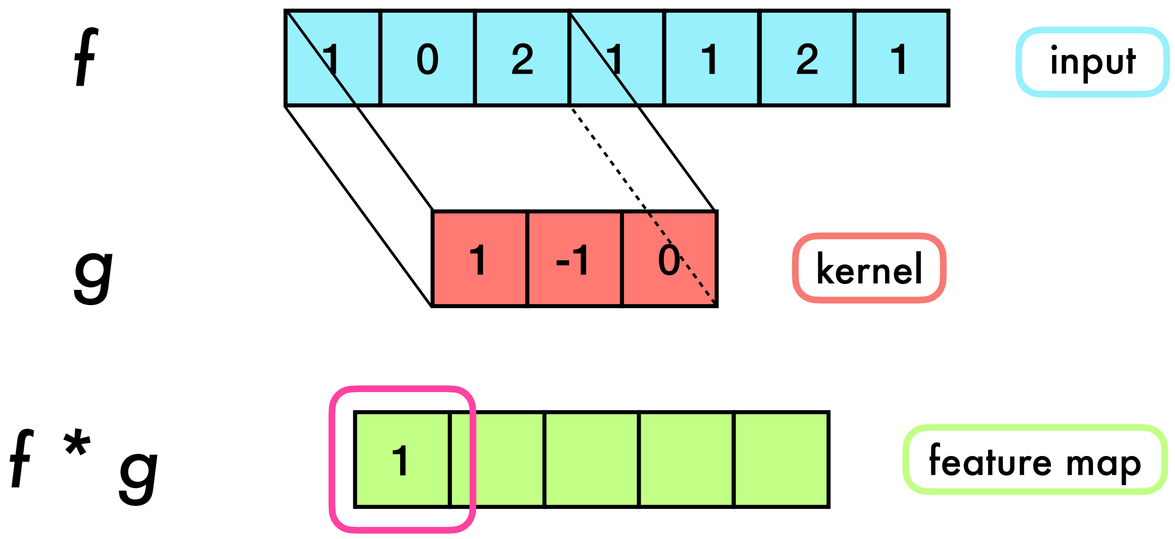
$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i + j - 1)$$

n = 7

m = 3

i = 1 2 3 4 5

j = 1 2 3



*1 * 1 = 1*

*-1 * 0 = 0*

*0 * 2 = 0*

1 + 0 + 0

Example

f is a 1-d input vector of length *n*

g is a 1-d filter (kernel) of length *m*

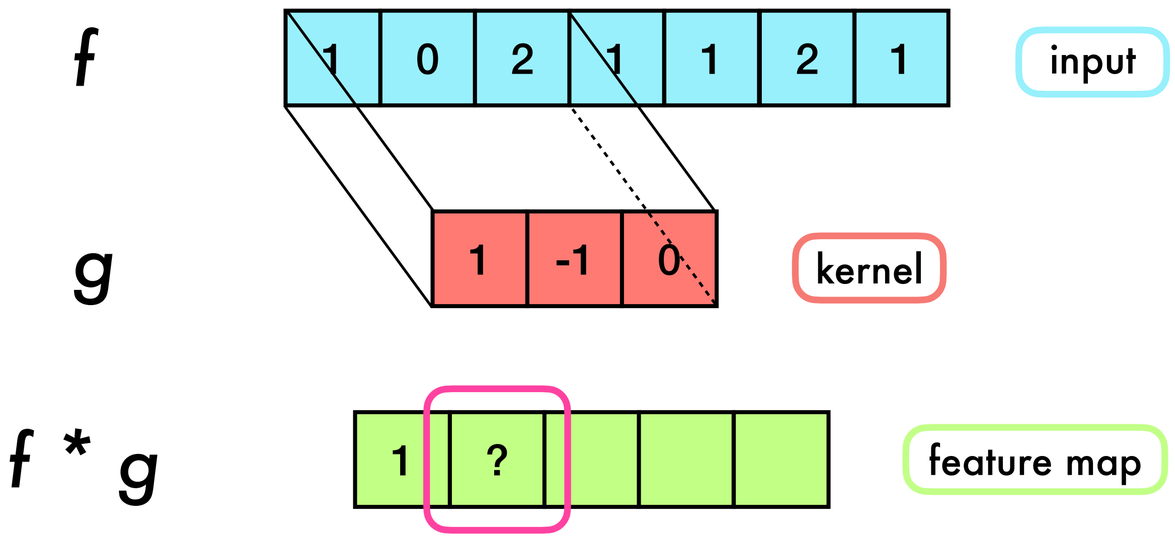
$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i + j - 1)$$

n = 7

m = 3

i = 1 2 3 4 5

j = 1 2 3



Example

f is a 1-d input vector of length *n*

g is a 1-d filter (kernel) of length *m*

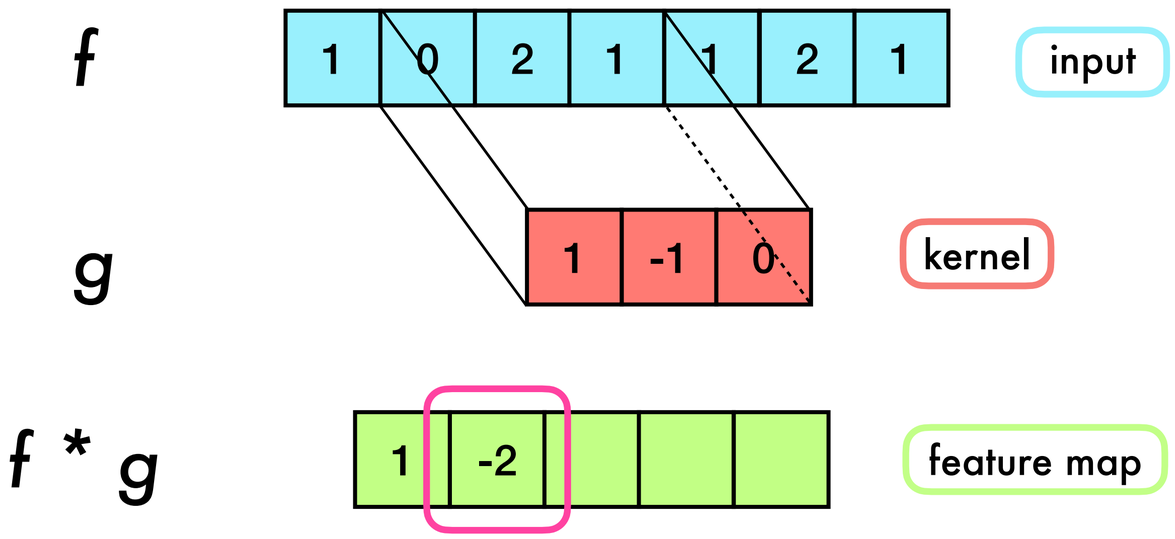
$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i + j - 1)$$

n = 7

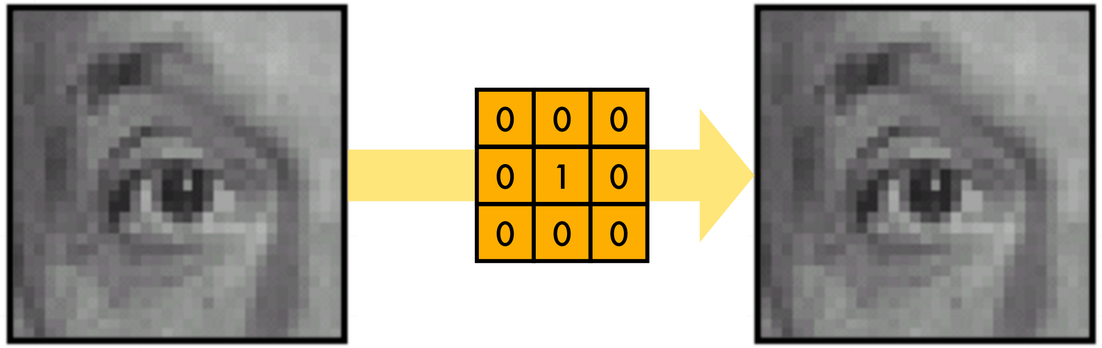
m = 3

i = 1 2 3 4 5

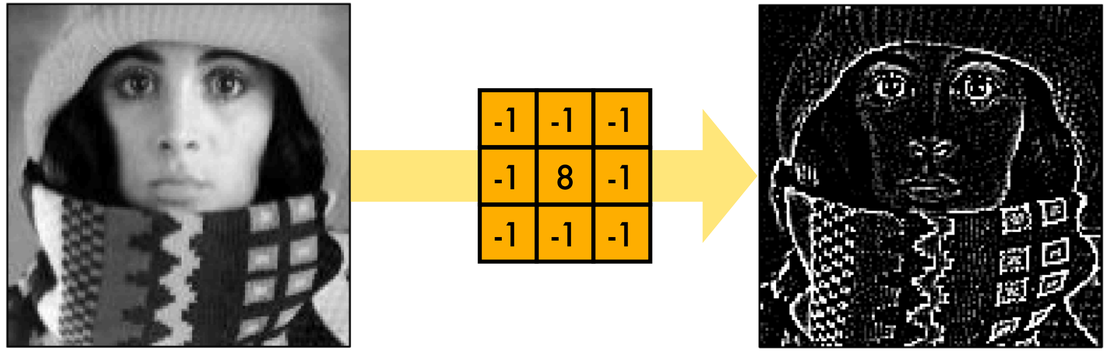
j = 1 2 3



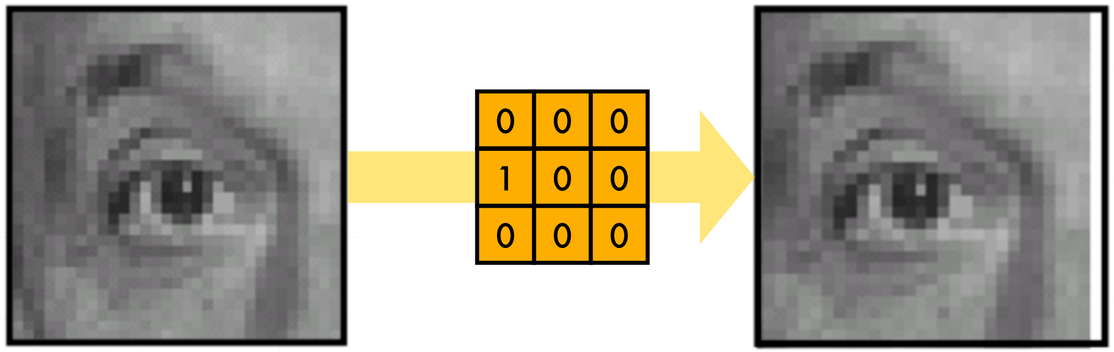
Example of defined kernels



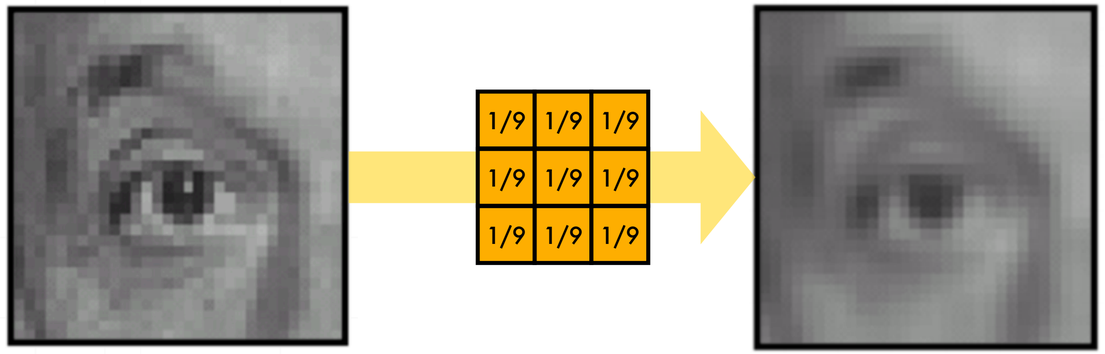
input original



input negative



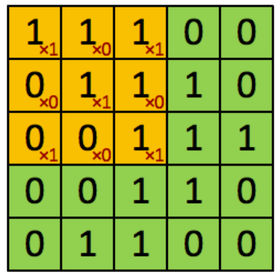
input shifted left by 1 pixel



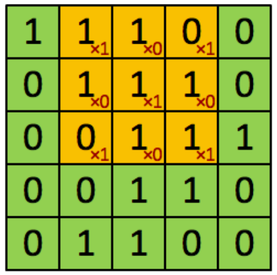
input blur (with a mean filter)

Two more concepts

- STRIDE: how much you shift your filter at each step

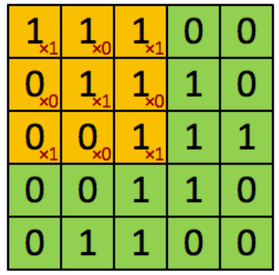


Step 1

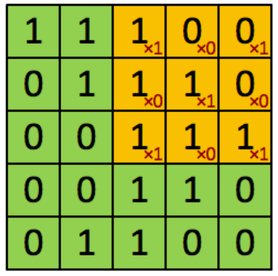


Step 2

stride = 1



Step 1

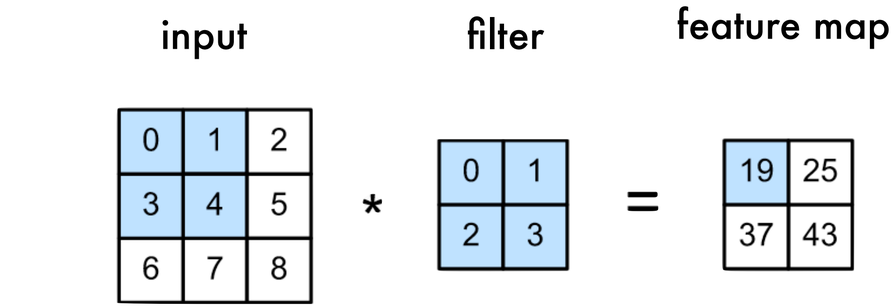


Step 2

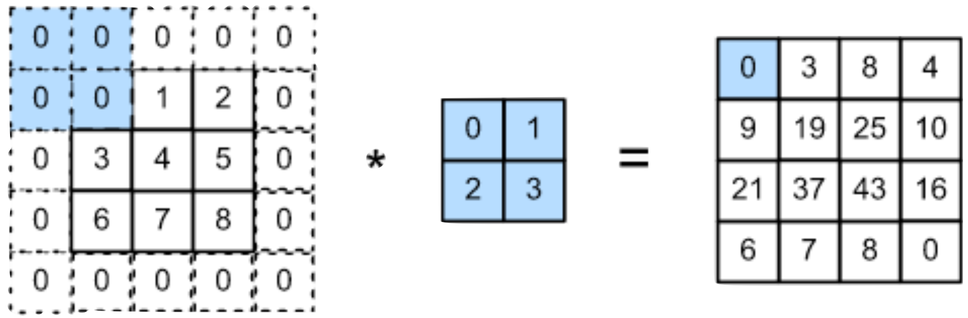
stride = 2

Two more concepts

- STRIDE: how much you shift your filter at each step
- PADDING: layers of zeros added to the input image



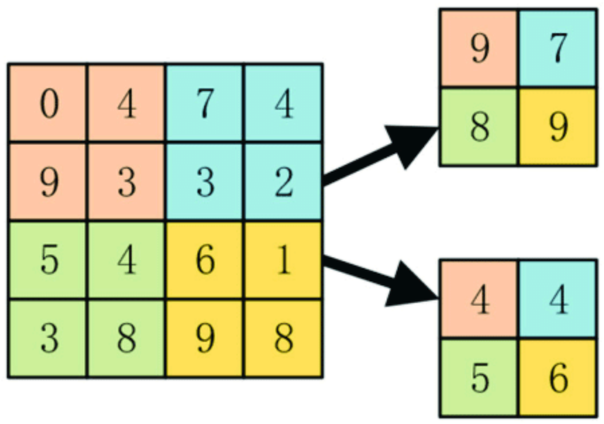
padding = 0



padding = 1

Pooling

- Reduces the spatial size of the feature maps
- Decrease the computational power required to process the data through dimensionality reduction by **extracting dominant features**



output of the convolution

MAX POOLING

- Returns the **maximum** value from the portion covered by the kernel
- *"Did you see this feature anywhere in the image?"*

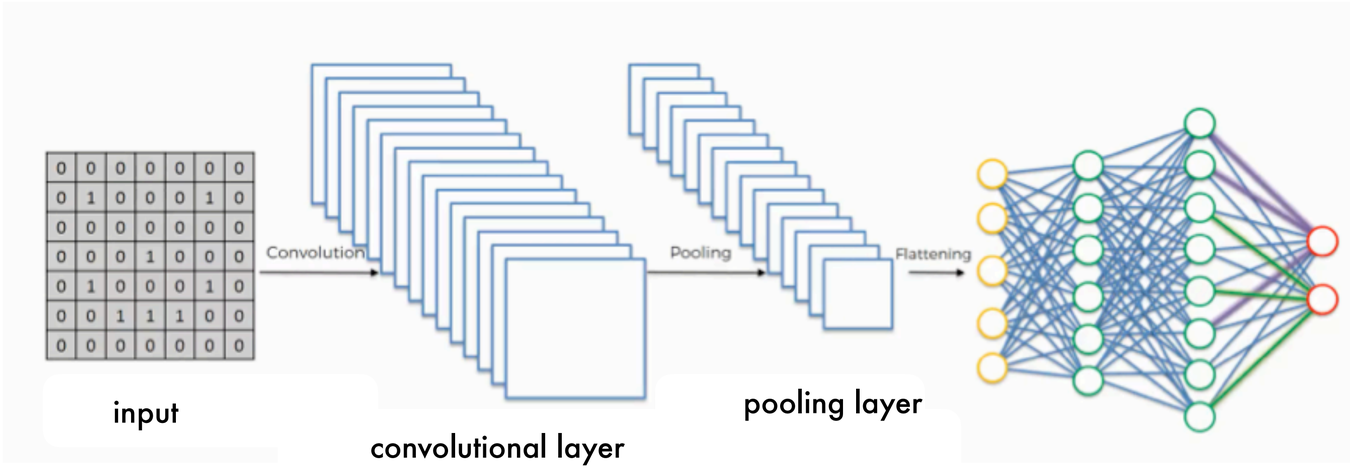
AVERAGE POOLING

- Returns the **average** of all values from the portion covered by the kernel
- *"How prevalent is this feature over the entire range?"*

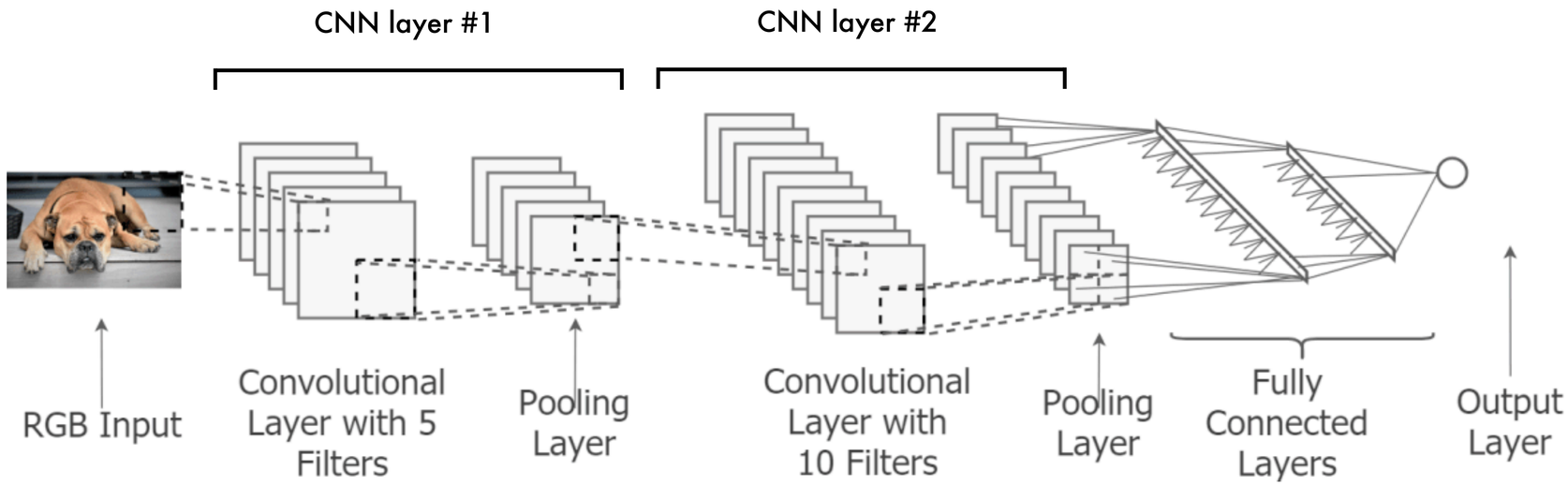
Pooling

- Pooling layers do not add parameters to the network
- Often, max-pooling performs better than average pooling
- A **Convolutional Layer** + a **Pooling Layer** form the *i*-th layer of a CNN

Overview of a CNN

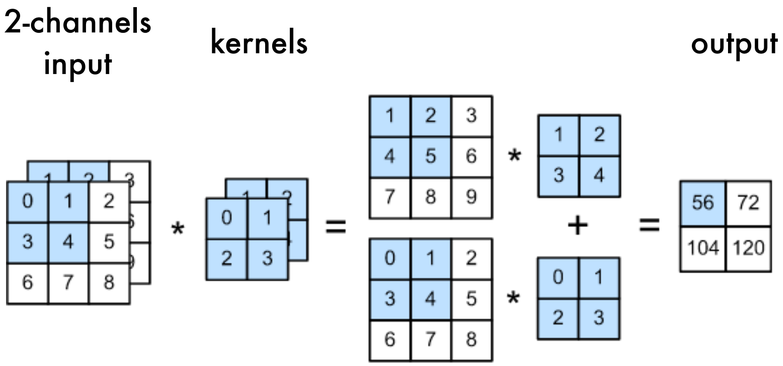


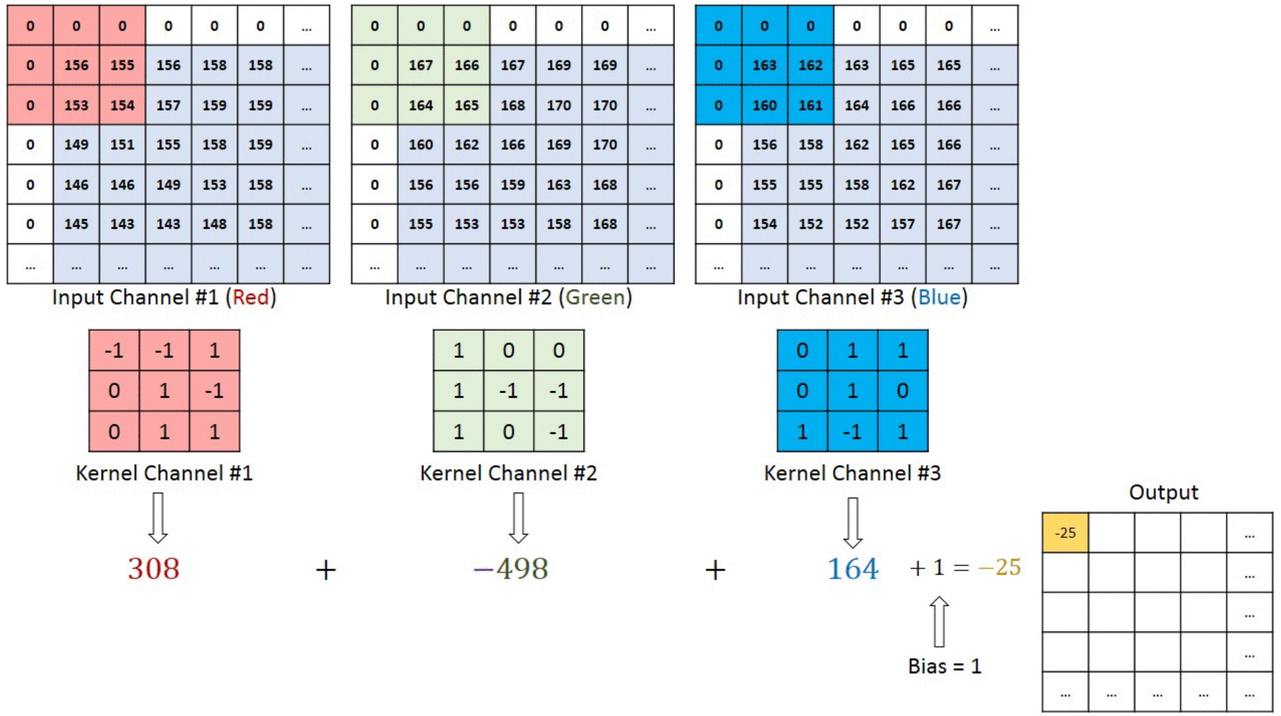
Multiple convolutional layers



- Convolutional layers can be applied both to the input data and to the output of other layers
- The stacking of convolutional layers allows a hierarchical decomposition of the input

Multiple channels





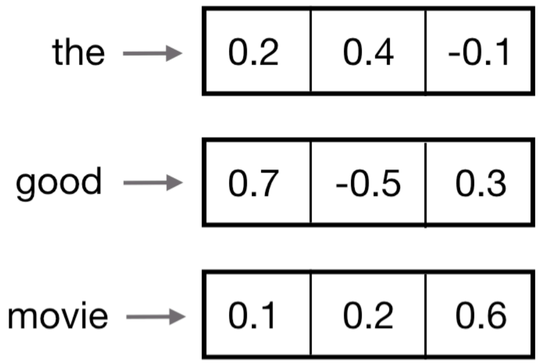
- Different "views" of the input

Questions about CNNs components/structure?

Okay...but we are working with text!

CNNs for text

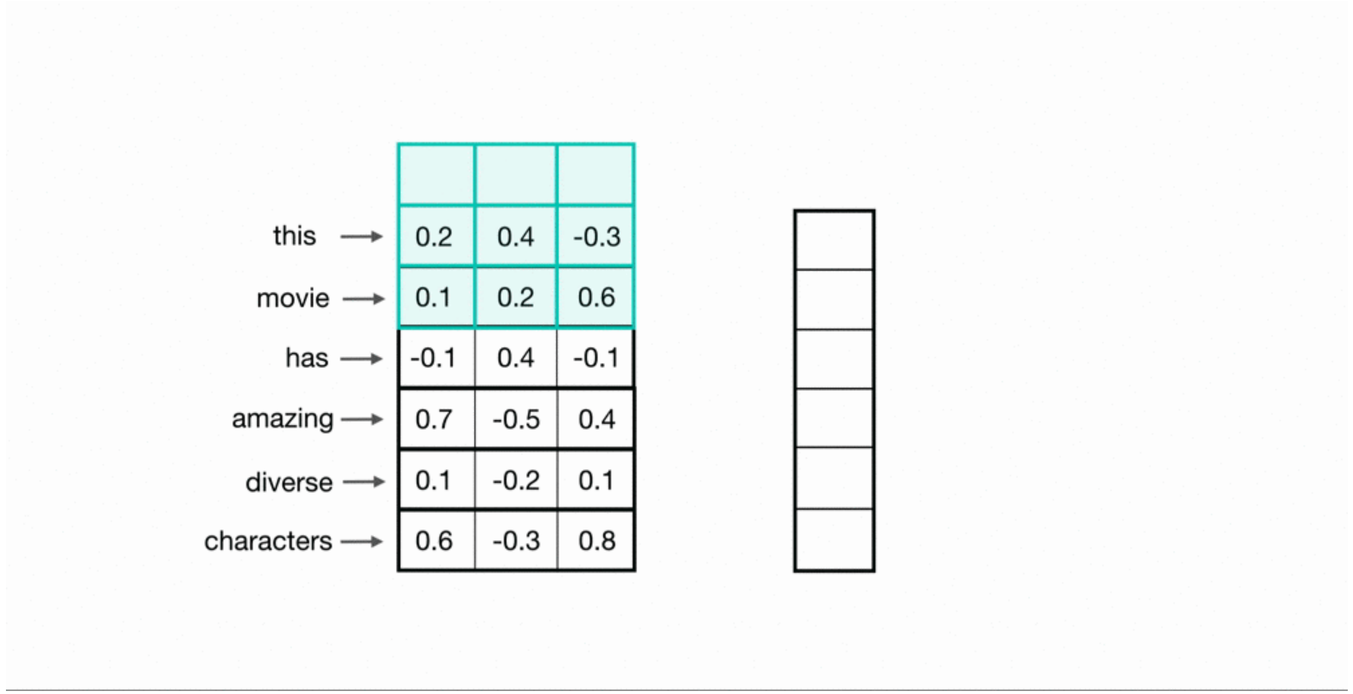
word2vec



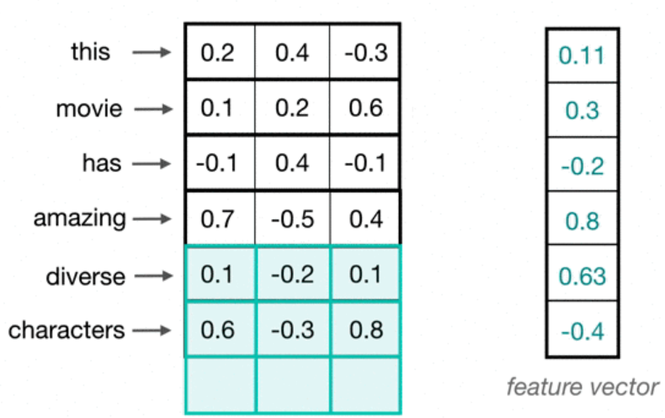
Images from: https://cezannec.github.io/CNN_Text_Classification/

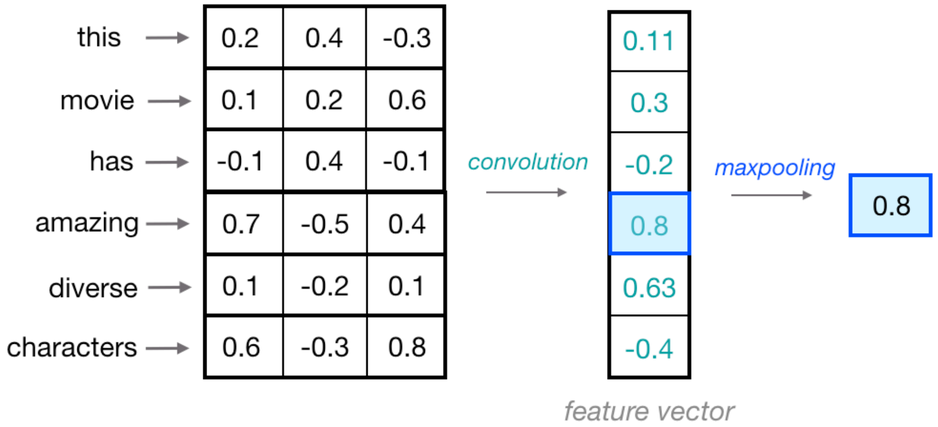
CNNs for text

CNNs for text



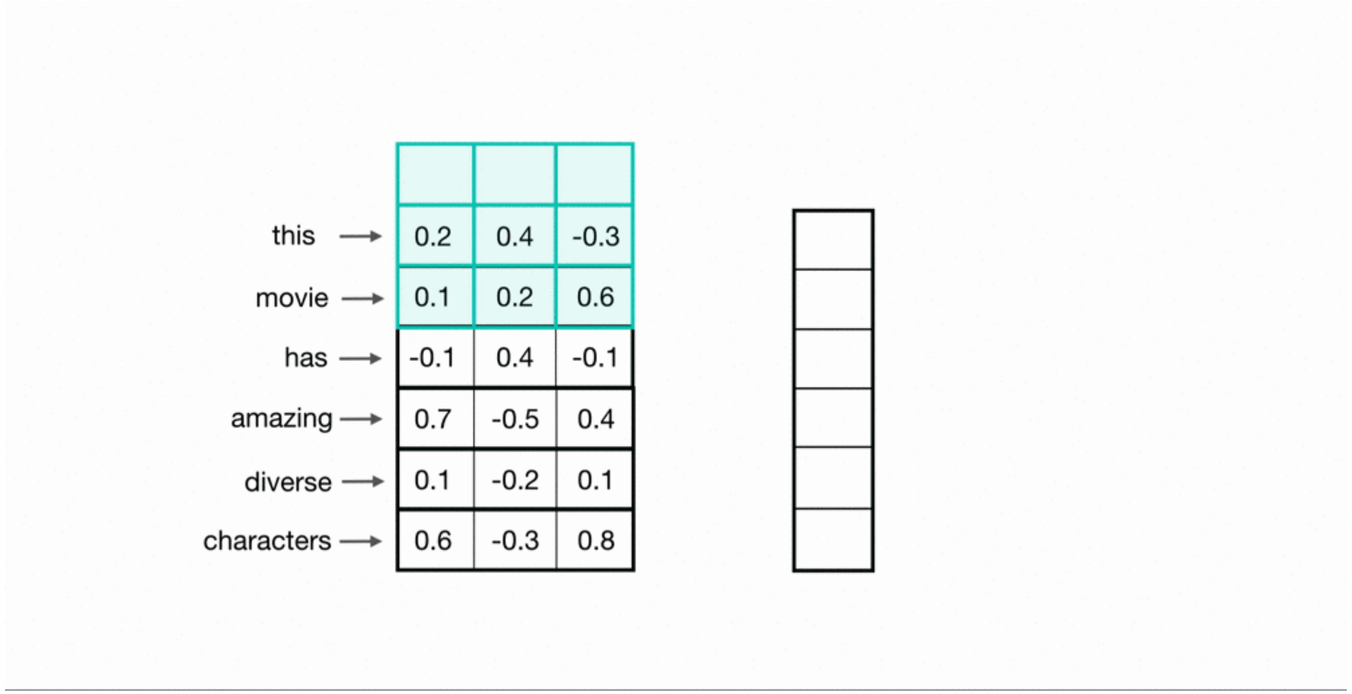
CNNs for text





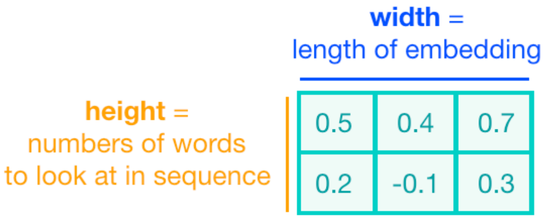
CNNs for text

- 1-direction convolution: analysis of the text over time
- kernel width = embedding length

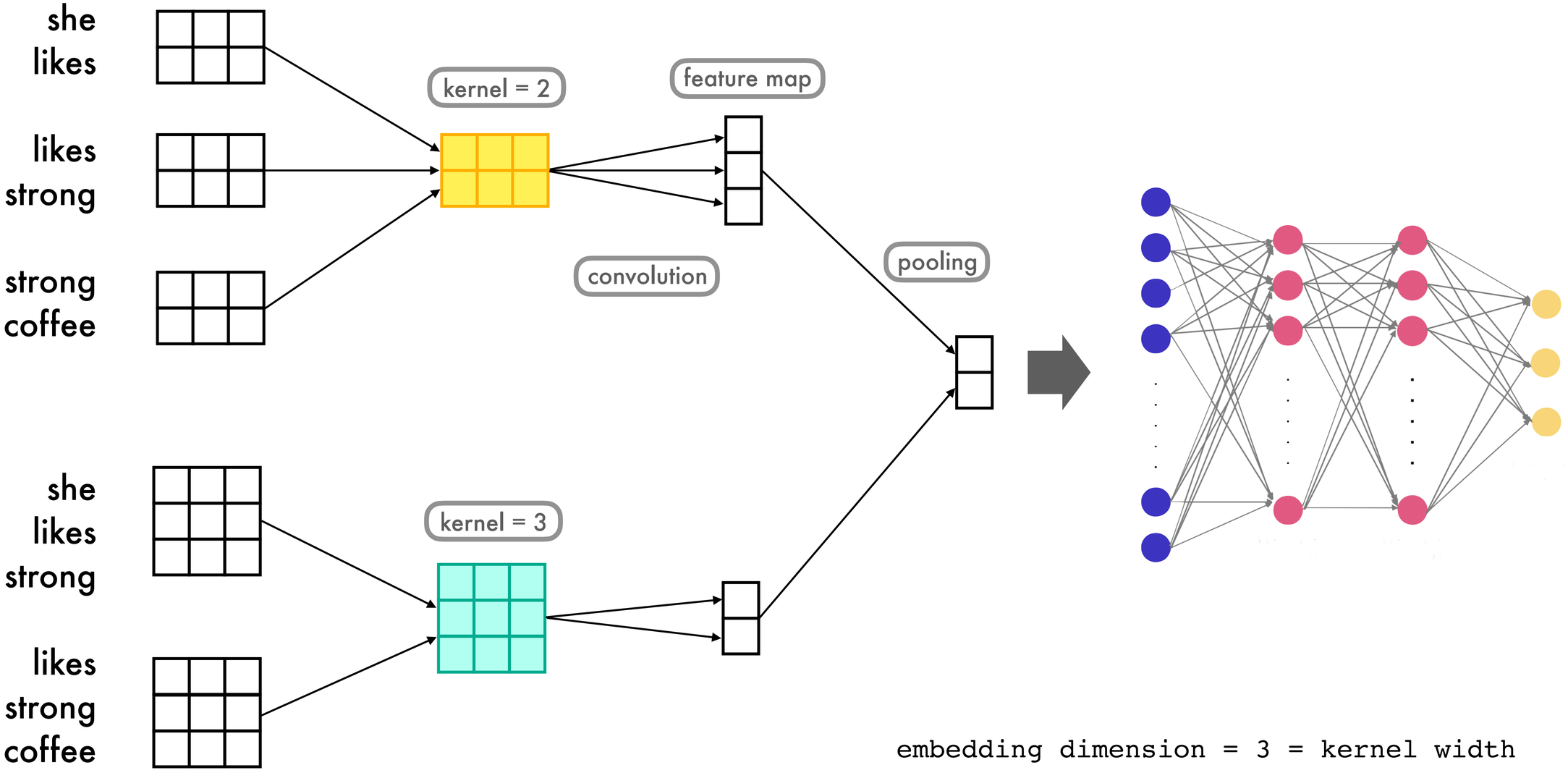


CNNs for text

- CNNs were introduced in NLP by Collobert et al. (2011) and later by Kim (2014) and Kalchbrenner et al. (2014)
- The intention is to let the network focus on the **most important "features" in the text**, regardless of their location
- The idea is to apply a non-linear (learned) function over each instantiation of a **k-word** sliding window over the text



Sample: *She likes strong coffee*




```
embedding dimension = 3 = kernel width

filter height = 2,  3
```

Word representations (embeddings)

Initialisation type:

- Traditional count-based methods
- word2vec: skip-gram, cbow
- Randomly initialised

Word representations (embeddings)

Initialisation type:

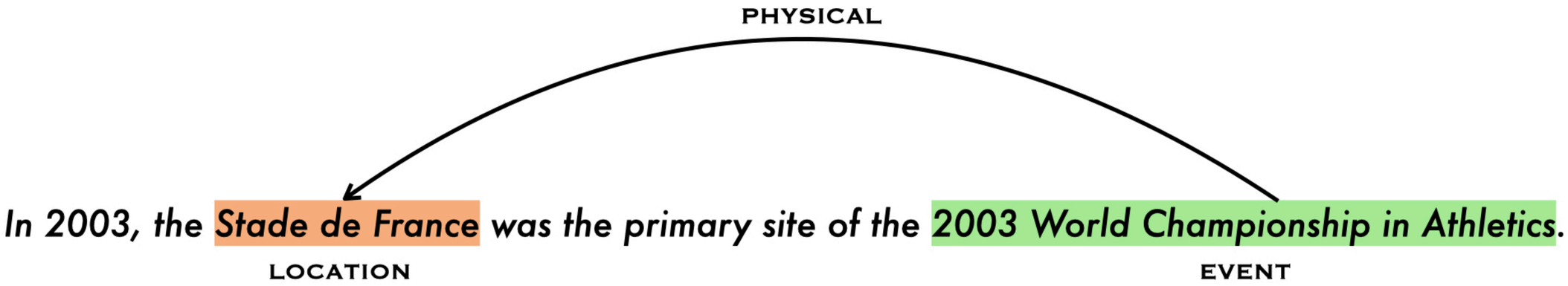
- Traditional count-based methods
- word2vec: skip-gram, cbow
- Randomly initialised

Embedding type:

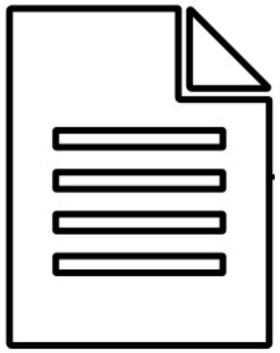
- Static
- Fine-tuned (non-static)

Questions about CNNs for NLP?

Relation Extraction

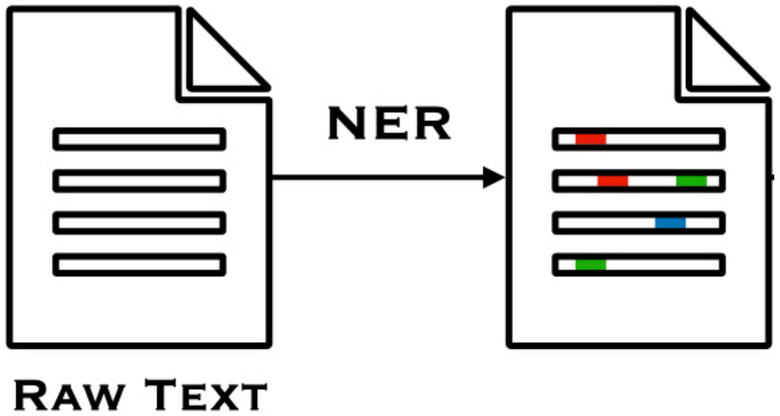


Relation Extraction



In 2003, the Stade de France was the primary site of the 2003 World Championship in Athletics.

Relation Extraction

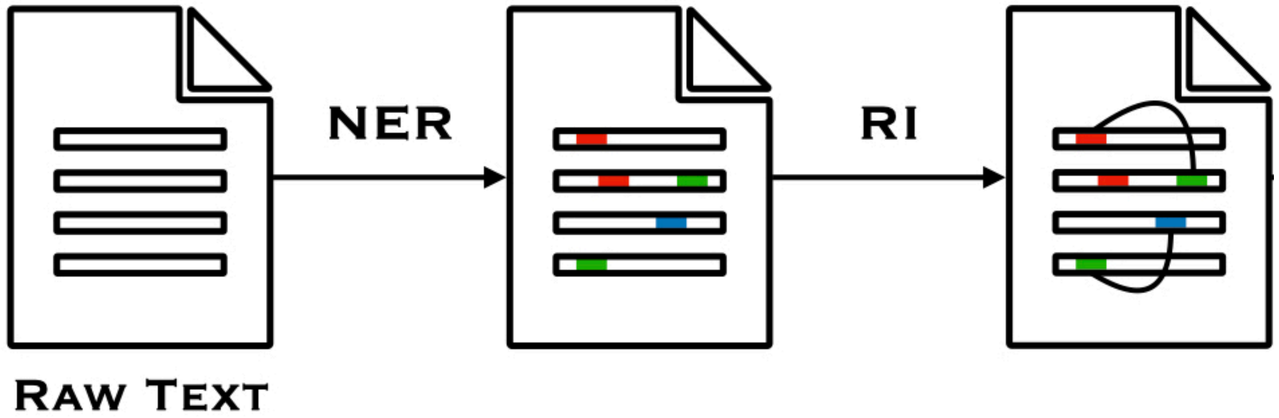


NAMED ENTITY RECOGNITION

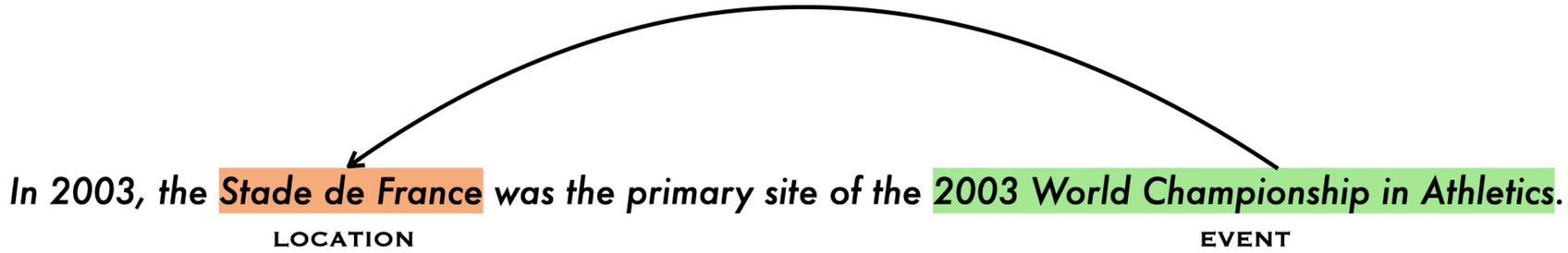
In 2003, the Stade de France was the primary site of the 2003 World Championship in Athletics.

LOCATION EVENT

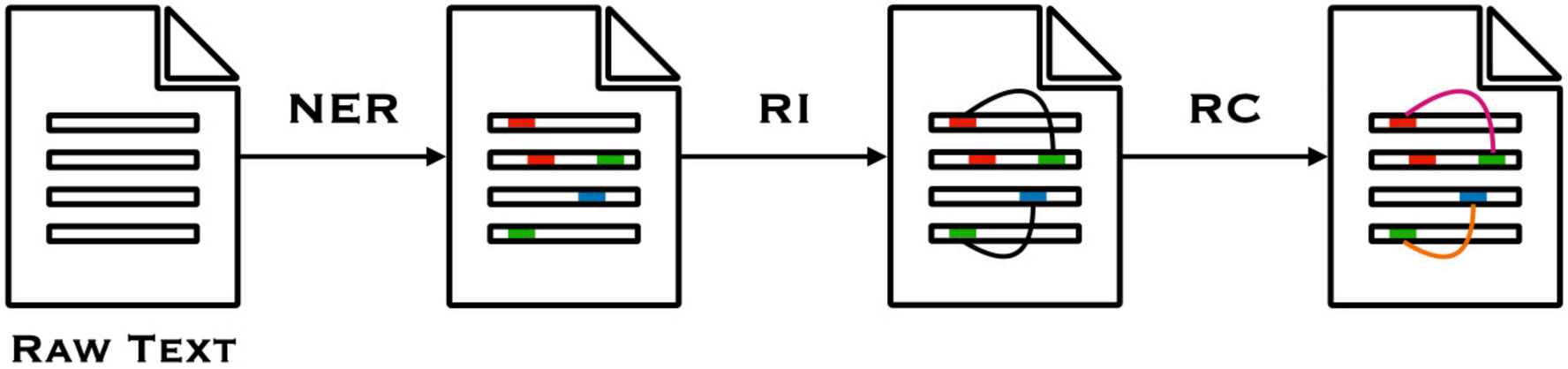
Relation Extraction



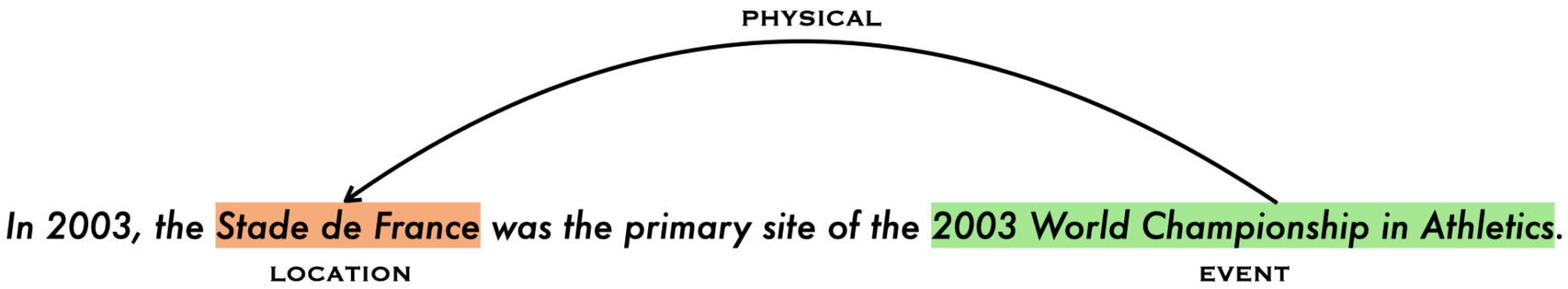
NAMED ENTITY RECOGNITION -> RELATION IDENTIFICATION



Relation Extraction



NAMED ENTITY RECOGNITION -> RELATION IDENTIFICATION -> RELATION CLASSIFICATION



How to evaluate Relation Extraction?

A predicted relation is considered correct if …

- *strict evaluation*:
- *boundaries evaluation*:
- *relaxed evaluation*:

… and the predicted relation type is correct

How to evaluate Relation Extraction?

A predicted relation is considered correct if …

- *strict evaluation*: the boundaries of the two entity spans, as well as the entity types are correct
- *boundaries evaluation*:
- *relaxed evaluation*:

… and the predicted relation type is correct

How to evaluate Relation Extraction?

A predicted relation is considered correct if …

- *strict evaluation*: the boundaries of the two entity spans, as well as the entity types are correct
- *boundaries evaluation*: the boundaries of the two spans are correct
- *relaxed evaluation*:

… and the predicted relation type is correct

How to evaluate Relation Extraction?

A predicted relation is considered correct if …

- *strict evaluation*: the boundaries of the two entity spans, as well as the entity types are correct
- *boundaries evaluation*: the boundaries of the two spans are correct
- *relaxed evaluation*: at least one token per entity span is correctly typed

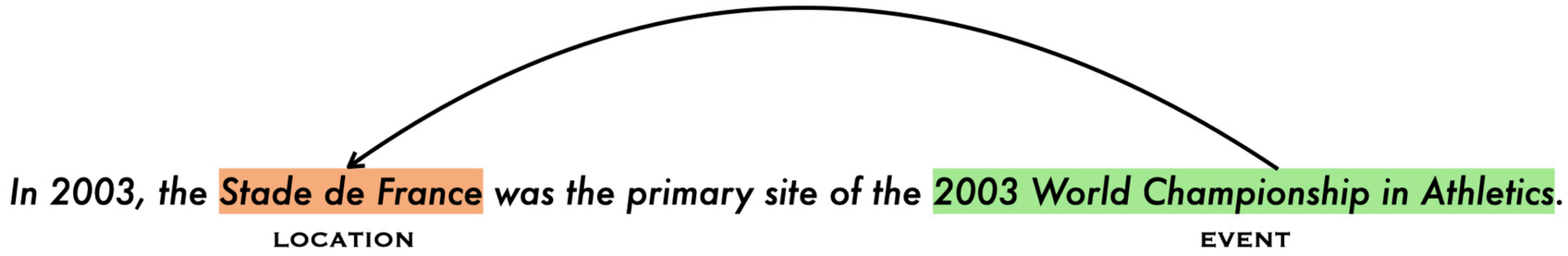
… and the predicted relation type is correct

Metrics

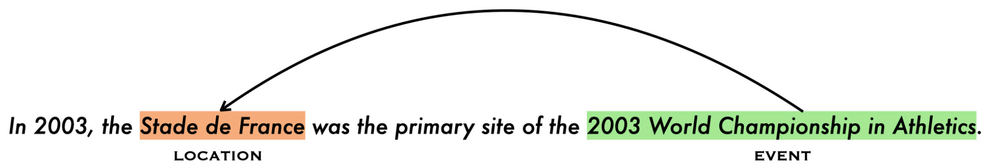
- *macro-F1*: how much the model learn of each class
- *micro-F1*: how the model perform in general on the task

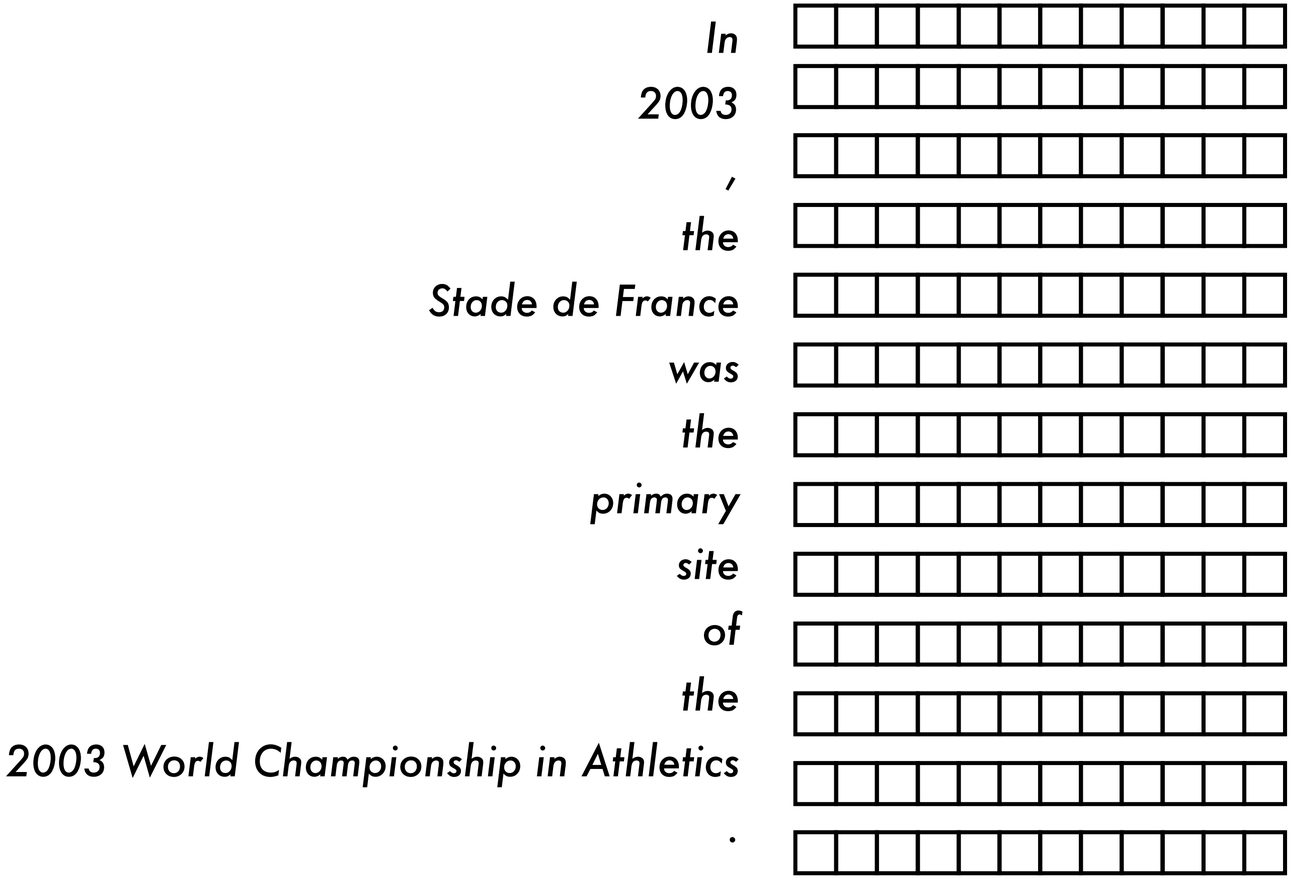
Relation Classification

- Input information:
  - Pair of ordered entities
- Output:
  - Relation type from a given label set {PHYSICAL, TEMPORAL, PART-OF, TYPE-OF}

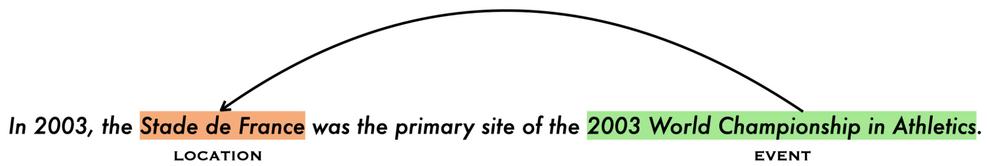


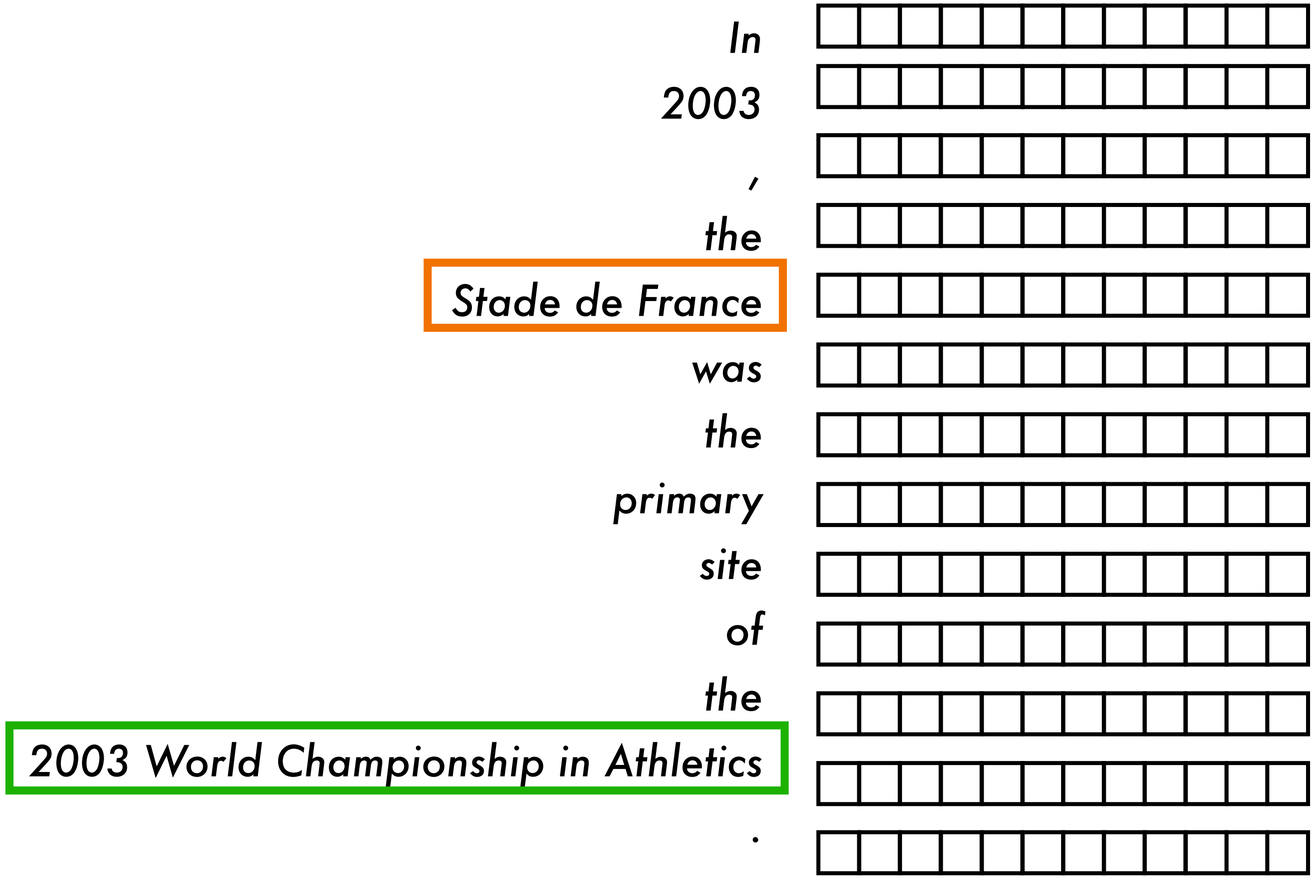
Relation Classification



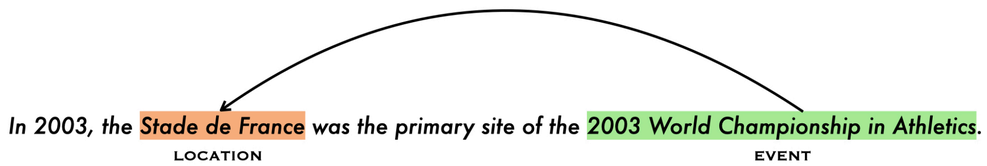


Relation Classification





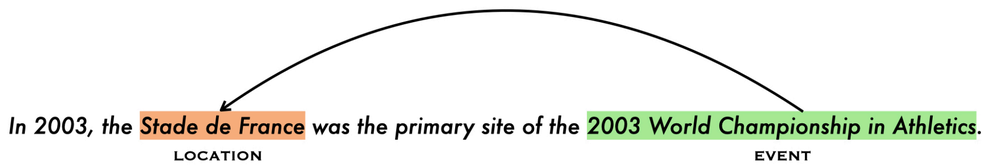
Relation Classification



In	0	0
2003	0	0
,	0	0
the	0	0
Stade de France	0	1
was	0	0
the	0	0
primary	0	0
site	0	0
of	0	0
the	0	0
2003 World Championship in Athletics	1	0
.	0	0

Information about which are the entities

Relation Classification

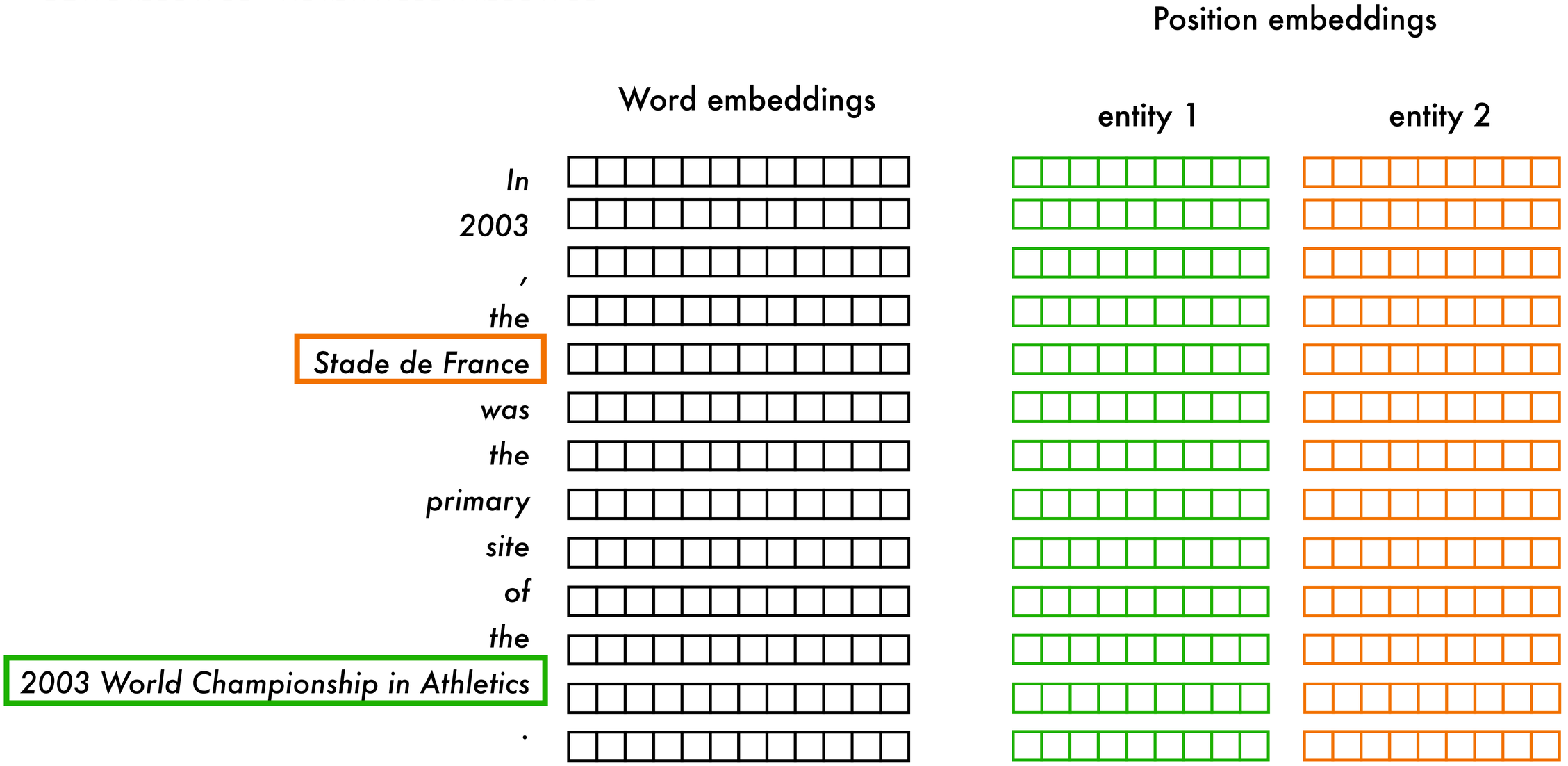


In	-11	-4
2003	-10	-3
,	-9	-2
the	-8	-1
Stade de France	-7	**0**
was	-6	1
the	-5	2
primary	-4	3
site	-3	4
of	-2	5
the	-1	6
2003 World Championship in Athletics	**0**	7
.	1	8

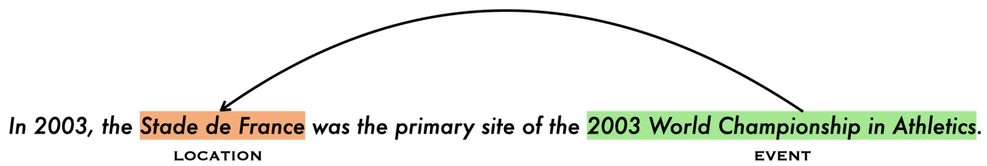
More fine-grained:

Relative distance of each token from the 2 entities!

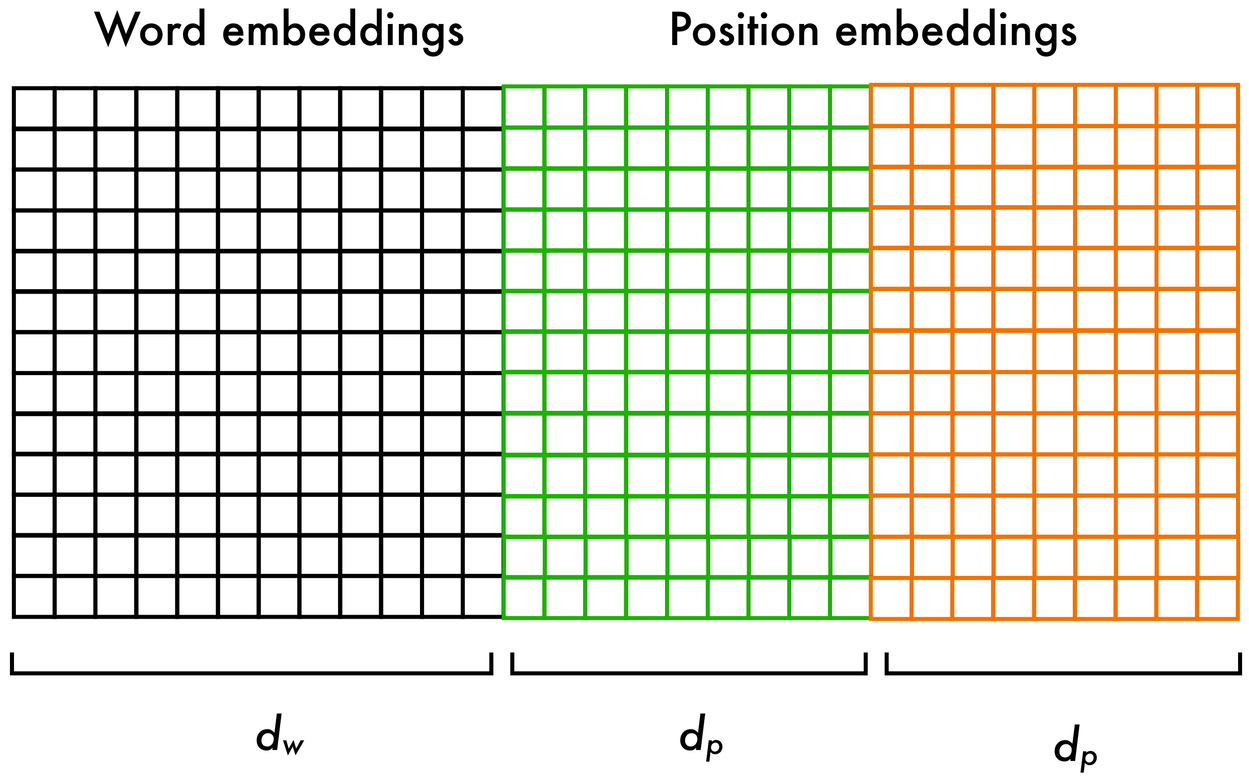
Relation Classification



Relation Classification

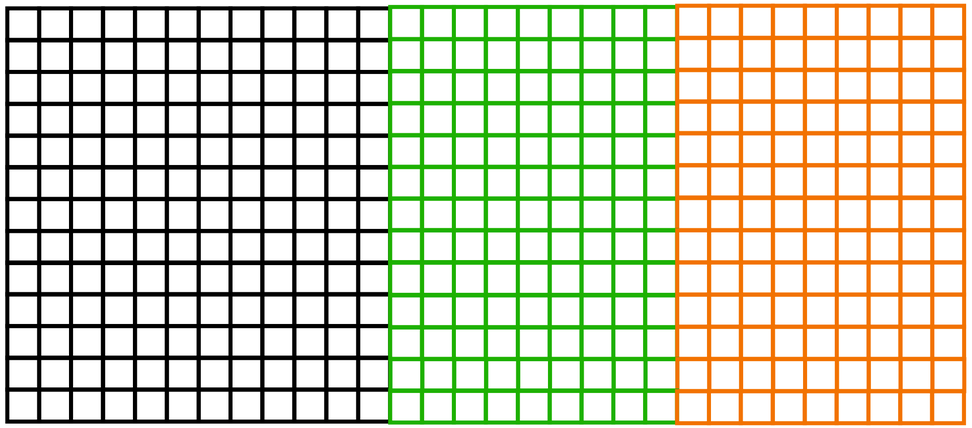


In
2003
,
the
Stade de France
was
the
primary
site
of
the
2003 World Championship in Athletics
.



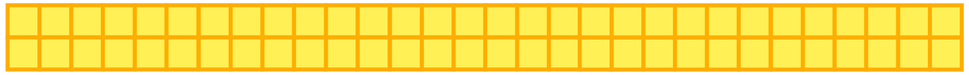
Relation Classification

input

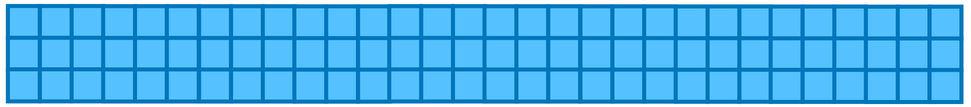


3 filters

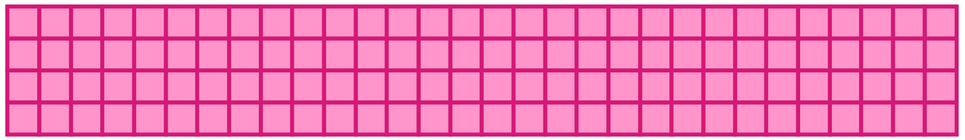
num of words x embedding length



2 x $(d_w + d_p + d_p)$

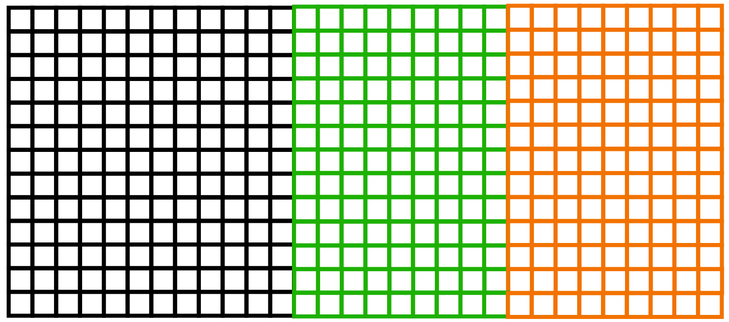


3 x $(d_w + d_p + d_p)$



4 x $(d_w + d_p + d_p)$

Relation Classification model example



input

convolution (3 kernels)

max pooling

FFNN (1 layer)

PHYSICAL
TEMPORAL
PART-OF
TYPE-OF

output

PHYSICAL

In 2003, the Stade de France was the primary site of the 2003 World Championship in Athletics.

LOCATION EVENT

Questions?

Sum up of the lecture

1. Recap of FFNN and their limitations
2. Introduction to Convolutional Neural Networks (CNNs)
3. CNNs for NLP
4. Case study with CNNs

Elisa Bassignana: elba@itu.dk

References

- Yoav Goldberg (Primer), 2015. A Primer on Neural Networks for Natural Language Processing. http://u.cs.biu.ac.il/~yogo/nnlp.pdf
- Kim 2014. Convolutional Neural Networks for Sentence Classification. https://www.aclweb.org/anthology/D14-1181/
- Cezanne Camacho blogpost. https://cezannec.github.io/CNN_Text_Classification/