CS772: Deep Learning for Natural Language Processing

Convolutional Neural Network

Pushpak Bhattacharyya

Computer Science and Engineering Department

IIT Bombay

Week 10 of 7th Mar, 2022

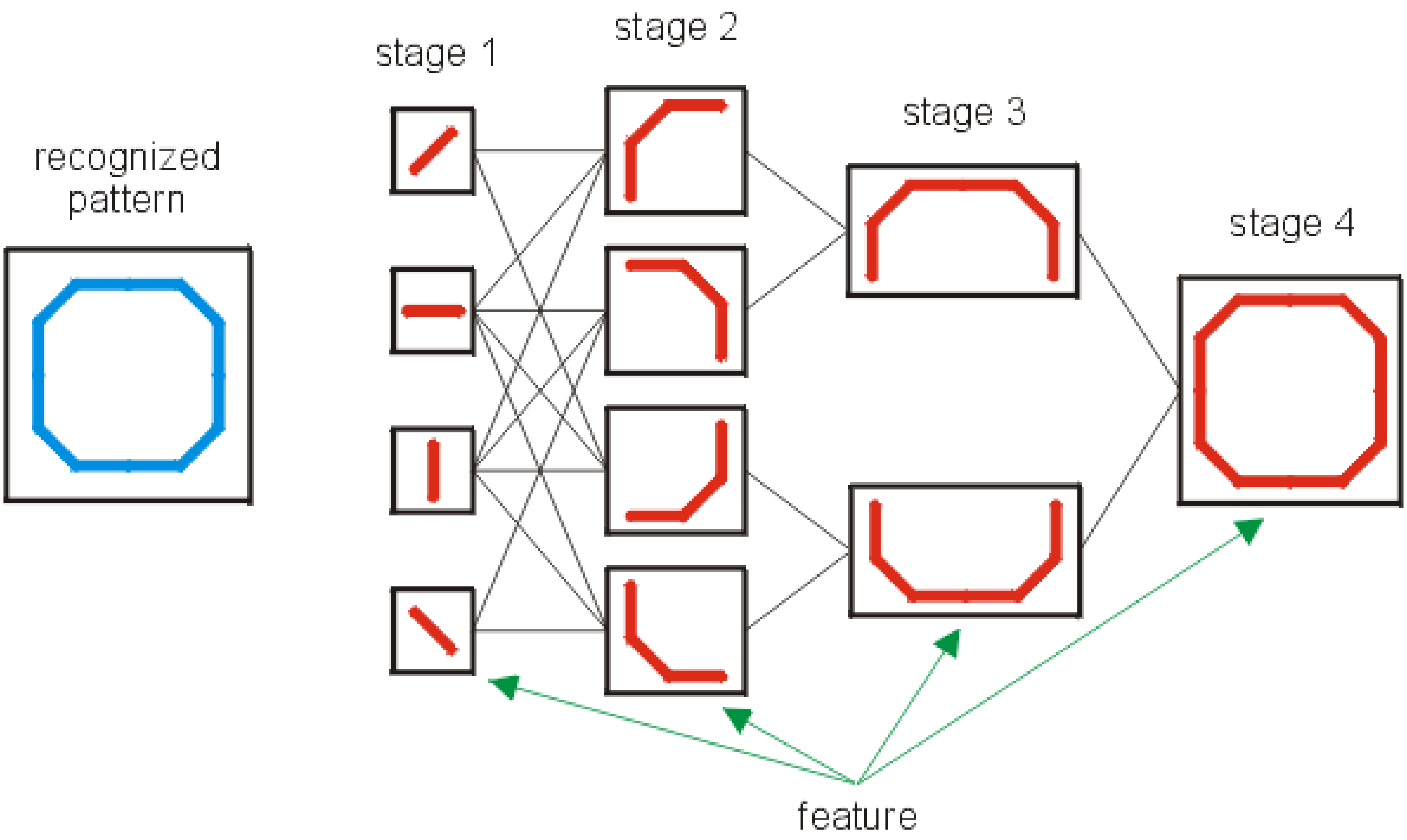
Two motivation points

- 1. Reduced number of parameters
- 2. Stepwise extraction of features
- These two are applicable to any AI situation, and not only vision and image processing

CNN= feedforward like + recurrent like!

- Whatever we learnt so far in FF-BP is useful to understand CNN
- So also is the case with RNN (and LSTM)
- Input divided into regions and fed forward
- Window slides over the input: input changes, but ‘filter’ parameters remain same
- That is like RNN

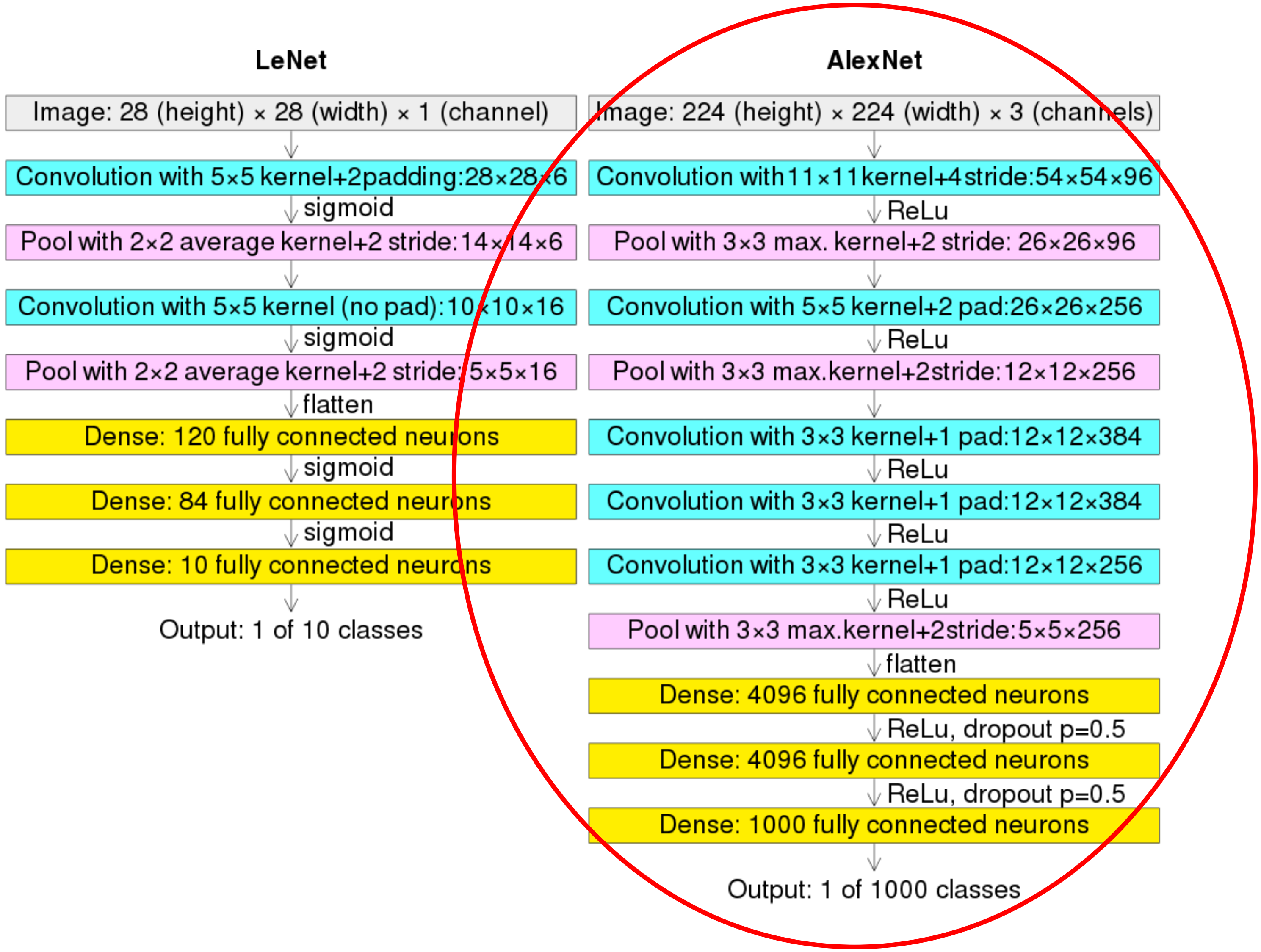
Genesis: Neocognitron (Fukusima, 1980)



Inspiration from biological processes

- Connectivity pattern between neurons resembles the organization of the animal visual cortex
- Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field
- Receptive fields of different neurons partially overlap such that they cover the entire visual field

The classic CNN (Wikipedia)



Convolution

1	**0**	**1**
0	1	0
1	0	1

Filter/kernel/

feature-detector

$1_{\times 1}$	$1_{\times 0}$	$1_{\times 1}$	0	0
$0_{\times 0}$	$1_{\times 1}$	$1_{\times 0}$	1	0
$0_{\times 1}$	$0_{\times 0}$	$1_{\times 1}$	1	1
0	0	1	1	0
0	1	1	0	0

B/W Image

4	3	4
2	4	3
2	3	4

Convolved Feature

4= 1.1+1.0+1.1
+0.0+1.1+1.0
+0.1+0.0+1.1

Convolution basics

Convolution: continuous and discrete

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau) g(t-\tau) d\tau$$

This is the area under the curve $f(\tau)$ **weighted by** $g(t-\tau)$

$$(f * g)[n] = \sum_{m=-\infty}^{+\infty} f(m) g(n-m)$$

Convolution of two vectors

V_1: <0, 1, 2, 3, 4, 5, 6, 7, 8, 9>

V_2: <1, 1, 1>

$V_1 \oplus V_2 =$

$<(0.1+1.1+2.1), (1.1+2.1+3.1), (2.1+3.1+4.1), (3.1+4.1+5.1), (4.1+5.1+6.1), (5.1+6.1+7.1), (6.1+7.1+8.1), (7.1+8.1+9.1)>$

=<3, 6, 9, 12, 15, 18, 21, 24>

Receptive field and selective emphasis/de-emphasis

- The filter <1,1,1> given equal “emphasis” to constituents of the “receptive field” which means region of interest
- Sliding of the filter corresponds to taking different receptive fields
- By designing the filter as <0,1,0>, we emphasise the center of the receptive field

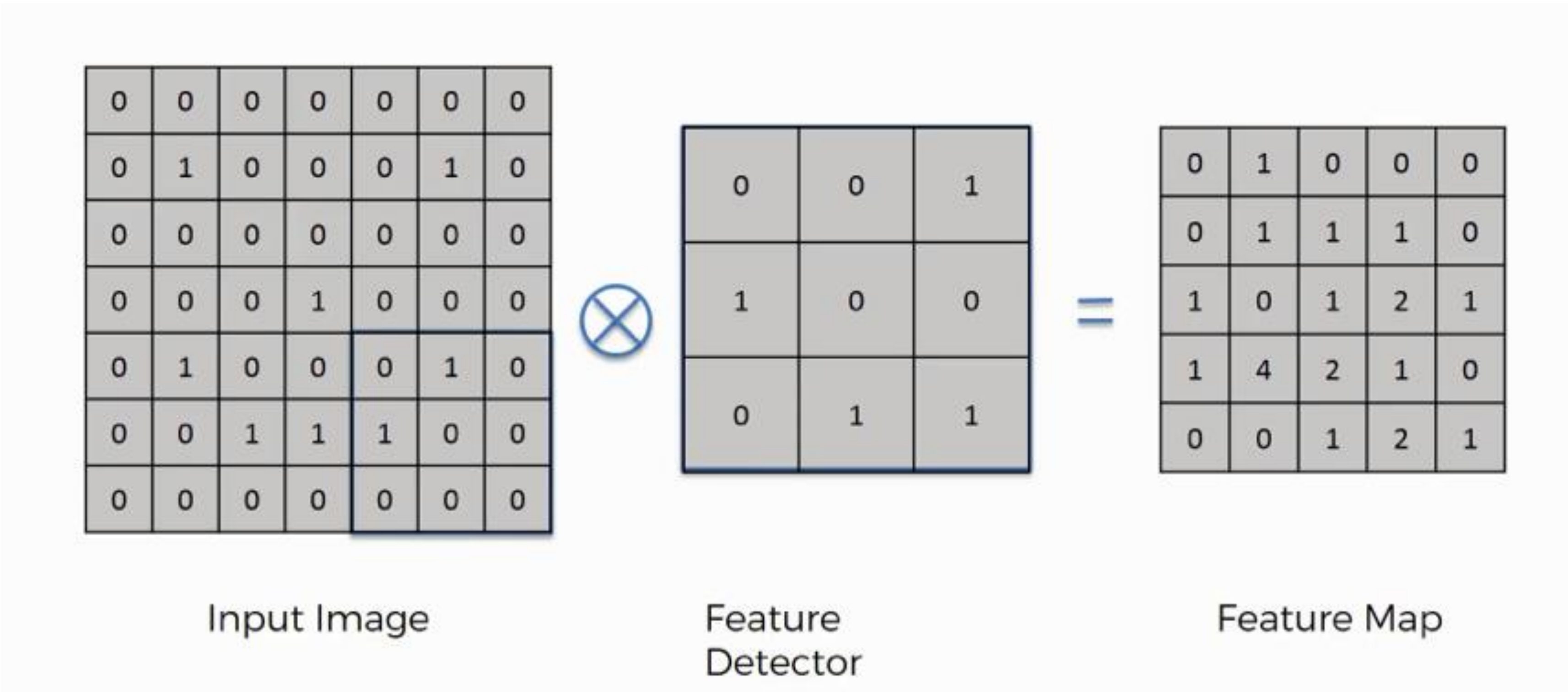
“dog” image and “cat” image

- For dog, the face is of conical shape
- For cat, the shape is round
- So, this distinguishing feature important for classification
- The filter should have the ability of detecting this kind of feature

Interpretation of convolution

- The filter/kernel/feature_extractor highlights features and obtains those features
- The sliding achieves the effect of focussing on “region” after “region”
- This resembles sequence processing
- The filter components are **LEARNT**

Convolution as feature extractor



CNN architecture

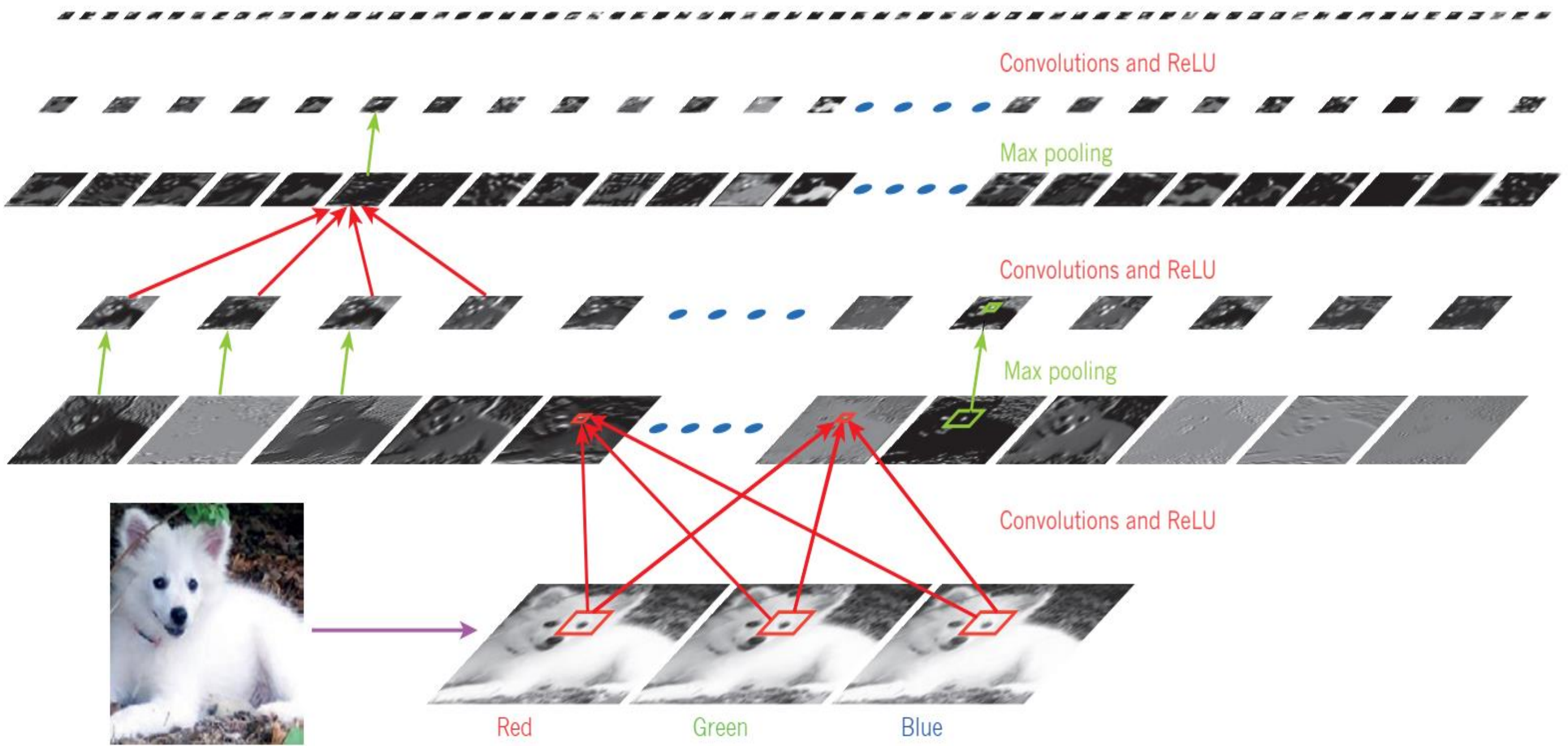
- Several layers of convolution with *tanh* or *ReLU* applied to the results
- In a traditional feedforward neural network we connect each input neuron to each output neuron in the next layer. That's also called a fully connected layer, or affine layer.
- In CNNs we use convolutions over the input layer to compute the output.
- This results in local connections, where each region of the input is connected to a neuron in the output

Key Ideas

Four key ideas that take advantage of the properties of natural signals:

- local connections,
- shared weights,
- pooling and
- the use of many layers

A typical ConvNet



Lecun, Bengio, Hinton, Nature, 2015

Why CNN became a rage: image

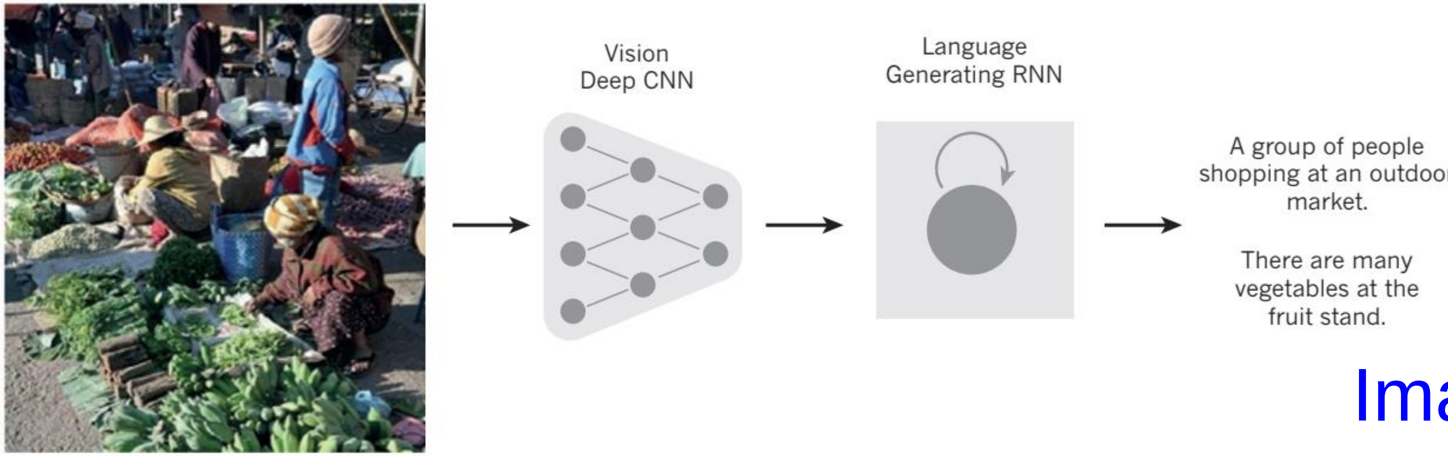


Image Captioning-1

A **stop** sign is on a road with a mountain in the background

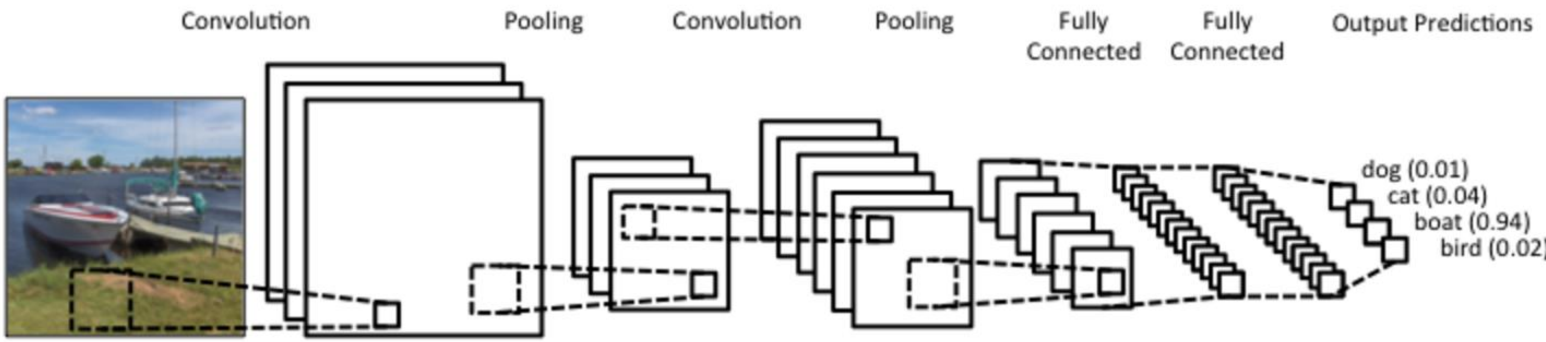
Image Captioning-2

Role of ImageNet

- Million images from the web
- 1,000 different classes
- Spectacular results!
- Almost halving the error rates of the best competing approaches1.

Learning in CNN

- **Automatically learns the values of its filters**
- For example, in Image Classification learn to
  - detect edges from raw pixels in the first layer,
  - then use the edges to detect simple shapes in the second layer,
  - and then use these shapes to deter higher-level features, such as facial shapes in higher layers.
  - The last layer is then a classifier that uses these high-level features.



http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

Pooling

- Gives invariance in translation, rotation and scaling
- Important for image recognition
- Role in NLP?

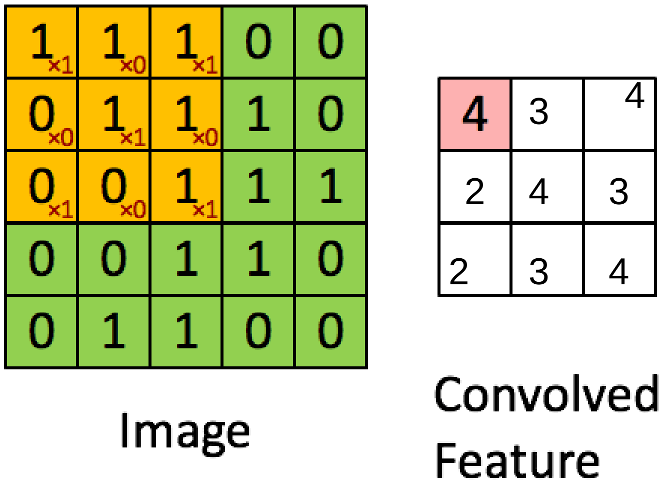
CNN for NLP

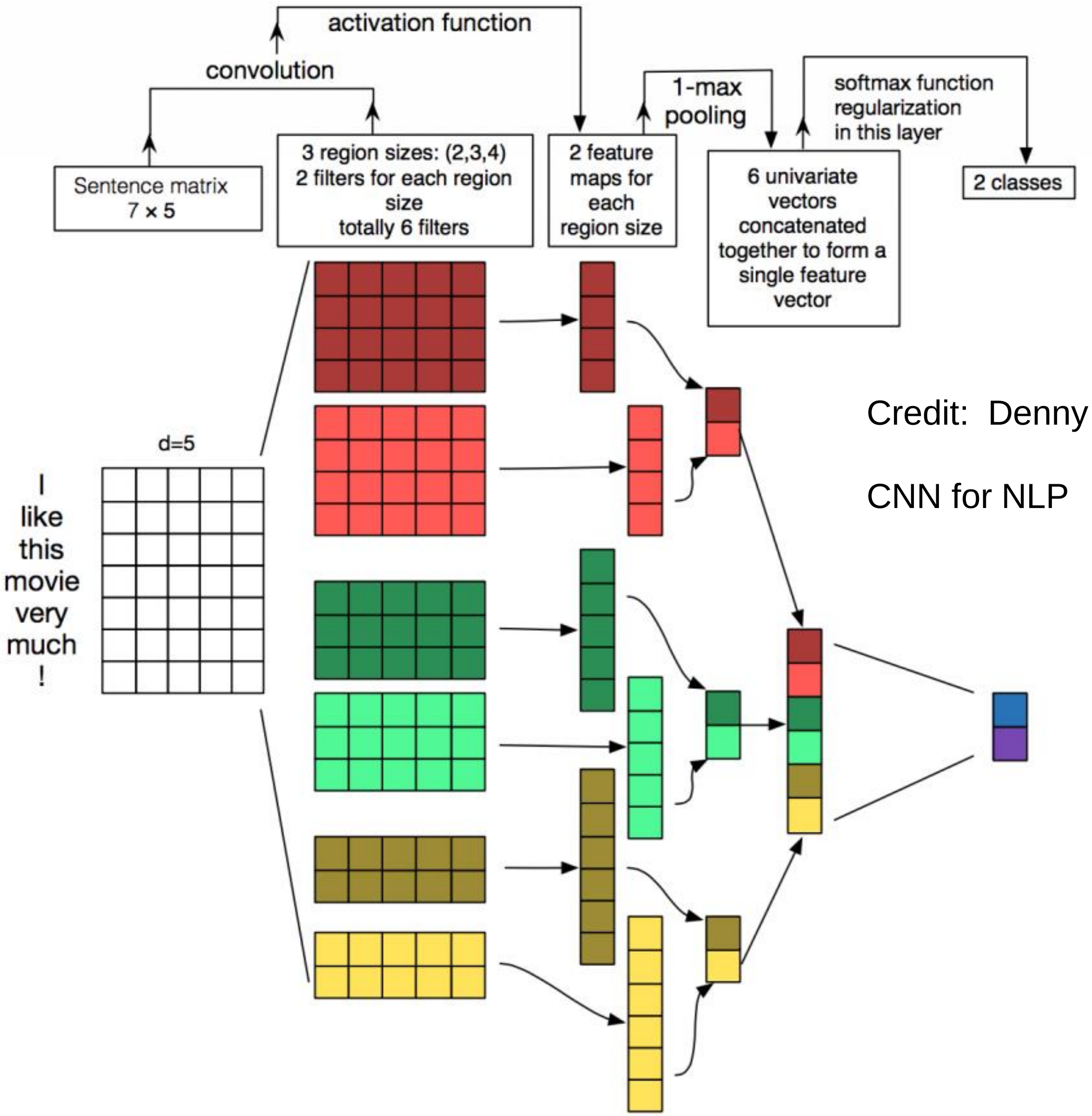
Input matrix for CNN: NLP

- “image” for NLP ←→ word vectors in the rows

- For a 10 word sentence using a 100-dimensional Embedding,

- we would have a 10×100 matrix as our input





Credit: Denny Britz

CNN for NLP

CNN Hyper parameters

- Narrow width vs. wide width
- Stride size
- Pooling layers
- Channels

Detailing out CNN layers

Credit: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

CNN stages

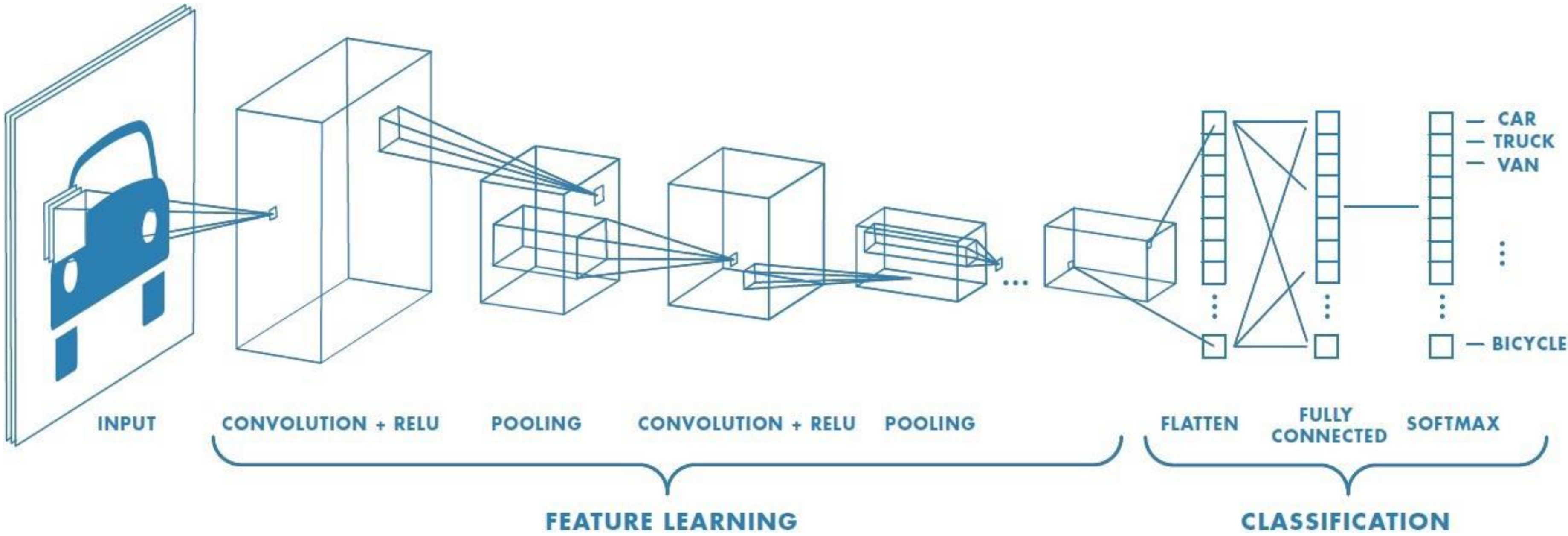


Image Credit: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Another depiction

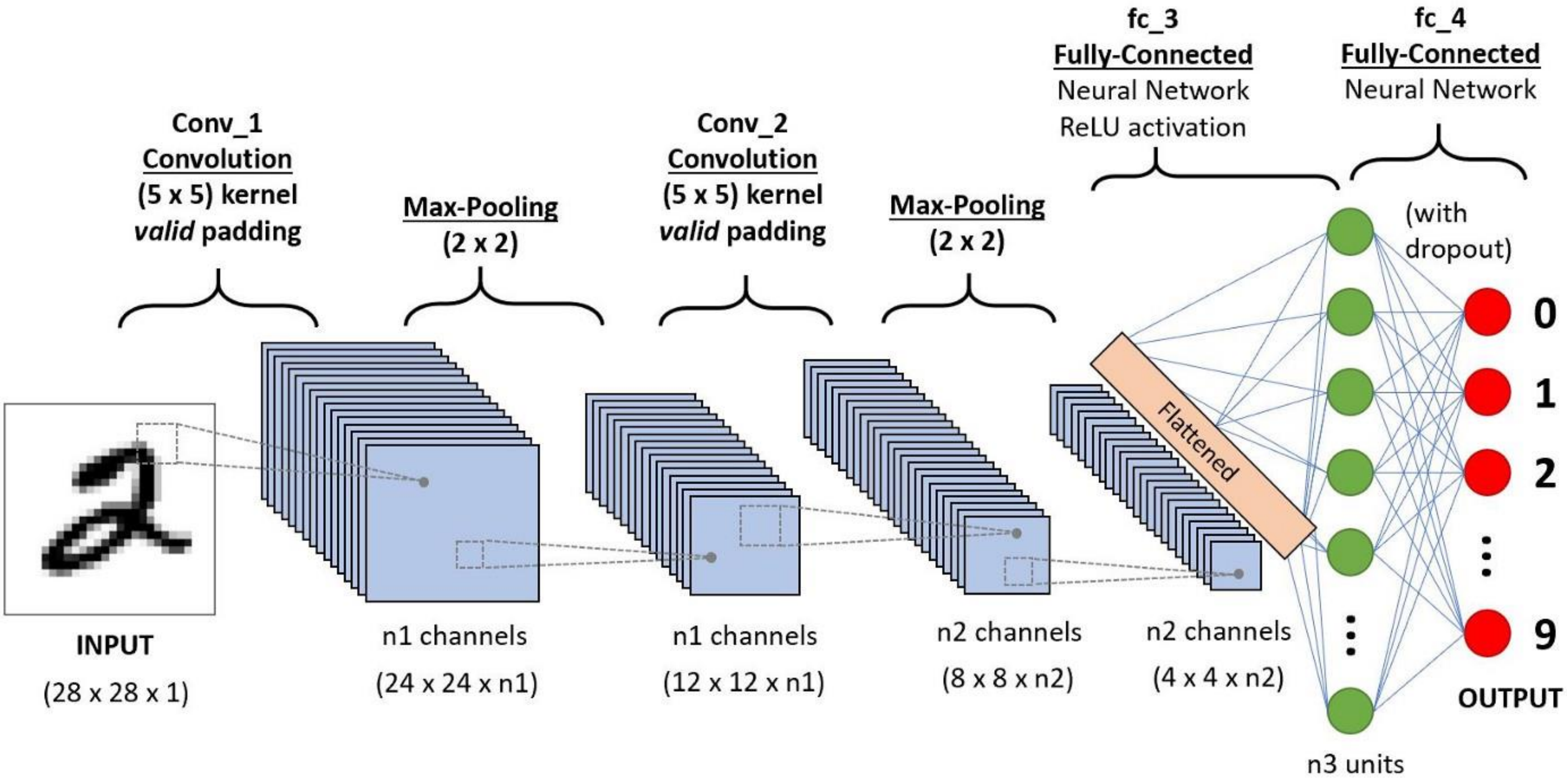
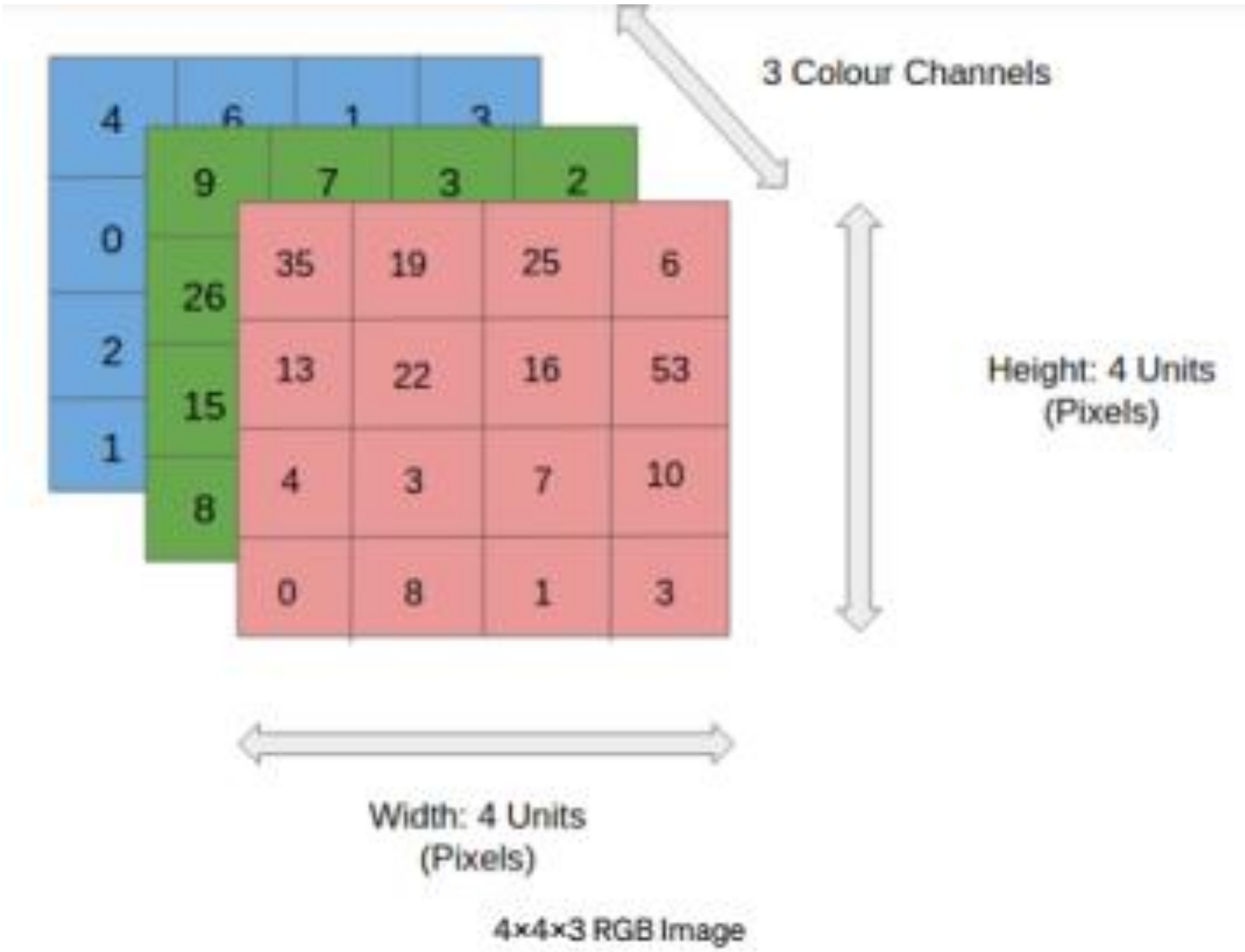
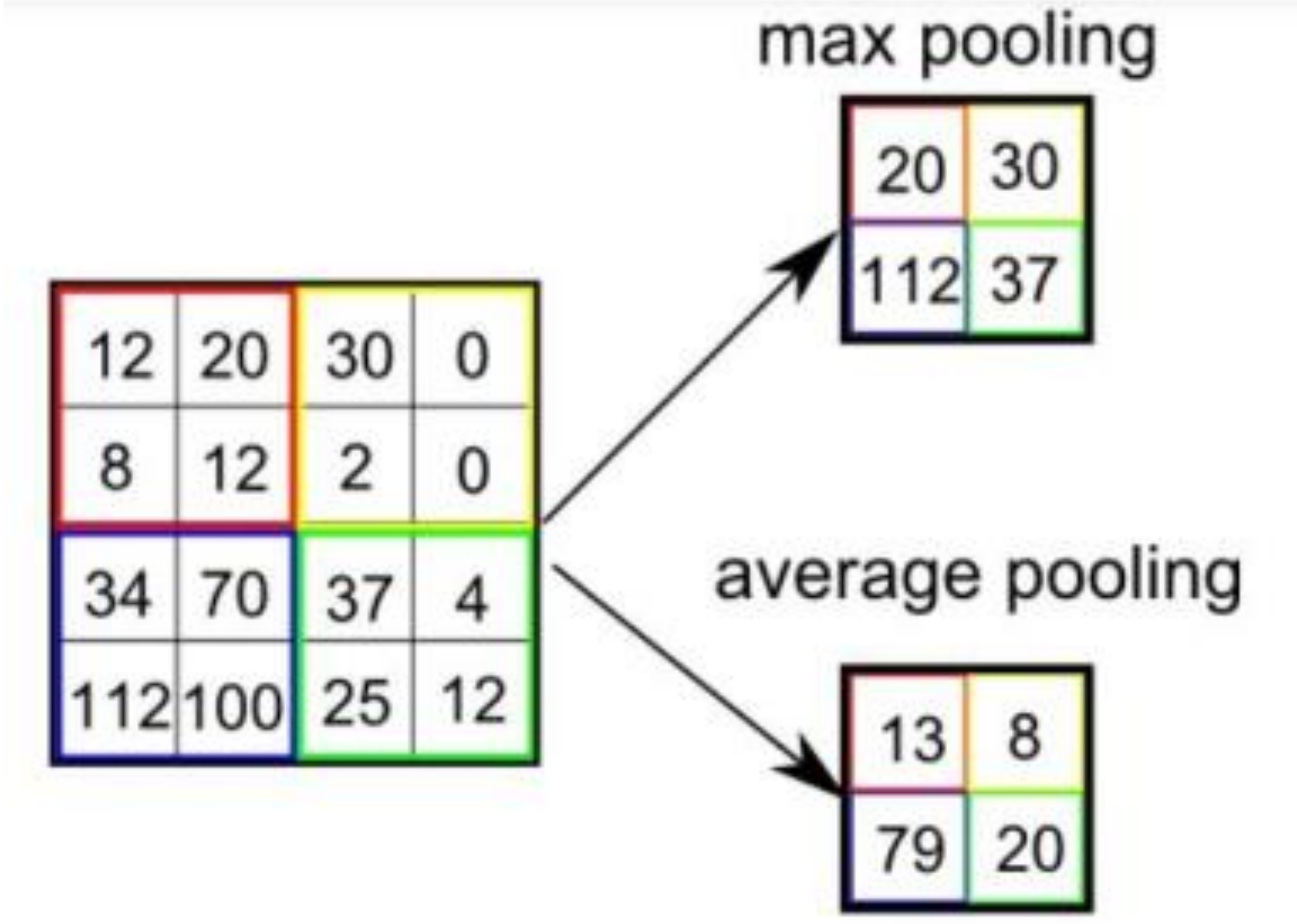


Image Credit: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Channelized Image

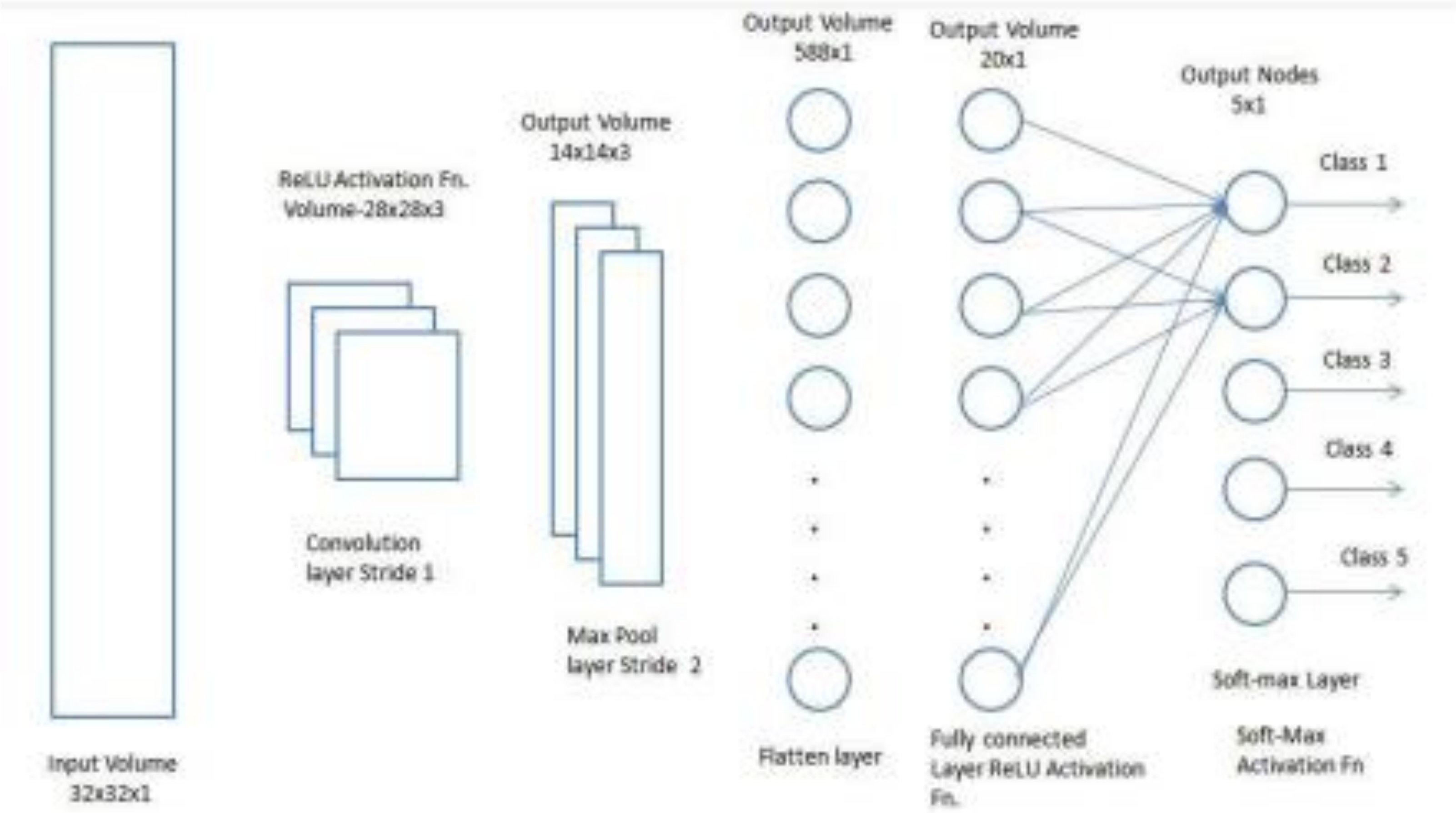


Pooling



Types of Pooling

Complete Architecture



Convolution Layer

- Input is a tensor with a shape
  - (number of inputs) x (input height) x (input width) x (input channels)
- After passing through a convolutional layer, the image becomes abstracted to a feature map, also called an activation map, with shape
  - (number of inputs) x (feature map height) x (feature map width) x (feature map channels).

Tensors and Vectors

- **Tensors**: vectors of vectors
- Vector, *V*: <1, 2, 3, 4, 5>
- Tensor, T1: <<1, 2, 3>, <4, 5, 6>>
- Tensor, T2: <<<1,2>, <3>>, <<4>, <5,6>>>
- **Channels**: R, G, B
- Each image consists of Red, Green and Blue channels- that is, 3 different matrices of pixel values

Pooling Layer

- "Pooling" involves sliding a two-dimensional filter over each channel of feature map
- Effect: summarizing the features
- For a feature map having dimensions $\mathbf{n_h}$ **x** $\mathbf{n_w}$ **x** $\mathbf{n_c}$, the output dimension after pooling is

$$\left(\frac{n_h - f_h + 1}{s}\right) \cdot \left(\frac{n_w - f_w + 1}{s}\right) (.n_c)$$

where, n_h= height of feature map, n_w=width, n_c= number of channels, f_h=height of filter, f_w=width of filter, s=stride length

Sarcasm Detection

Our work spans multiple areas of SA/EA with multiple techniques

Problem-vs-Technique	**Basic Sentiment /Emotion Detection**	**Thwarting**	**Sarcasm**	**Emoji**	**Cross and Multi-Lingual SA/EA**	**SA/EA in Dialogues**
Rule Based	year 2000 onwards	2012	2013			
Classical ML Based				2016	2015	2018
Deep Learning Based						
Hybrid						

Since 2000

Sentiment and Sarcasm

1. Aditya Joshi, Vinita Sharma, Pushpak Bhattacharyya, **ACL 2015**

2. Joe Ross, Abhijit Mishra, and Pushpak Bhattacharyya, **CogACLL 2016**

3.Abhijit Mishra, Kevin Patel, Pushpak Bhattacharyya, **ACL 2016**

4. Abhijit Mishra, Kuntal Dey and Pushpak Bhattacharyya, *Learning Cognitive Features from Gaze Data for Sentiment and Sarcasm Classification Using Convolutional Neural Network*, **ACL 2017**, Vancouver, Canada, July 30-August 4, 2017.

Definition

- **Sentiment Analysis**: The task of identifying if a certain piece of text contains any opinion, emotion or other forms of affective content.

SA- Background

- Research spanning over 2 decades (Liu and Zhang, 2012)
- **Statistical**
  - Supervised (Pang et al.,2002; Benamara et al., 2007; Mullen and Collier, 2004; Pang and Lee, 2008)
  - Unsupervised (Mei et al., 2007; Lin and He, 2009)
- **Supervised**
  - Bag of unigrams, bigrams (Dave et al., 2003; Ng et al., 2006)
  - Syntactic properties (Martineau, 2009; Nakagawa et al.,2010)
  - Semantic propreties (Balamurali ,2011; Ikeda et al. 2008)
- **Deep/Representation Learning:** (**CNN Based**, Maas et al. 2011);

  RNN Based (dos Santos and Gatti 2014)

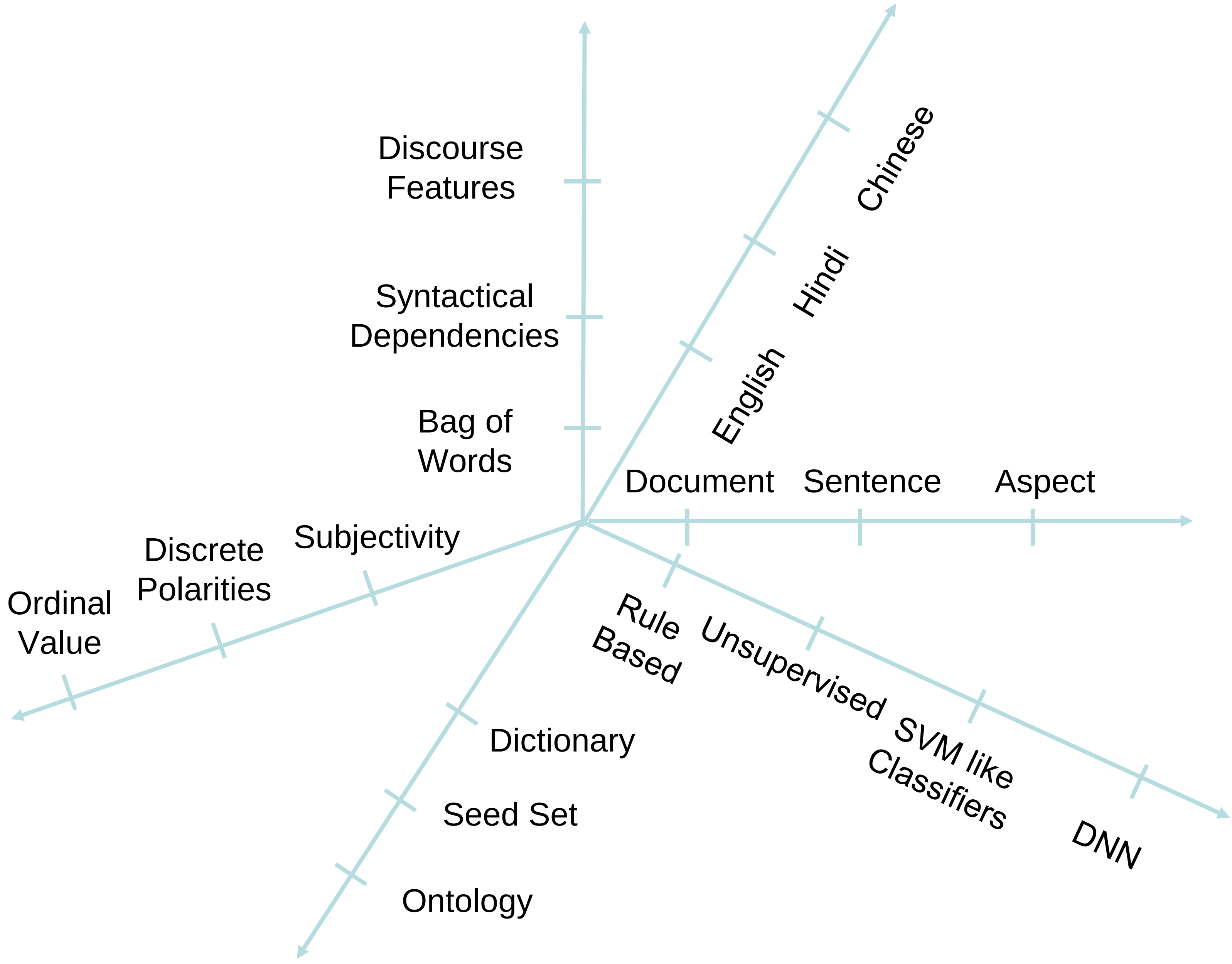
Challenges in SA

- **Lexical Challenges**
  - Data sparsity, (Unseen words)

    The movie is messy, uncouth, incomprehensible, vicious and absurd.
  - Lexical Ambiguity, (Resolving word senses)

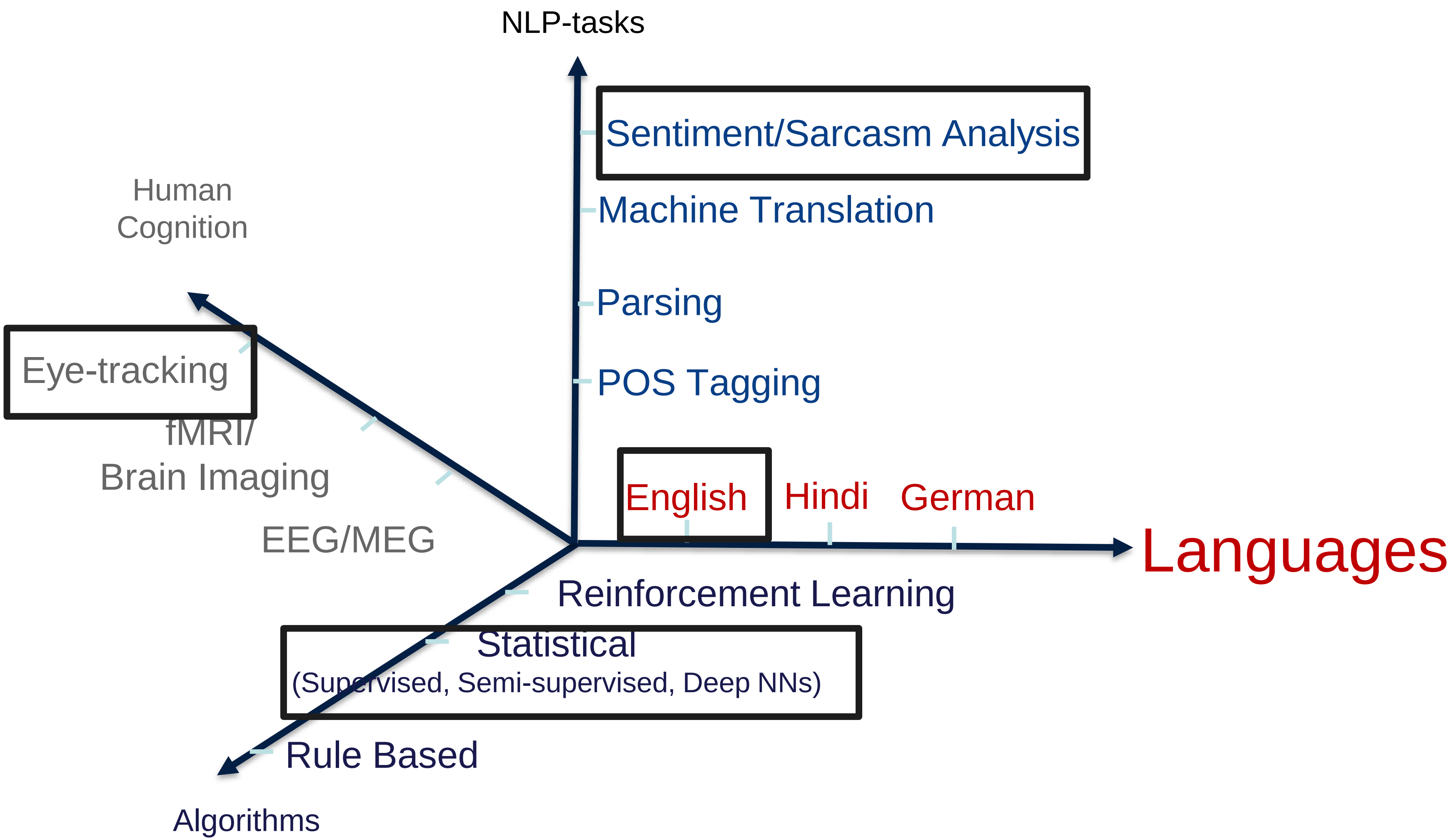
    His face fell when he was dropped from the team VS the boy fell from tree
  - Domain Dependency

    Unpredictable Movie vs. Unpredictable Steering
- **Syntactic Challenges**
  - Complex synactic structure with long distance attachment
    - A somewhat crudely constructed but gripping, questing look at a person so racked with self-loathing, he becomes an enemy to his own race.



Dimensions of Sentiment Analysis

NLP-trinity (augmented)



Sarcasm

Etymology

- Greek: *‘sarkasmós’*: ‘to tear flesh with teeth’

- Sanskrit: *‘vakrokti’*: ‘a twisted (vakra) utterance (ukti)’

Definition- Foundation is Irony

Mean opposite of what is on surface

"A form of irony that is intended to express contempt or ridicule." The Free Dictionary	"Verbal irony that expresses negative and critical attitudes toward persons or events." (Kreuz and Glucksberg, 1989)
"The use of irony to mock or convey contempt." Oxford Dictionary	"Irony that is especially bitter and caustic" (Gibbs, 1994)

Allied concept: **Humble Bragging**- "Oh my life is miserable, have to sign 500 autographs a day!!

Types of Sarcasm

Sarcasm (Camp, 2012)

Propositional	**Embedded**	**Like-prefixed**	**Illocutionary**
A proposition that is intended to be sarcastic. *'This looks like a perfect plan!'*	Sarcasm is embedded in the meaning of words being used. *'I love being ignored'*	'Like/As if' are common prefixes to ask rhetorical questions. *'Like you care'*	Non-speech acts (body language, gestures) contributing to the sarcasm *'(shrugs shoulders) Very helpful indeed!'*

Illocutionary sarcasm

Impact of Sarcasm on Sentiment Analysis (SA) (1/2)

Two SA systems:

MeaningCloud: https://www.meaningcloud.com/

NLTK (Bird, 2006)

Two datasets:

Sarcastic tweets by Riloff et al (2013)

Sarcastic utterances from our dataset of TV transcripts (Joshi et al 2016b)

Impact of Sarcasm on Sentiment Analysis (2/2)

	Precision (Sarc)	Precision (Non-sarc)
Conversation Transcripts		
MeaningCloud[1]	20.14	49.41
NLTK (Bird, 2006)	38.86	81
Tweets		
MeaningCloud[1]	17.58	50.13
NLTK (Bird, 2006)	35.17	69

[1] www.meaningcloud.com

Clues for Sarcasm

- Use of laughter expression
  - *haha, you are very smart xD*
  - *Your intelligence astounds me. LOL*
- Heavy Punctuation
  - *Protein shake for dinner!! Great!!!*
- Use of emoticons
  - *i LOVE it when people tweet yet ignore my text X-(*
- Interjections
  - *3:00 am work YAY. YAY.*
- Capital Letters
  - *SUPER EXCITED TO WEAR MY UNIFORM TO SCHOOL TOMORROW ! ! :D lol.*

Incongruity: at the heart of things!

- *I love being ignored*
- *3:00 am work YAY. YAY.*
- *Up all night coughing. yeah me!*
- *No power, Yes! Yes! Thank you storm!*
- *This phone has an awesome battery back-up of 2 hour (Sarcastic)*

Two kinds of incongruity

- **Explicit incongruity**
  - Overtly expressed through sentiment words of both polarities
  - Contribute to almost 11% of sarcasm instances

    'I <u>love</u> being <u>ignored</u>'
- **Implicit incongruity**
  - Covertly expressed through phrases of implied sentiment

    'I <u>love</u> this paper so much that I <u>made a doggy bag out of it</u>'

Sarcasm Detection Using Semantic incongruity

Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya and Mark Carman, *Are Word Embedding-based Features Useful for Sarcasm Detection?*, **EMNLP 2016**, Austin, Texas, USA, November 1-5, 2016.

Also covered in: How Vector Space Mathematics Helps Machines Spot Sarcasm, MIT Technology Review, 13th October, 2016.

www.cfilt.iitb.ac.in/sarcasmsuite/

Feature Set

Lexical	
Unigrams	Unigrams in the training corpus
Pragmatic	
Capitalization	Numeric feature indicating presence of capital letters
Emoticons & laughter expressions	Numeric feature indicating presence of emoticons and 'lol's
Punctuation marks	Numeric feature indicating presence of punctuation marks
Implicit Incongruity	
Implicit Sentiment Phrases	Boolean feature indicating phrases extracted from the implicit phrase extraction step
Explicit Incongruity	
#Explicit incongruity	Number of times a word is followed by a word of opposite polarity
Largest positive /negative subsequence	Length of largest series of words with polarity unchanged
#Positive words	Number of positive words
#Negative words	Number of negative words
Lexical Polarity	Polarity of a tweet based on words present

Datasets

Name	Text-form	Method of labeling	Statistics
Tweet-A	Tweets	Using sarcasm-based hashtags as labels	5208 total, 4170 sarcastic
Tweet-B	Tweets	Manually labeled (Given by Riloff et al(2013))	2278 total, 506 sarcastic
Discussion-A	Discussion forum posts (IAC Corpus)	Manually labeled (Given by Walker et al (2012))	1502 total, 752 sarcastic

Results

Features	P	R	F
Original Algorithm by Riloff et al. (2013)			
Ordered	0.774	0.098	0.173
Unordered	0.799	0.337	0.474
Our system			
Lexical (**Baseline**)	0.820	0.867	0.842
Lexical+Implicit	0.822	0.887	0.853
Lexical+Explicit	0.807	0.985	0.8871
All features	0.814	0.976	**0.8876**

Tweet-A

Approach	P	R	F
Riloff et al. (2013) (**best reported**)	0.62	0.44	0.51
Maynard and Greenwood (2014)	0.46	0.38	0.41
Our system (all features)	**0.77**	**0.51**	**0.61**

Tweet-B

Features	P	R	F
Lexical (**Baseline**)	0.645	0.508	0.568
Lexical+Explicit	0.698	0.391	0.488
Lexical+Implicit	0.513	0.762	0.581
All features	0.489	0.924	**0.640**

Discussion-A

Abhijit Mishra, Kuntal Dey and Pushpak Bhattacharyya, *Learning Cognitive Features from Gaze Data for Sentiment and Sarcasm Classification Using Convolutional Neural Network*, **ACL 2017**, Vancouver, Canada, July 30-August 4, 2017.