Imperial College London

Natural Language Processing

Week 2: Classification

Introduction - Joe

- NLP PhD student supervised by Marek
- Research interests in:
    - Removing dataset biases
    - Creating more interpretable models
    - Making models generalise better to new datasets
- Current work combines logical reasoning and neural networks
- Ex teacher/consultant

Classification

Outline

1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNs)
6. CNNs
7. Our recent research

On Thursday: Coursework walk-through, evaluation and de-biasing methods. The coursework will involve a classification task

What is classification?

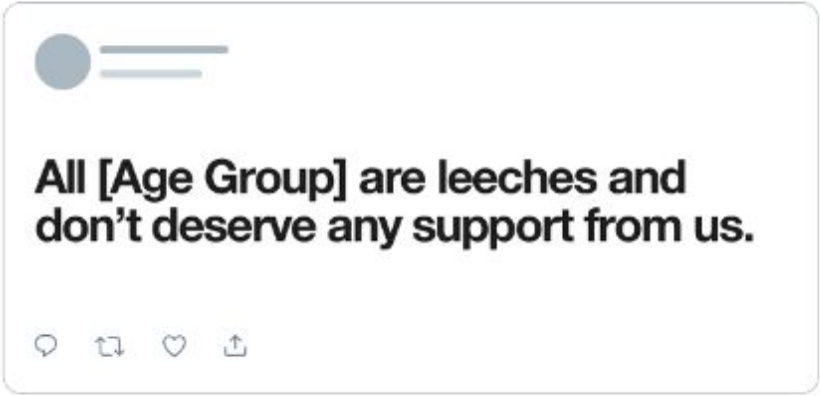
Classification:

Predicting which ‘class’ an observation belongs to.

$$\hat{y} = argmax_y P(y|x)$$

Examples of binary classification

Predicting if a text (or tweet) contains hate speech:



Hate speech

Or

Not hate speech

Examples of binary classification

Predicting if a text (or tweet) contains hate speech:

Model produces a score (logit)

↓

Sigmoid makes this score between 0 and 1

↓

0.5 is our decision boundary

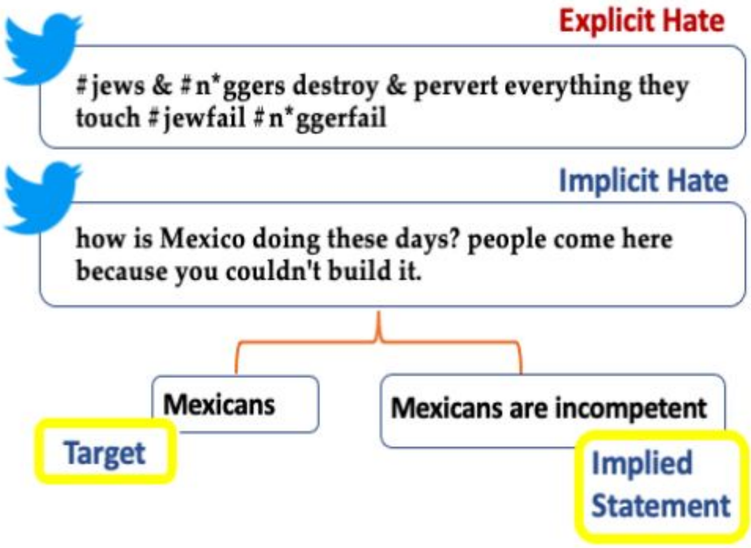
Hate speech

Or

Not hate speech

Examples of multi-class classification

Predicting if a text (or tweet) contains hate speech:



Explicit hate speech

Or

Implicit hate speech

Or

Not hate speech

Source: *Latent Hatred: A Benchmark for Understanding Implicit Hate Speech, EMNLP 2021*

Common NLP classification tasks

My pick of common NLP classification tasks:

1. Hate speech detection
2. Sentiment analysis
3. Fact verification
4. Spam detection
5. Error detection
6. Natural Language Inference (NLI)

More NLP classification tasks / uses

1. Topic classification
2. Emotion detection
3. Paraphrase detection
4. Plagiarism detection
5. Multi-choice questions
6. Identifying presence of illness
7. Identifying children at risk (using social services data)
8. Intended sarcasm detection
9. Joke and humour detection
10. Patronising content detection
11. Fake news detection
12. Propaganda detection
13. Purpose of dark web pages
14. Predicting legal judgements

And many more…

Natural Language Inference

- **Premise**: The kitten is climbing the curtains again
- **Hypothesis**: The kitten is sleeping

Labels:

- **Entailment**: if the hypothesis is implied by the premise
- **Contradiction**: if the hypothesis contradicts the premise
- **Neutral**: otherwise

Questions so far?

10 mins

Outline

1. NLP classification tasks
2. **Naive Bayes**
3. Logistic Regression
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNS)
6. CNNs
7. Our recent research

Naive Bayes Classifier

Introducing Naive Bayes

Naive Bayes Classifier

Bayes' rule:

likelihood prior

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}$$

posterior evidence

Naive Bayes Classifier

$$\hat{y} = argmax_y P(y|x) = argmax_y \overset{\text{likelihood}}{P(x|y)}\,\overset{\text{prior}}{P(y)}$$

Naive Bayes Classifier

- x is a set of features $x_1, x_2, \ldots, x_I$

$$\hat{y} = argmax_y P(x_1, x_2, \ldots, x_I | y) P(y)$$

- Naive Bayes **independence assumption:**

$$\boxed{P(x_1, x_2, \ldots, x_I | y) = P(x_1|y) \cdot P(x_2|y) \cdot \ldots \cdot P(x_I|y)}$$

$$\hat{y} = argmax_y P(y) \prod_{i=1}^{I} P(x_i|y)$$

Input representations

- **Question 1**: Does conditional independence imply independence?
  - E.g: If P(A | C) and P(B | C) are independent,
    Are P(A) and P(B) independent?

Input representations

- **Question 1**: Does conditional independence imply independence?
    - So P(A | C) and P(B | C) are independent in this example
    - But P(A) and P(B) are not

Example 1.27

A box contains two coins: a regular coin and one fake two-headed coin ($P(H) = 1$). I choose a coin at random and toss it twice. Define the following events.

- A= First coin toss results in an H.
- B= Second coin toss results in an H.
- C= Coin 1 (regular) has been selected.

Source:
https://www.probabilitycourse.com/chapter1/1_4_4_conditional_independence.php

Input representations

- **Question 2**: Does independence imply conditional independence?
    - E.g: If P(A) and P(B) are independent,
      Are P(A|C) and p(B|C) independent?

Input representations

- **Question 2**: Does independence imply conditional independence?
    - E.g: If P(A) and P(B) are independent,
      Are P(A|C) and p(B|C) independent?

$$\boxed{P(A \cap B) = P(A)P(B) \quad \textbf{(Eq.1)}}$$

Consider rolling a dice, where:

A = {if 1 or 2 are rolled}
B = {if 2, 4, or 6 are rolled}
C = {if 1 or 4 are rolled}

Source:
https://www.probabilitycourse.com/chapter1/1_4_4_conditional_independence.php

Naive Bayes Classifier

Naive Bayes for language

(using a sentiment analysis example - movie reviews)

Input representations

- Raw input is transformed into a numerical representation,

  i.e. each input x is represented by a feature vector:

$$[x_1, x_2, \ldots, x_I]$$

Input representations

After any pre-processing:

- We can use a Bag of Words (BoW) approach

Another good movie for holiday watchers.... a little twist from the ordinary scrooge movie. Enjoyable.

Input representations

Example Bag of Words representation:

Review #1:

This **was another** good movie for holiday watchers. There **was a** nice little twist at the end.

	a	**about**	**another**	**and**	**...**	**was**	**you**
Review #1	1	0	1	0		2	0

Naive Bayes Classifier

Collecting statistics from our training data

Naive Bayes Classifier

Training corpus	class
Another good movie for holiday watchers. A little twist from the ordinary scrooge movie. Enjoyable.	+
It seems like just about everybody has made a Christmas Carol movie. Others are just bad and the time period seems to be perfect.	+
If you're looking for the same feel good one but in a new setting, this one's for you.	+
This is a first for me, I didn't like this movie. It was really bad.	-
It was good but the Christmas Carol by Dickens was emotionally moving.	-

Naive Bayes Classifier

With some limited data processing….

Training corpus	class
another good movie for holiday watchers . a little twist from the ordinary scrooge movie . enjoyable .	+
it seems like just about everybody has made a christmas carol movie . others are just bad and the time period seems to be perfect .	+
if you're looking for the same feel good one but in a new setting , this one's for you .	+
this is a first for me , i didn't like this movie . it was really bad .	-
it was good but the christmas carol by dickens was emotionally moving .	-

Naive Bayes Classifier

Our bag of words representation….

Review	a	about	another	and	…	was	you
1	1	0	1	0		0	0
2	1	1	0	1		0	0
3	1	0	0	0		0	2
4	1	0	0	0		1	0
5	0	0	0	0		1	0

Naive Bayes Classifier

Alternatively, using some feature extraction....

Training corpus	**good**	**movie**	**bad**	class
another **good movie** for holiday watchers . a little twist from the ordinary scrooge movie . enjoyable .	1	1	0	**+**
it seems like just about everybody has made a christmas carol movie . others are just **bad** and the time period seems to be perfect .	0	0	1	**+**
if you 're looking for the same feel **good** one but in a new setting , this one 's for you .	1	0	0	**+**
this is a first for me , i didn 't like this **movie** . it was really **bad** .	0	1	1	**-**
it was **good** but the christmas carol by dickens was emotionally moving .	1	0	0	**-**

Naive Bayes Classifier

Training a 'model' just involves collecting statistics

$$P(y)$$

$$P(+) = \frac{3}{5}$$

3 positive examples

$$P(-) = \frac{2}{5}$$

2 positive examples

Naive Bayes Classifier

Frequency of the word for this class

$$P(good|+) = \frac{2}{4}$$

Total count of words for this class

Half (two out of four) of the words within the positive class are ‘good’

Naive Bayes Classifier

What happens if one of our probabilities is 0?

Naive Bayes Classifier

Add-one smoothing:

$$P(x_i|y) = \frac{count(x_i,y)+1}{\sum_{x \in V}(count(x,y)+1)} = \frac{count(x_i,y)+1}{(\sum_{x \in V} count(x,y))+|V|}$$

V is the vocabulary across both classes

Naive Bayes Classifier

Add-one smoothing:

$$P(x_i|y) = \frac{count(x_i,y)+1}{\sum_{x\in V}(count(x,y)+1)} = \frac{count(x_i,y)+1}{(\sum_{x\in V} count(x,y))+|V|}$$

V is the vocabulary across classes

$$P(good|+) = \frac{2+1}{4+3}$$

Naive Bayes Classifier

Test example:

Not as **good** as the old **movie**, rather **bad**.

Pre-process & extract some features

good **movie** **bad**

Naive Bayes Classifier

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(good|+) = \frac{2+1}{4+3}$$

$P(movie|+) =$ R

$P(bad|+) =$ L

$$P(good|-) = \frac{1+1}{3+3}$$

$P(movie|-) =$ R

$P(bad|-) =$ L

Naive Bayes Classifier

Alternatively, using some feature extraction….

Training corpus	**good**	**movie**	**bad**	class
another **good movie** for holiday watchers . a little twist from the ordinary scrooge movie . enjoyable .	1	1	0	+
it seems like just about everybody has made a christmas carol movie . others are just **bad** and the time period seems to be perfect .	0	0	1	+
if you 're looking for the same feel **good** one but in a new setting , this one 's for you .	1	0	0	+
this is a first for me , i didn 't like this **movie** . it was really **bad** .	0	1	1	-
it was **good** but the christmas carol by dickens was emotionally moving .	1	0	0	-

Naive Bayes Classifier

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(good|+) = \frac{2+1}{4+3}$$

$$P(movie|+) = \frac{1+1}{4+3}$$

$$P(bad|+) = \frac{1+1}{4+3}$$

$$P(good|-) = \frac{1+1}{3+3}$$

$$P(movie|-) = \frac{1+1}{3+3}$$

$$P(bad|-) = \frac{1+1}{3+3}$$

Naive Bayes Classifier

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(+) = \frac{3}{5}$$

$$P(+)P(x|+) = \frac{3}{5} \times \frac{3 \times 2 \times 2}{7^3}$$

$$P(+)P(x|+) = 0.021$$

$$P(-) = \frac{2}{5}$$

$$P(-)P(x|-) = \frac{2}{5} \times \frac{2 \times 2 \times 2}{6^3}$$

$$P(-)P(x|-) = 0.014$$

Naive Bayes Classifier

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(+) = \frac{3}{5}$$

$$P(+)P(x|+) = \frac{3}{5} \times \frac{3 \times 2 \times 2}{7^3}$$

$$P(+)P(x|+) = 0.021$$

$$P(-) = \frac{2}{5}$$

$$P(-)P(x|-) = \frac{2}{5} \times \frac{2 \times 2 \times 2}{6^3}$$

$$P(-)P(x|-) = 0.014$$

Naive Bayes Classifier

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(+) = \frac{3}{5}$$

$$P(-) = \frac{2}{5}$$

$$P(+)P(x|+) = \frac{3}{5} \times \frac{3\times2\times2}{7^3}$$

$$P(-)P(x|-) = \frac{2}{5} \times \frac{2\times2\times2}{6^3}$$

$$P(+)P(x|+) = 0.021$$

$$P(-)P(x|-) = 0.014$$

Improvements

Some improvements we can make for sentiment analysis….

Improvement #1

How about: Not as **good** as the old **movie**, rather **bad movie**.

$$P(+) = \frac{3}{5}$$

$$P(+)P(x|+) = \frac{3}{5} \times \frac{3 \times 2 \times 2}{7^3}$$

$$P(-) = \frac{2}{5}$$

$$P(-)P(x|-) = \frac{2}{5} \times \frac{2 \times 2 \times 2}{6^3}$$

This variant is called “Binary Naive Bayes”

Improvement #2

Review:

I didn’t like the movie, but it was better than Top Gear

Becomes:

I didn’t NOT_like NOT_the NOT_movie, but it was better than Top Gear

We append ‘NOT_’ after any logical negation (e.g. ***n’t***, ***not***, ***no***, ***never***) until the next punctuation mark

Problems

- Conditional independence assumption
- Context not taken into account
- New words (not seen at training) cannot be used

Questions so far?

35 mins

Outline

1. NLP classification tasks
2. Naive Bayes
3. **Logistic Regression**
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNS)
6. CNNs
7. Our recent research

Logistic Regression

- Discriminative vs Generative algorithms
- Discriminative algorithm to directly learn what features from the input are most useful to discriminate between the different classes

Generative models make use of:

$$P(x|y)$$

Logistic Regression

Logistic function

$$y(x) = g(z) = \frac{1}{1+e^{-z}}$$

$$z = \mathbf{w} \cdot \mathbf{x} + b$$

w = How important an input feature is to the classification decision

Logistic function is a linear transformation followed by a sigmoid…

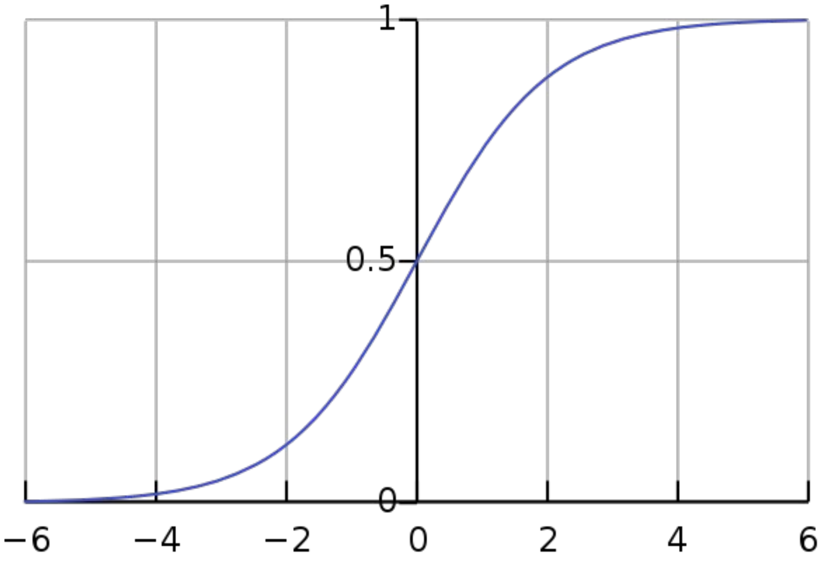
Sigmoid Function

$$P(y=1) = \frac{1}{1+e^{-(w \cdot x + b)}}$$

How do we make a decision?

$$\hat{y} = \begin{cases} 1 & \text{if } P(y=1|x) \text{ is } > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Sigmoid:



Logistic Regression

- What weights would we expect for our three features?
    - ***movie***, ***bad*** *and* ***good***

For simplicity, in our worked example we only consider the features ***bad*** and ***movie***

** We have y=1 if a movie is POSITIVE

Logistic Regression

How inference works if we have learnt w and b:

- Test example: *Not as good as the old **movie**, rather **bad**.*

x_1	count of **movie**	1
x_2	count of **bad**	1

$$x = [1.0, 1.0]$$
$$w = [-5.0, 2.5]$$
$$b = 0.1$$

Logistic Regression

Test example: *Not as good as the old **movie**, rather **bad**.*

$$x = [1.0, 1.0] \qquad w = [-5.0, 2.5] \qquad b = 0.1$$

$$P(y=1|x) = g(z)$$
$$= g([-5.0, 2.5] \cdot [1.0, 1.0] + 0.1)$$
$$= g(-2.4)$$
$$= 0.08$$

$$P(y=0|x) = 1 - g(z)$$
$$= 0.92$$

Logistic Regression

We learn parameters to make the model predictions close to the gold output:

- Our loss function measures the distance between true and predicted label
- Optimization algorithm minimises this function, usually **gradient descent**

Logistic Regression

How close is the predicted distribution Q to the true distribution P?

$$H(P,Q) = -\sum_i P(y_i)\log Q(y_i)$$

Logistic Regression

Finding the loss from our example:

$$P = [1\ 0]$$

$$Q_1 = [0.92\ 0.08]$$

$$H(P, Q) = -(1\, log\, 0.92 + 0\, log\, 0.08)$$
$$= -\, log\, 0.92 = 0.08$$

Logistic Regression

Finding the loss from our example:

$$P = [1 \; 0]$$

$$Q_2 = [0.56 \; 0.44]$$

$$H(P,Q) = -(1\,log\,0.56 + 0\,log\,0.44)$$
$$= -\,log\,0.56 = 0.58$$

Logistic Regression

Sentiment analysis with 3 classes: +, - and neutral:

Input features:

x_1	count of **bad**	1
x_2	count of **good**	1
x_3	count of **and**	2

Logistic Regression

Weights are learnt per class

$$w_{+} = \begin{bmatrix} 0.0 \\ 1.9 \\ 0.0 \end{bmatrix} \qquad w_{-} = \begin{bmatrix} 1.5 \\ 0.4 \\ 0.0 \end{bmatrix} \qquad w_{n} = \begin{bmatrix} 0.0 \\ 0.0 \\ 0.0 \end{bmatrix}$$

Logistic Regression

$$z_1 = ([1.0, 1.0, 2.0] \cdot [0.0, 1.9, 0.0]) + 0.1$$

$$z_2 = ([1.0, 1.0, 2.0] \cdot [1.5, 0.4, 0.0]) \ \mathbf{-0.9}$$

$$z_3 = ([1.0, 1.0, 2.0] \cdot [0.0, 0.0, 0.0]) + 0.1$$

Logistic Regression

Probability distribution over classes:

$$\mathbf{z} = [2.0,\ 1.0,\ 0.1]$$

Softmax function

$$y = g(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \qquad 1 \leq i \leq k$$

Replaces our sigmoid function: $y = g(z) = \frac{1}{1+e^{-z}}$

Logistic Regression

$$y = g(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$$

$$\mathbf{z} = [2.0,\ 1.0,\ 0.1]$$

$$y = [\frac{e^{2.0}}{e^{2.0}+e^{1.0}+e^{0.1}},\ \frac{e^{1.0}}{e^{2.0}+e^{1.0}+e^{0.1}},\ \frac{e^{0.1}}{e^{2.0}+e^{1.0}+e^{0.1}}] = [0.66,\ 0.24,\ 0.1]$$

Logistic Regression

Logistic Regression:

- Considers the importance of features, so better at dealing with correlated features
- Better with larger datasets

Naive Bayes:

- Very quick to train
- Some evidence it works well on small datasets

Some problems remain, e.g. considering the interaction of different features

Questions so far?

45 mins

Simple NLP baselines

Why should I care....

When might you use them?

Why could this still be useful:

1. They can help us to understand which features are influential or correlate with each class

   This can help us better understand our dataset and which features are important

2. We can compare to more powerful models to understand the nature of the task

Simple NLP baselines

Example why we might want to better understand our data....

Natural Language Inference

- **Premise**: The kitten is climbing the curtains again
- **Hypothesis**: The kitten is sleeping

Label: *Contradiction*

What we can find out:

- The word *sleeping* strongly correlates with the contradiction class
- So do words such as *No*, *Nobody*, *Never*, *Nothing*, *Aliens*, *Mars*

Break

50 mins

Outline

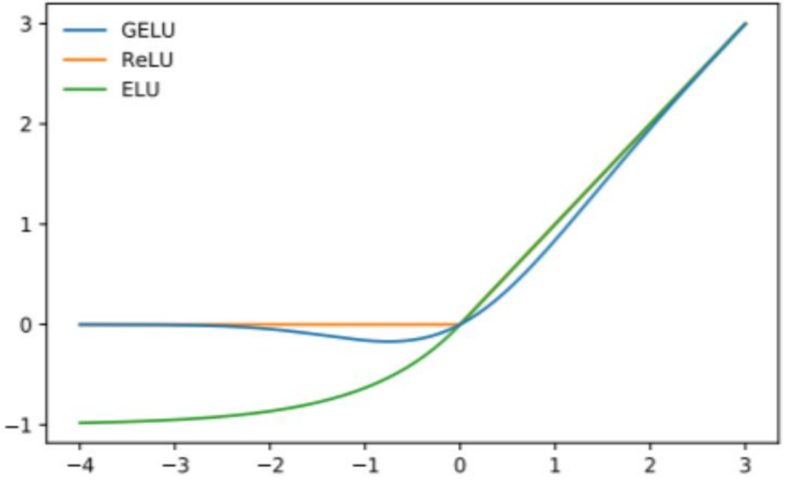
1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. **Neural Networks (NNs)**
5. Recurrent neural networks (RNNS)
6. CNNs
7. Our recent research

Neutral Networks

Linear layer: $z = \mathrm{w} \cdot \mathrm{x} + b = \sum_{i=0}^{I} w_i x_i + b$

Non-linear activation function: $y = g(z)$

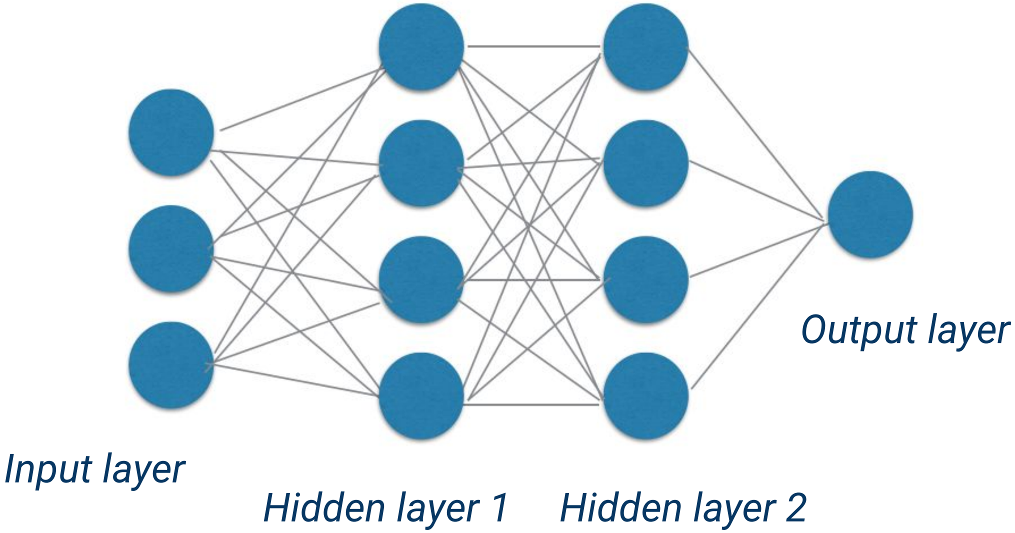
'GELU' is used in BERT:



Neutral Networks

Fully-connected layers

$$h^1 = g^1(x\mathsf{W}^1 + \mathsf{b}^1) \quad h^2 = g^2(h^1\mathsf{W}^2 + \mathsf{b}^2) \quad y = h^2\mathsf{W}^3$$



$$FFN(x) = (g^2(g^1(x\mathsf{W}^1 + \mathsf{b}^1))\mathsf{W}^2 + \mathsf{b}^2)\mathsf{W}^3$$

Learnt feature representations

Inputs (very basic model)

- One-hot representation of words

the	1	0	0	0
movie	0	1	0	0
is	0	0	1	0
good	0	0	0	1

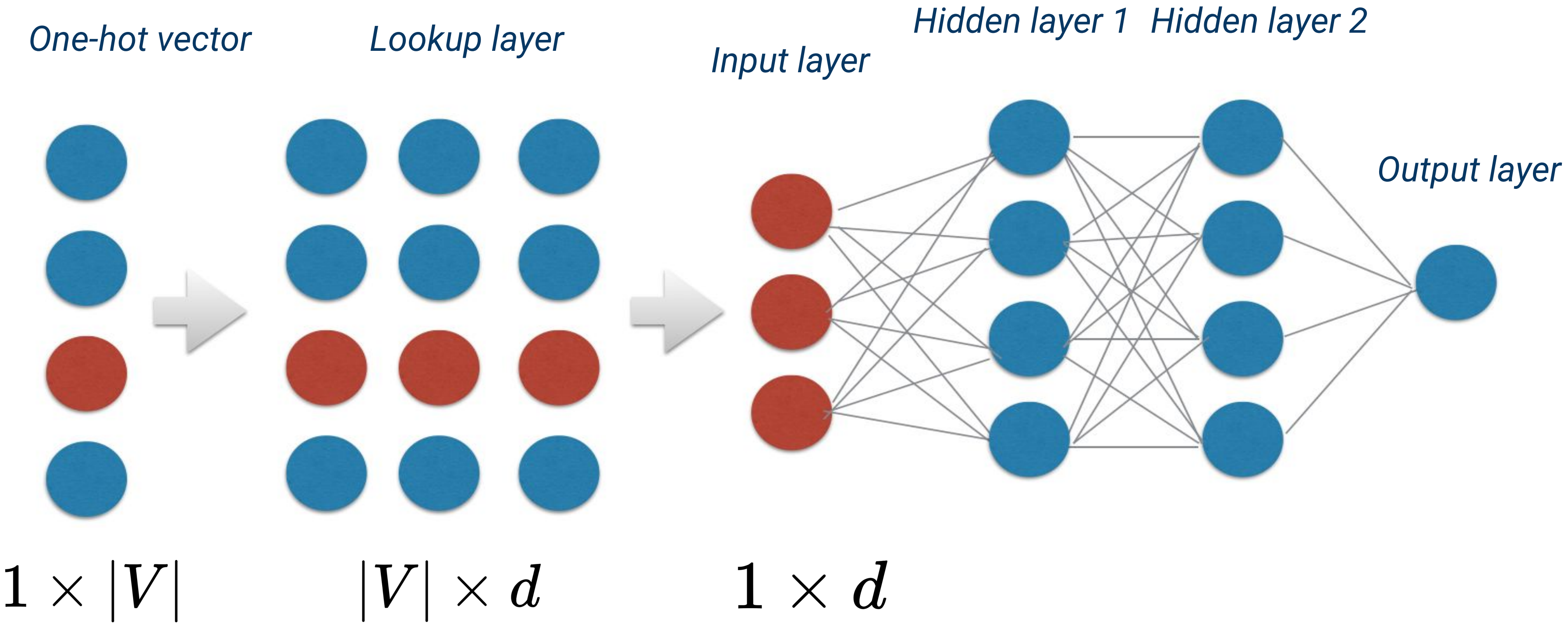
Inputs (better model)

- Automatically learnt dense feature representations, or
- Pre-trained dense representations

the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6

Neutral Networks

For a single word:



Document representation

How to get a document representation of sentence of fixed dimensionality?

Document 1: l = 4

the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6

Document 2: l = 2

excellent	0.4	0.2	-0.1
!	0.8	-0.5	0.4

Document representation

Document 1: l = 4

the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6
average	0.55	-0.175	0.25

Document 2: l = 2

excellent	0.4	0.2	-0.1
!	0.8	-0.5	0.4
average	0.6	-0.15	0.15

Document representation

Document 1: l = 4

the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6

Document 2: l = 2

excellent	0.4	0.2	-0.1
!	0.8	-0.5	0.4
-	0	0	0
-	0	0	0

- Could I do this?

Document representation

Document 1: l = 4

the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6

Document 2: l = 2

excellent	0.4	0.2	-0.1
!	0.8	-0.5	0.4
-	0	0	0
-	0	0	0

- This is a really bad idea:
  - Model architecture fixed to sentence length size
  - Model weights learnt for specific word positions

Why neural networks

- Automatically learned features
- Flexibility to fit highly complex relationships in data
  - **But**: they require more data to learn more complex patterns

Questions so far?

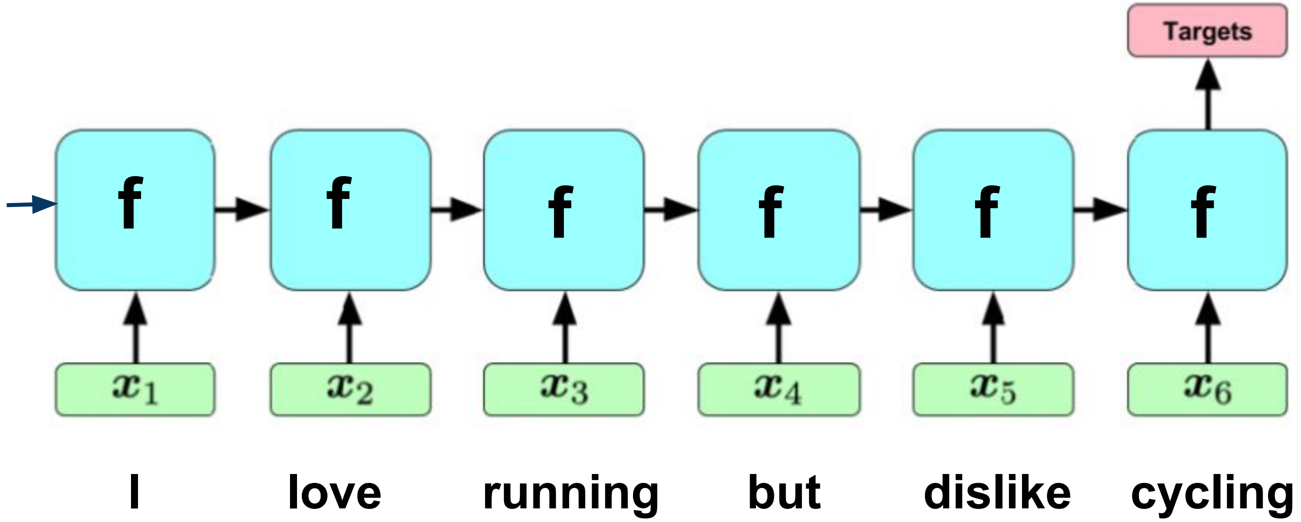
1hr 10 mins

Outline

1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. Neural Networks (NNs)
5. **Recurrent neural networks (RNNS)**
6. CNNs
7. Our recent research

Recurrent Neural Networks

- Natural language data - sequences
- Value of a unit depends on own previous outputs
- Usually the last hidden state is the input to the output layer



Vanilla RNNs

- RNN (f) computes its next state h_{t+1} based on:
  - **Hidden state vector** and **input vector** at time t

$$h_{t+1} = f(h_t, x_t) = \tanh\left(W h_t + U x_t\right)$$

- Its hidden state is carried along (memory)
- Main parameters, matrices W and U:

$W \in \mathbb{R}^{????}$ hidden-to-hidden

$U \in \mathbb{R}^{????}$ input-to-hidden

Vanilla RNNs

- RNN (f) computes its next state h_{t+1} based on:
  - **Hidden state vector** and **input vector** at time t

$$h_{t+1} = f(h_t, x_t) = \tanh\left(W h_t + U x_t\right)$$

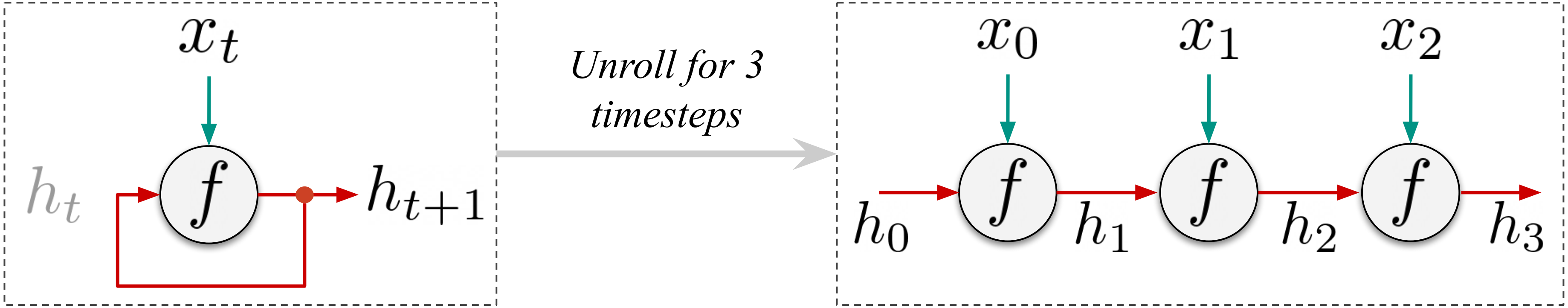
- Its hidden state is carried along (memory)
- Main parameters, matrices W and U:

$$W \in \mathbb{R}^{H \times H} \quad \text{hidden-to-hidden}$$

$$U \in \mathbb{R}^{E \times H} \quad \text{input-to-hidden}$$

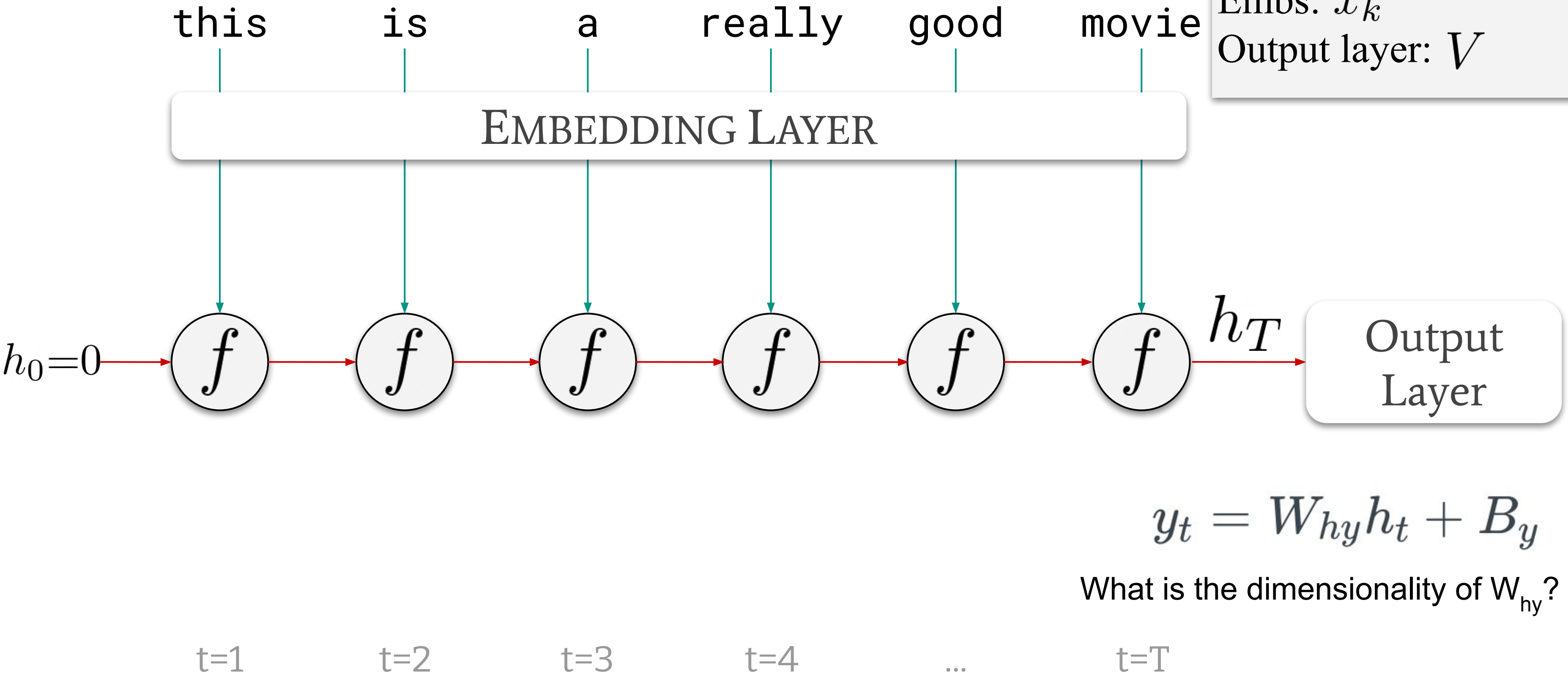
Vanilla RNNs

- Unrolling an RNN yields a deep feed-forward network
  - Easier to conceptualise



$$h_{t+1} = f(h_t, x_t) = \tanh(Wh_t + Ux_t)$$

Vanilla RNNs

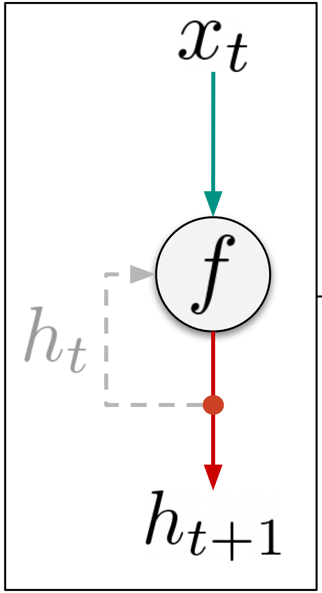


RNN params: W, U
Embs: x_k
Output layer: V

$$y_t = W_{hy} h_t + B_y$$

What is the dimensionality of W_{hy}?

Vanilla RNNs




```
size_H = 100, size_E = 20
E = torch.nn.Embedding(vocab_size, size_E)
U = torch.rand(size_H, size_E, requires_grad=True)
W = torch.rand(size_H, size_H, requires_grad=True)

sent = ["this", "is", "a", "really", "good", "movie"]

# Start as zero
h_t = torch.zeros(size_H, 1)
loss = 0

for i in range(len(sent) - 1):
  x_t = E(sent[i])
  h_t = torch.tanh(W.matmul(h_t) + U.matmul(x_t))
```

Limitations - the vanishing gradient problem

- **Vanishing gradient problem**
  - The model is less able to learn from earlier inputs:
    - Tanh derivatives are between 0 and 1
    - Sigmoid derivatives are between 0 and 0.25
  - Gradient for earlier layers involves repeated multiplication of the same matrix W
    - Depending on the dominant eigenvalue this can cause gradients to either ‘vanish’ or ‘explode’

Questions so far?

1hr 25 mins

Outline

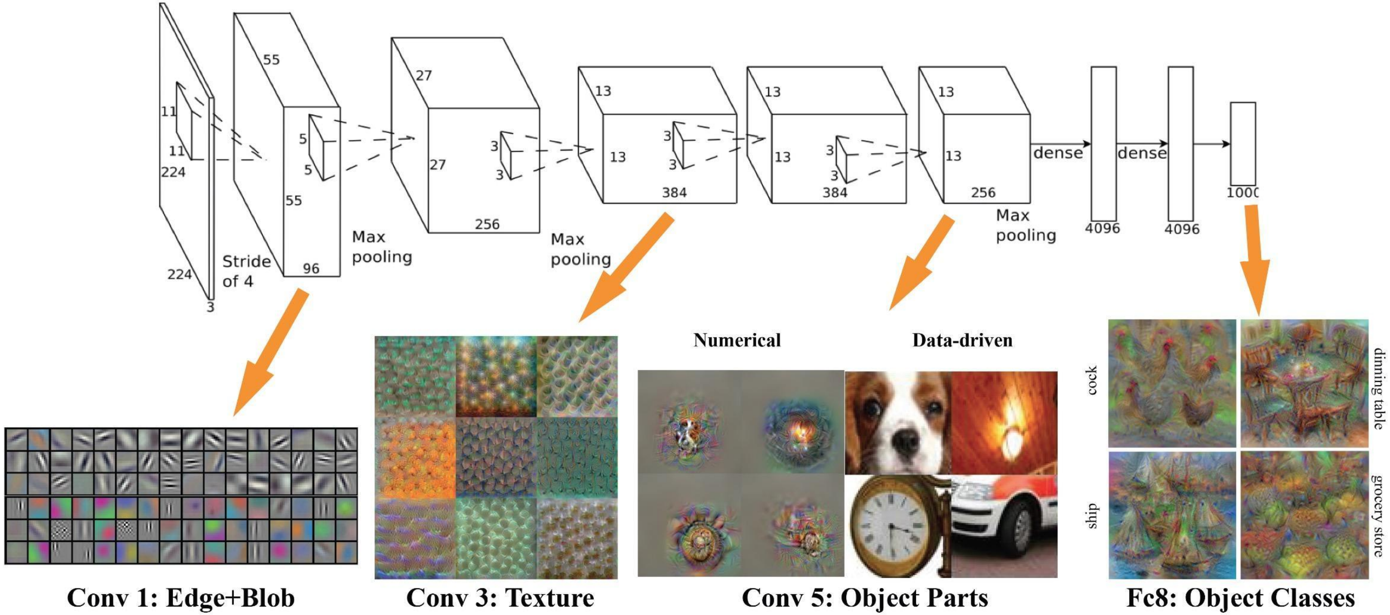
1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNS)
6. **CNNs**
7. Our recent research

Convolutional Neural Networks

- CNNs are composed of a series of **convolution** layers, **pooling** layers and **fully connected** layers
  - Convolution layers:
    - Detect important patterns in the inputs
  - Pooling layers:
    - Reduce dimensionality of features
    - Transform them into a fixed-size
  - Fully connected layers:
    - Train weights of learned representation for a specific task

Convolutional Neural Networks

- CNN learns values of its filters based on task. E.g. object classification:



https://www.cc.gatech.edu/~hays/compvision2017/proj6/deepNetVis.png

Convolutional Neural Networks

- Filter: sliding window over full rows (words) in one direction
  - **Filter width** = embedding dimension
  - **Filter height** = normally 2 to 5 (bigrams to 5-grams)

I	0.6	0.1	-0.2
really	0.2	0.1	0.5
like	-0.5	0.9	0.3
it	0.1	0.2	0.2
a	0.3	0.4	0.1
lot	0.5	0.7	0.1

Convolutional Neural Networks

2X3 filter

$\mathbf{X} \cdot \mathbf{W}$

0.6	0.4	-0.3
0.3	-0.7	0.5

window $n = 2$

I	0.6	0.1	-0.2
really	0.2	0.1	0.5
like	-0.5	0.9	0.3
it	0.1	0.2	0.2
a	0.3	0.4	0.1
lot	0.5	0.7	0.1

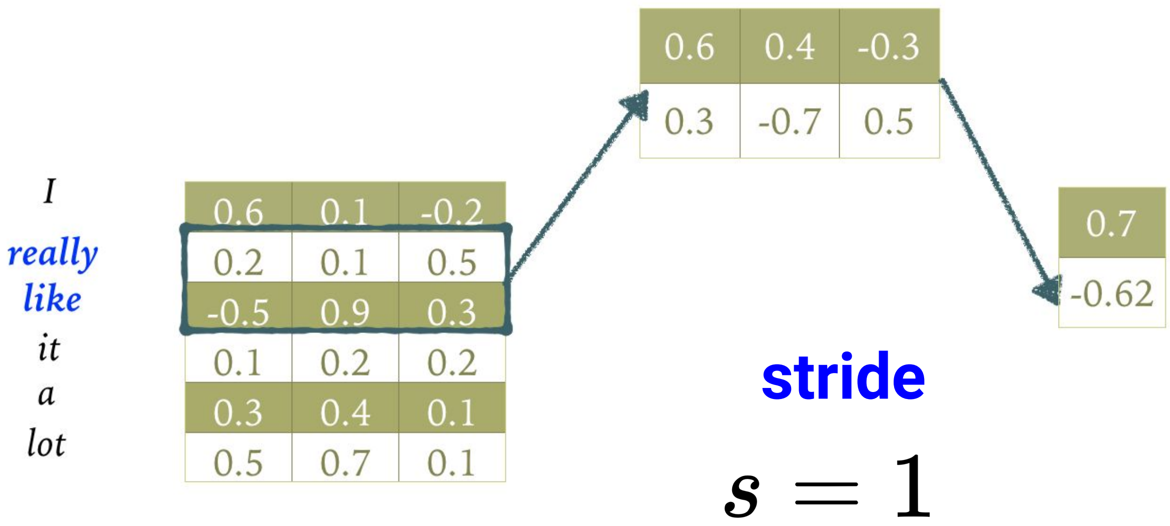
0.7

You also have padding and strides…

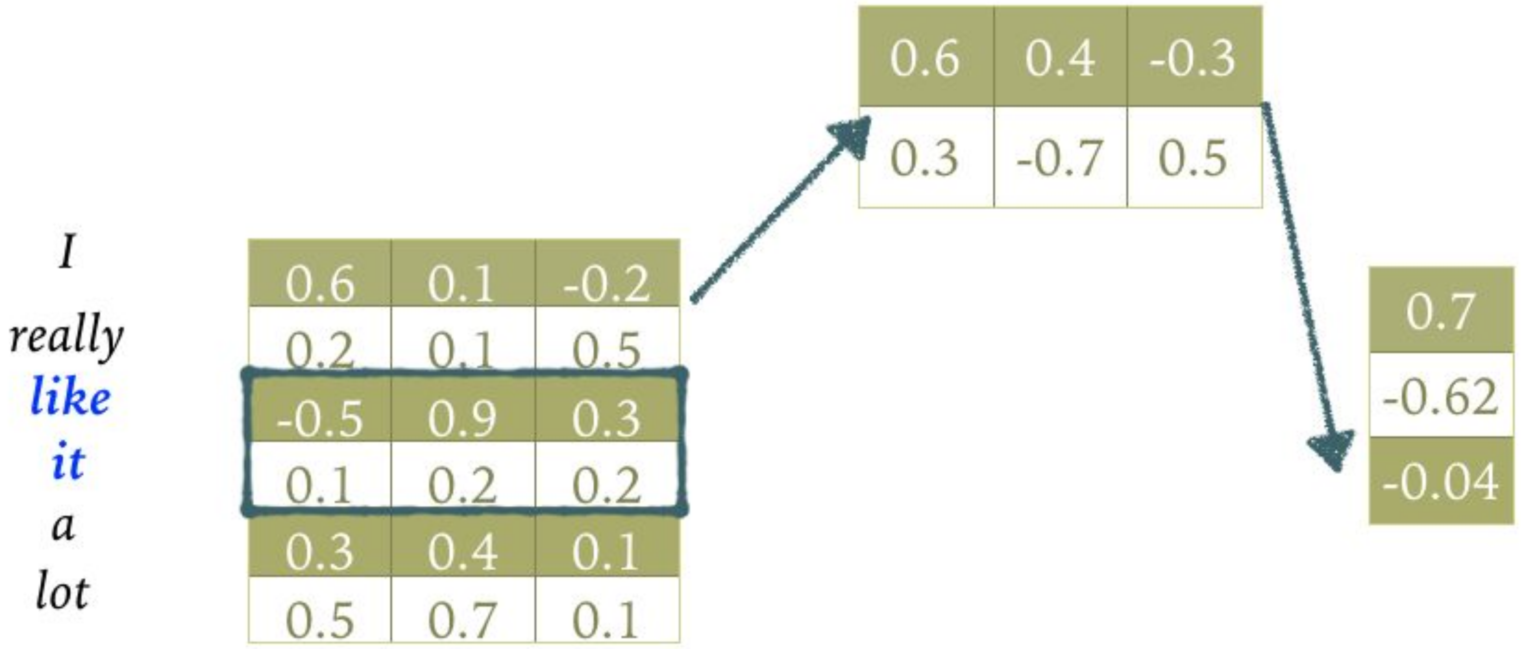
Convolutional Neural Networks

- **Stride size** - how much to shift the filter at each step

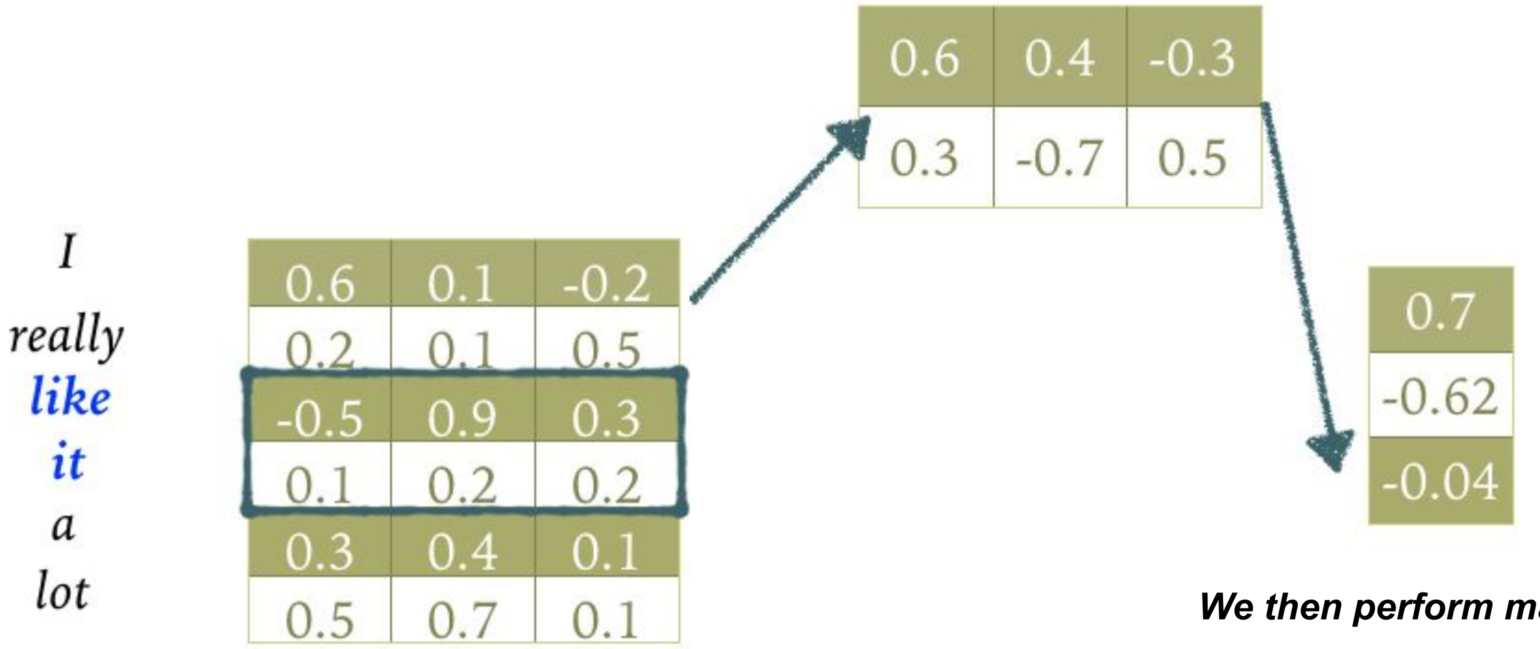
2X3 filter



Convolutional Neural Networks



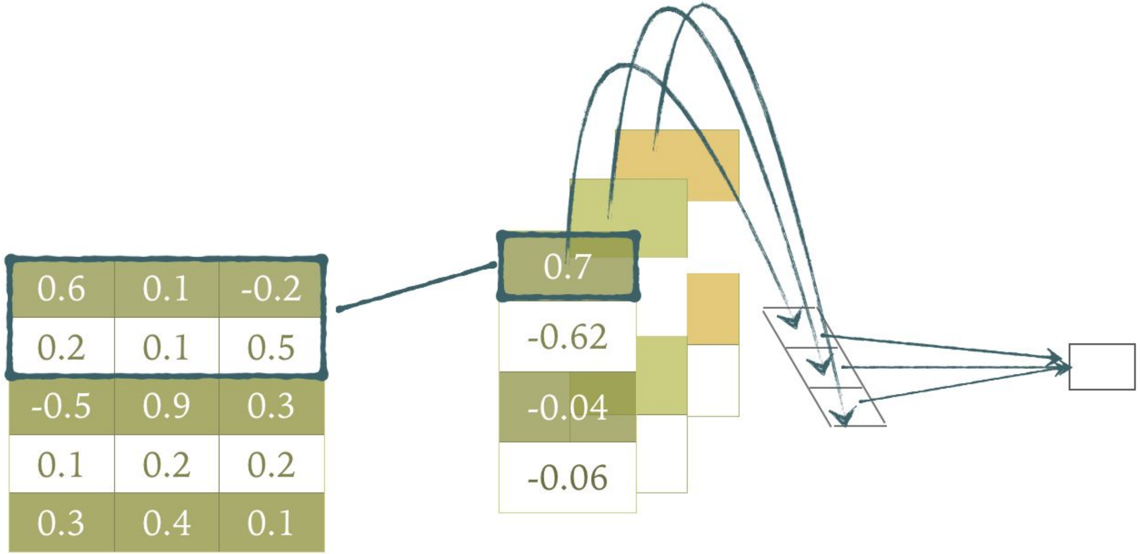
Convolutional Neural Networks



We then perform max pooling

Convolutional Neural Networks

- If we have d different parallel filters, then we have a d-dimensional representation



RNNs vs CNNs

- Understanding the strengths of **both types of models:**

- CNNs can perform well if the task involves key phrase recognition
- Whereas RNNs perform better when you need to understand longer range dependencies

Determining if an article is Fake News based on the headline:

- `“Hillary Clintons election fraud finally exposed. California stolen from Bernie Sanders!”`

- `Hillary Clinton Needed Someone to 'Sober Her Up' at 4:30 in the Afternoon`

My own work:

Logical reasoning in NLI
(a simplified introduction)

1hr 35 mins

My most recent work - combining logical reasoning with neural networks

Why logical reasoning? What are we trying to achieve?

We need to break down the premise and hypothesis into **logical atoms**.

This section ‘My own work’ is not examinable, so just for your interest

What do I mean by 'logical atoms'?

Building blocks of the task, that we apply logical rules to.

E.g. if you have a dataset with family relations:

Atom 1: Mikkel is the brother of Marta
Atom 2: Marta is the daughter of Ulrich

Q: What is the relationship between Mikkel and Ulrich?

This section 'My own work' is not examinable, so just for your interest

NLI example

What class is the example below?

- Entailment, Contradiction, or Neutral

Premise: the man in the black wetsuit is walking out of the water

Hypothesis: a man in a wetsuit walks out of the water carrying a surfboard

This section 'My own work' is not examinable, so just for your interest

What are our ‘logical atoms’?

What are our logical atoms?

Premise: the man in the black wetsuit is walking out of the water

Hypothesis: a man in a wetsuit walks out of the water carrying a surfboard

This section ‘My own work’ is not examinable, so just for your interest

What do I mean by ‘logical atoms’?

What are our logical atoms?

Premise: the man in the black wetsuit is walking out of the water

Hypothesis: a man in a wetsuit walks out of the water carrying a surfboard

This section ‘My own work’ is not examinable, so just for your interest

Why is this so cool?

Contradiction span *Neutral span*

Example 1 (in-distribution):

Premise: two women are embracing while holding to go packages.

Hypothesis: the sisters are hugging goodbye while holding to go packages after just eating lunch.

Example 2 (out-of-distribution):

Premise: your contribution helped make it possible for us to provide our students with a quality education.

Hypothesis: your contributions were of no help with our students' education.

"It's like giving our model explainability super powers…"

What's the key?

- Our model predicts a logit score for each atom
- We supervise the maximum score per atom
(e.g. a score for each span)
- 'Max' acts like a logical OR operator....

$$\mathcal{L}_n^{\text{Span}} = (\max_i(\widetilde{a}_{n,i}) - y_n)^2$$

This section 'My own work' is not examinable, so just for your interest

Our logical rules

Logical rules for training:

Sentence label:		Cont. spans:	Neutral spans:
Contradiction	=>	At least one	Unknown
Neutral	=>	None	At least one
Entailment	=>	None	None

Logical rules for evaluation:

Cont. spans:	Neutral spans:		Sentence label:
At least one		=>	Contradiction
None	At least one	=>	Neutral
None	None	=>	Entailment

This section 'My own work' is not examinable, so just for your interest

Another use-case

Grammatical error detection:

This is a very badly ritten sentence.

This section ‘My own work’ is not examinable, so just for your interest

Another use-case

Our logical model almost fully retains performance

Accuracy	SNLI	Δ
BERT (baseline)	90.77	
Feng et al. (2020)	81.2	*-9.57*
Wu et al. (2021)	84.53	*-6.24*
Feng et al. (2022)	87.8	*-2.97*
SLR-NLI	90.33	*-0.44*
SLR-NLI+esnli	**90.49**	***-0.28***

And performs very well on small datasets…

Dataset	Baseline	SLR-NLI
SICK	81.11	**81.33**
SNLI-dev	38.50	**46.96**‡
SNLI-test	38.17	**46.88**‡
SNLI-hard	38.34	**44.58**‡
MNLI-mismatch.	40.90	**47.85**†
MNLI-match.	39.72	**46.51**†
HANS	**53.22**	50.61

This section 'My own work' is not examinable, so just for your interest

Any questions?

If you're curious, or want to understand this work in more detail, you can read our paper.

Logical Reasoning with Span-Level Predictions for Interpretable and Robust NLI Models

Joe Stacey
Imperial College London
j.stacey20@imperial.ac.uk

Pasquale Minervini
University of Edinburgh & UCL
p.minervini@ed.ac.uk

Haim Dubossarsky
Queen Mary University of London
h.dubossarsky@qmul.ac.uk

Marek Rei
Imperial College London
marek.rei@imperial.ac.uk

This section 'My own work' is not examinable, so just for your interest

Questions so far?

1hr 45 mins

Coursework introduction

- The task is a binary classification task, where you will need to classify whether a text contains **condescending or patronising** language

Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities

Carla Pérez-Almendros **Luis Espinosa-Anke** **Steven Schockaert**
School of Computer Science and Informatics
Cardiff University, United Kingdom
{perezalmendrosc,espinosa-ankel,schockaerts1}@cardiff.ac.uk

Abstract

In this paper, we introduce a new annotated dataset which is aimed at supporting the development of NLP models to identify and categorize language that is patronizing or condescending towards

Coursework introduction

- The task is a binary classification task, where you will need to classify whether a text contains **condescending or patronising** language

- Extracts from a news corpus contains keywords relating to vulnerable communities (more information is found in the paper)

Don’t Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities

Carla Pérez-Almendros **Luis Espinosa-Anke** **Steven Schockaert**
School of Computer Science and Informatics
Cardiff University, United Kingdom
{perezalmendrosc,espinosa-ankel,schockaerts1}@cardiff.ac.uk

Abstract

In this paper, we introduce a new annotated dataset which is aimed at supporting the development of NLP models to identify and categorize language that is patronizing or condescending towards

Next time

- Detailed walkthrough of the coursework spec
- Using transformer models with HuggingFace
- Evaluating classification models
- De-biasing models

Questions so far?

1hr 50 mins

Debiasing

1hr 50 mins

Motivation

Self driving cars:

- Humans can learn to drive from just 80 hours

- However, machine learning models learn from millions of hours of driving data, but still make mistakes in new situations

How can we make AI (and NLP) models generalise better?

Natural Language Inference

- **Premise**: The kitten is climbing the curtains again
- **Hypothesis**: The kitten is sleeping

Labels:

- **Entailment**: if the hypothesis is implied by the premise
- **Contradiction**: if the hypothesis contradicts the premise
- **Neutral**: otherwise

Debiasing

Possible strategies:

- Augment with more data so that it ‘balances’ the bias
- Filter your data
- Prevent a classifier finding the bias in your model representations
- Make your model predictions different from predictions that only consider the bias

Debiasing

We combine together:

- The probabilities of each class given our bias (b_i)...
  - I will call these the “bias probabilities”

- … and the probabilities from our model (p_i)

$$\hat{p}_i = softmax(\log(p_i) + \log(b_i))$$

Debiasing

We combine together:

- The probabilities of each class given our bias (b_i)...

- … and the probabilities from our model (p_i)

$$\hat{p}_i = softmax(\log(p_i) + \log(b_i))$$

```
class BiasProduct(ClfDebiasLossFunction):
  def forward(self, hidden, logits, bias, labels):
    logits = logits.float()  # In case we were in fp16 mode
    logits = F.log_softmax(logits, 1)
    return F.cross_entropy(logits+bias.float(), labels)
```

Code from Clark et al (2019):
“Don’t Take the Easy Way Out Ensemble Based Methods for Avoiding Known Dataset Biases”

Debiasing

How can you get the ‘bias probabilities’ b_i:

- Target a specific known ‘biased’ feature

Or creating a ‘biased’ model…

Debiasing

Creating a ‘biased’ model:

1. Use a model not powerful enough for the task (e.g. BoW model)
2. Use incomplete information (e.g. only the hypothesis in NLI)
3. Use a shallow model (e.g. TinyBERT)
4. Train a classifier on a very small number of observations

Debiasing

Another approach is to weight the loss of examples based on the performance of our biased model:

- When training the current model, multiply the loss by:
  $1 - b_i$

- Where b_i is the probability that the bias model predicts the correct class

Clark et al (2019):
"Don't Take the Easy Way Out Ensemble Based Methods for Avoiding Known Dataset Biases"
In this implementation, the loss is normalized across each minibatch so the total minibatch loss remains the same

Debiasing

When is it desirable to stop a model learning from shallow-heuristics in the dataset?

Debiasing

- You will find results something like

	In-distribution test set	**out-of-distribution test set**
Normal training	Model performs great	Not so great
Training with PoE	Little bit worse than normal training	Better than normal training

Questions so far?

Coursework introduction

- The task is a binary classification task, where you will need to classify whether a text contains **condescending or patronising** language
- Extracts from a news corpus contains keywords relating to vulnerable communities (more information is found in the paper)

Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities

Carla Pérez-Almendros **Luis Espinosa-Anke** **Steven Schockaert**
School of Computer Science and Informatics
Cardiff University, United Kingdom
{perezalmendrosc,espinosa-ankel,schockaerts1}@cardiff.ac.uk

Abstract

In this paper, we introduce a new annotated dataset which is aimed at supporting the development of NLP models to identify and categorize language that is patronizing or condescending towards