Statistical Natural Language Processing
Text Classification

Çağrı Çöltekin

University of Tübingen
Seminar für Sprachwissenschaft

Summer Semester 2019

Some examples

is it spam?

```
From: Dr Pius Ayim <mikeabass15@gmail.com>
Subject: Dear Friend / Lets work together

Dear Friend,

My name is Dr. Pius Anyim, former senate president of the Republic Nigeria
under regime of Jonathan Good-luck.

I am sorry to invade your privacy; but the ongoing ANTI-CORRUPTION GRAFT
agenda of the rulling government is a BIG problem that I had to get your
contact via a generic search on internet as a result of looking for a
reliable person that will help me to retrieve funds I deposited at a
financial institute in Europe.

…
```

* From my 'spambox' which I stopped checking regularly long time ago.

Some examples

is the customer happy?

> I never understood what's the BIG deal behind this album. Yes,
> the production is wonderfull but the songwriting is childish and
> rubbish. They definitly can not write great lyrics like Bob
> Dylan sometimes do. "God Only Know" and "Wouldnt Be nice" are
> indeed masterpieces...but the rest of the album is background
> music.

Some examples

is the customer happy?

> I never understood what's the BIG deal behind this album. Yes, the production is wonderfull but the songwriting is childish and rubbish. They definitly can not write great lyrics like Bob Dylan sometimes do. "God Only Know" and "Wouldnt Be nice" are indeed masterpieces...but the rest of the album is background music.

> @DB_Bahn mußten sie für den Sauna-Besuch zuzahlen ?

Some examples
is the customer happy?

> I never understood what's the BIG deal behind this album. Yes,
> the production is wonderfull but the songwriting is childish and
> rubbish. They definitly can not write great lyrics like Bob
> Dylan sometimes do. "God Only Know" and "Wouldnt Be nice" are
> indeed masterpieces...but the rest of the album is background
> music.

> @DB_Bahn mußten sie für den Sauna-Besuch zuzahlen ?

- Sentiment analysis is one of the popular applications of text classification

Some examples
which language is this text in?

Član 3. Svako ima pravo na život, slobodu i ličnu bezbjednost.

Some examples
which language is this text in?

Član 3. Svako ima pravo na život, slobodu i ličnu bezbjednost.

- Detecting language of the text is often the first step for many NLP applications.
- Easy for the most part, but tricky for
  - closely related languages
  - text with code-switching

More questions

- Who wrote the book?
- Find the author's
  - age
  - gender
  - political party affiliation
  - native language
- Is the author/speaker depressed?
- What is the proficiency level of a language learner?
- What grade should a student essay get?
- What is the diagnosis, given a doctor's report?

- What category should a product be listed based on its description?
- What is the genre of the book?
- Which department should answer the support email?
- Is this news about
  - politics
  - sports
  - travel
  - economy
- Is the web site an institutional or personal web page?

Text classification

- In many NLP applications we need to classify text documents
- Documents of interest vary from short messages to complete books
- The classification task can be binary or multi-class
- The core part of the solution is a classifier
- The way to extract features from the documents is important (and interacts with the classification method)
- Many of the methods apply to 'text regression' as well

Text classification

the definition

- Given a document, our aim is to classify it into one (or more) of the known classes
- During prediction

  input a document

  otput the predicted document class
- During training

 input a set of documents with associated labels

  otput a classifier

Essentially, the task is supervised learning (classification).
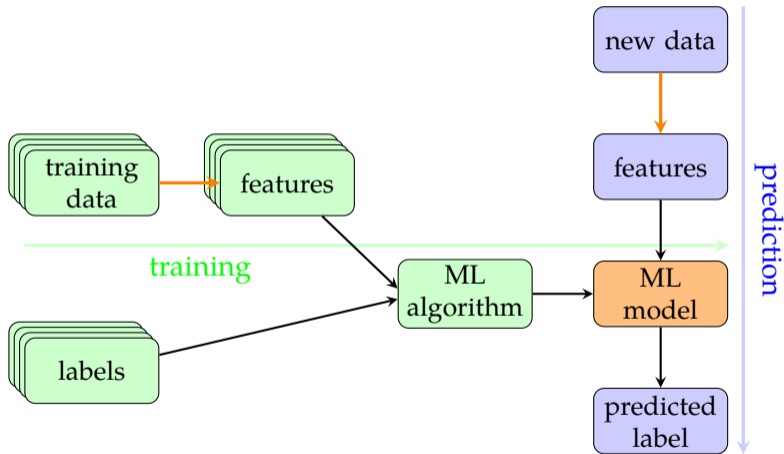
How about a rule-based method?

- They exist, and still used often in the industry
- Rule-based approaches are language specific
- It is difficult to adapt them to new environments

We will stick to statistical / machine learning approaches

Supervised learning
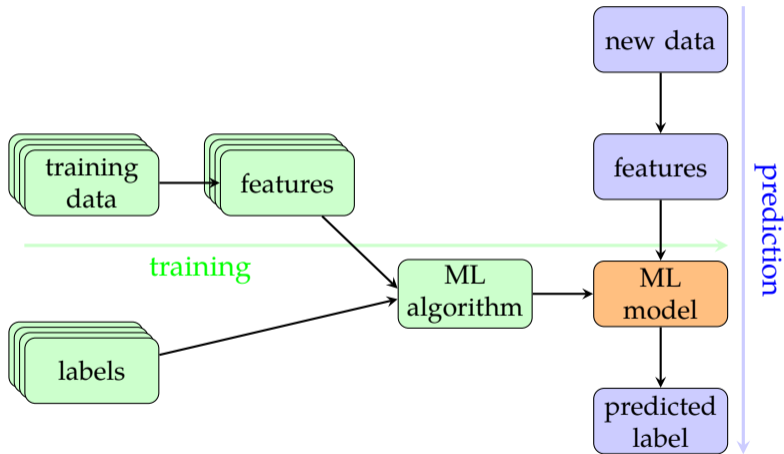
Supervised learning

Two important parts

- How do we represent a document?
    - what features to use?
        - words? characters? both?
        - n-grams of words or characters?
        - some simple measures of text, like document lenght or TTR?
        - linguistic features: POS tags, dependencies?
    - what value to assign to each feature?
- What classification algorithm should we use?

Bag of words (BoW) representation

The idea: use words that occur in text as features without paying attention to their order.

The document

It's a good thing most animated sci-fi movies come from Japan, because "titan a.e." is proof that Hollywood doesn't have a clue how to do it. I don't know what this film is supposed to be about.

BoW representation

how japan good n't I thing film what , proof titan a because . 's know does most hollywood is animated it do sci-fi a.e. " " supposed be come clue to this that from have movies about .

Bag of words representation
with binary features

The document

It's a good thing most animated sci-fi movies come from Japan, because "titan a.e." is proof that Hollywood doesn't have a clue how to do it. I don't know what this film is supposed to be about.

- If the word is in the document, the value of 1, otherwise 0
- The feature vector contains values for all words in our document collection

feature	value
to	1
do	1
a	1
thing	1
have	1
good	1
be	1
clue	1
great	0
pathetic	0
masterpiece	0
…	

Bag of words representation
with (document) frequencies

The document

It's a good thing most animated sci-fi movies come from Japan, because "titan a.e." is proof that Hollywood doesn't have a clue how to do it. I don't know what this film is supposed to be about.

feature	value
to	2
do	2
a	2
thing	1
have	1
good	1
be	1
clue	1
great	0
pathetic	0
masterpiece	0
…	

- Use frequencies rather than binary vectors
- May help in some cases, but
  - effect of document length
  - frequent is not always good

Bag of words representation
with relative frequencies

The document

It's a good thing most animated sci-fi movies come from Japan, because "titan a.e." is proof that Hollywood doesn't have a clue how to do it. I don't know what this film is supposed to be about.

- Relative frequencies are less sensitive to document length
- Still, high-frequency words dominate

feature	value
to	0.06
do	0.06
a	0.06
thing	0.03
have	0.03
good	0.03
be	0.03
clue	0.03
great	0.00
pathetic	0.00
masterpiece	0.00
…	

tf-idf weighting

- Intuition:
  - Words that appear multiple times in a document is important/representative for the document
  - Words that appear in many documents are not specific
- tf-idf uses two components
  - tf term frequency - frequency of the word in the document
  - idf inverse document frequency - inverse of the ratio of documents that contain the term
- Both components are typically normalized

$$\text{tf-idf}_{t,d} = \overbrace{\frac{\overbrace{C_{t,d}}^{\text{term count in doc}}}{\underbrace{|d|}_{\text{doc length}}}} \times \log \frac{\overbrace{N}^{\text{number of docs}}}{\underbrace{n_t}_{\text{number of docs with t}}}$$

tf-idf example

Document 1 (d_1)		Document 2 (d_2)		Document 3 (d_3)	
the	5	the	2	the	1
good	2	a	2	a	2
bad	1	book	1	good	3

$$\text{tf-idf}(t, d) = \text{tf}(t, d) \times \text{idf}(t)$$

$\text{tf-idf}(\text{good}, d_1) = ?$ $\qquad\qquad$ $\text{tf-idf}(\text{the}, d_1) = ?$

$\text{tf-idf}(\text{bad}, d_1) = ?$ $\qquad\qquad$ $\text{tf-idf}(\text{good}, d_3) = ?$

tf-idf example

Document 1 (d_1)		Document 2 (d_2)		Document 3 (d_3)	
the	5	the	2	the	1
good	2	a	2	a	2
bad	1	book	1	good	3

$$\text{tf-idf}(t, d) = \text{tf}(t, d) \times \text{idf}(t)$$

$$\text{tf-idf}(\text{good}, d_1) = \frac{2}{8} \times \log\frac{3}{2} = 0.15 \qquad \text{tf-idf}(\text{the}, d_1) = \frac{5}{8} \times \log\frac{3}{3} = 0.00$$

$$\text{tf-idf}(\text{bad}, d_1) = \frac{1}{8} \times \log\frac{3}{1} = 0.20 \qquad \text{tf-idf}(\text{good}, d_3) = \frac{3}{6} \times \log\frac{3}{2} = 0.29$$

Some notes on tf-idf

- tf-idf is a very effective method for term weighting
- In practice it is common to work with non-relative
- It was originally used for information retrieval, where it brought substantial improvements over other methods
- It is also very effective on text classification when using linear models
- There are some alternatives (e.g., BM25), and many variations frequencies for TF, or take the logarithm
- But it has been difficult to improve over it (since 1970's)

A document is more than a BoW

> **The example document for sentiment analysis**
>
> It's a good thing most animated sci-fi movies come from Japan, because "titan a.e." is proof that Hollywood doesn't have a clue how to do it. I don't know what this film is supposed to be about.

- So far, we considered documents as simple BoW words
- BoW representations is surprisingly successful in many fields (IR, spam detection, …)
- However, word order matters
  - According to a sentiment dictionary, our example contains one positive and one negative word

A document is more than a BoW

> **The example document for sentiment analysis**
>
> It's a good thing most animated sci-fi movies come from Japan, because "titan a.e." is proof that Hollywood doesn't have a clue how to do it. I don't know what this film is supposed to be about.

- So far, we considered documents as simple BoW words
- BoW representations is surprisingly successful in many fields (IR, spam detection, …)
- However, word order matters
    - According to a sentiment dictionary, our example contains one positive and one negative word
- Paying attention to longer sequences allows us to get better results

Bag of n-grams

- Using n-grams rather than words allows us to capture more information in the data
- We can still use the same weighting methods (tf-idf)
- It is a common practice to use a range of n-grams
- This results in large set of features (millions for most practical applications)
- Data sparsity is a problem for higher order n-grams

The unreasonable effectiveness of character n-grams

feature	value
it	2
t'	1
's	2
s␣	3
␣a	4
a␣	5
␣g	2
it's␣	2
t's␣a	1
's␣a␣	1
s␣a␣g	1
␣a␣go	2
a␣goo	2
…	

The example document

It's a good thing …

- For a number of text classification tasks (authorship attribution, language detection), character n-gram features found to be effective
- The idea is to use a range of character n-grams
- Works for both linear models and ANN (convolution, RNN) models

What about the linguistic features?

- Linguistic features such as
  - lemmas
  - sequences of POS tags
  - parser output: dependency triplets, or partial trees

  are also used in some tasks
- It is often difficult to get improvements over simple features
- It also makes systems more complex and language dependent
- Linguistic features can particularly be useful if the amount of data is limited
- They are also interesting if the aim is finding linguistic explanations (rather than solving an engineering problem)

Preprocessing

- Some preprocessing steps are common in some tasks
- Preprocessing include
    - removing formatting (e.g., HTML tags)
    - tokenization
    - case normalization
    - spelling correction
    - replacing numbers with a special symbol
    - removing punctuation
- Depending on the task, preprocessing can hurt!

Feature selection

- Feature selection is a common step for
  - reducing the size of the feature vectors
  - reducing noise
- Depending on the task, 'stopwords' can also be removed
- One option is dimensionality reduction (e.g., PCA)
- Another solution is to use a 'feature weighting' method, common methods include
  - Frequency
  - Information Gain
  - χ^2 (chi-squared)

Choice of (linear) classifiers

- Once we have our feature vectors, we can use (almost) any classifier
- Many methods has been used in practice
  - Naive Bayes
  - Decision trees
  - kNN
  - Logistic regression
  - Support vector machines (SVM)
  - Neural networks
- SVMs often perform better than others, but choice is task dependent

Ensemble classifiers

- When we have multiple sets of features, we can either
  - Concatenate the feature vectors, and train a single classifier
  - Train multiple classifiers with each feature set and combine their results
- Ensemble methods combine multiple classifiers
  - we can train a separate classifier for each feature set, and a higher-level classifier to make the final decision
  - use output of one (or more) classifiers as an input to another classifier
- In a number of tasks, ensemble models are reported to perform better

Neural networks for text classification

- As powerful non-linear classifiers, ANNs are also useful in text classification tasks
- Some of the tricks used for linear classifiers are not necessary for ANNs
    - We do not need to include n-gram features: ANNs are able to combine the effects of individual words
    - ANNs (ideally) can also learn importance of the features
- Even with single-word features, however, the BoW representations are too large for (current) ANNs
- Common methods for text classification with ANNs include convolutional networks and recurrent networks

Embeddings

- For text classification with ANNs, we use the term vectors (instead of document vectors)
- Embeddings help reducing dimensionality
- We often want to use 'task specific' embeddings
    - The first layer of the network (conceptually) learns a mapping from one-hot representations to dense representations
    - Task-specific embeddings represent information important for solving the task
    - Initializing embeddings with 'general purpose' embeddings is useful in some tasks

Bag of embeddings

- Embeddings are typically used as a first step for convenient learning with other ANN methods (e.g., CNNs or RNNs)
- A simple (and often surprisingly effective) method is
  - use a function of embeddings (e.g., average) as the document vector
  - use a (multi-layer) classifier on the document vectors
- This is similar to BoW approach, except dense representations are used
- Simple, but sometimes more effective than more elaborate methods

Convolutional networks

Classifier

Features

Pooling

Feature maps

Convolution

Word vectors

Input

not   really   worth   seeing
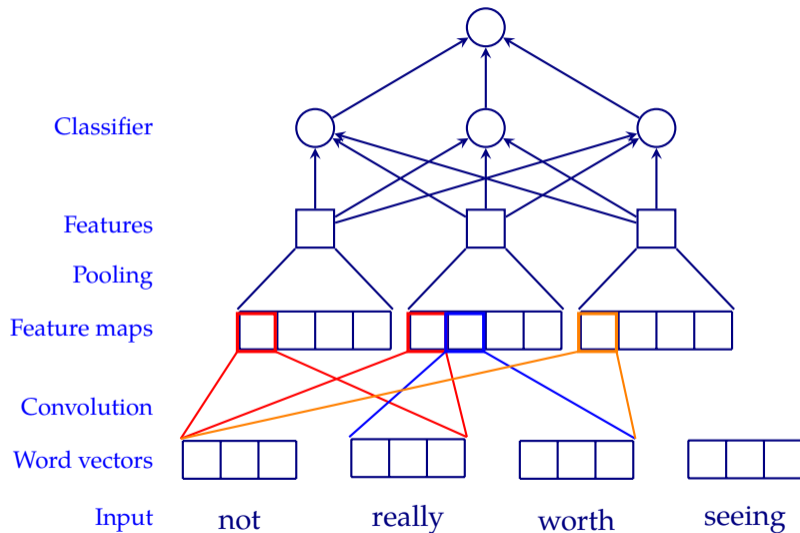
# Convolutional networks for text classification

- CNNs are used a number of text classification tasks successfully
- Convolutions learn feature maps from n-grams
- It is common to use both word- and character-based CNNs
- Pooling is often performed over the whole document (generally, we are not interested in non-local dependencies)

Recurrent networks
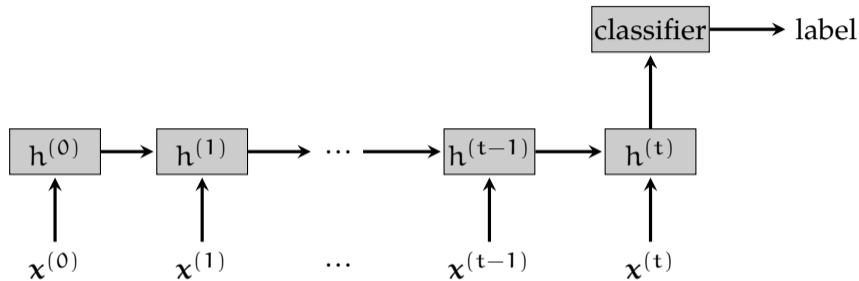
• Typically, we get a 'document representation' from the final representation

Recurrent networks
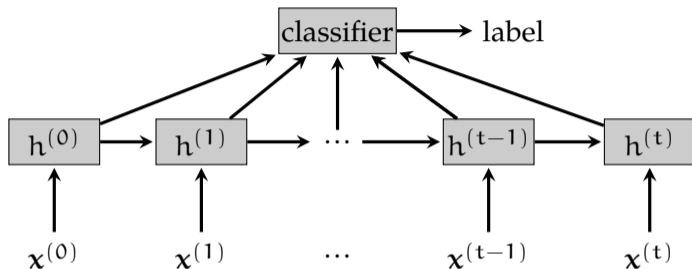
- Typically, we get a 'document representation' from the final representation
- Another common alternative is combining representations from all steps
- The use of both word and character inputs is common
- Bidirectional RNNs are often useful

Some final remarks

- Both linear classifiers and ANNs are currently used successfully in text classification
- Performance differ based on task and amount of data
- Some variations include
  - Assigning documents to multiple classes: multi-label classification
  - Hierarchical classification: e.g., category of a web page in a web directory like Yahoo!
- In case there is no labeled data, clustering is another option

Summary

- There are numerous applications of text classification in NLP
- Both linear classifiers and ANNs are used
- For linear classifiers term weighting is important
- For ANNs embeddings are crucial

Summary

- There are numerous applications of text classification in NLP
- Both linear classifiers and ANNs are used
- For linear classifiers term weighting is important
- For ANNs embeddings are crucial

Next:

Fri Parsing

Mon Parsing

Fri An overview of NLP applications (not covered in the class)