Chapter NLP:III

III. Text Models

Text Classification

Text Classification Problems

Sentiment: positive or negative?

There really needs nothing to be said about how good this is for a 1970s movie, especially in terms of technical aspects. The Academy certainly got it right by giving them all these Oscars for it.

Text Classification

Text Classification Problems

Sentiment: positive or negative?

There really needs nothing to be said about how good this is for a 1970s movie, especially in terms of technical aspects. The Academy certainly got it right by giving them all these Oscars for it.

The script is not the greatest achievement in general. It's the usual black-and-white characterization with good vs evil and the characters have no real shades, no facets.

Text Classification

Text Classification Problems

Sentiment: positive or negative?

There really needs nothing to be said about how good this is for a 1970s movie, especially in terms of technical aspects. The Academy certainly got it right by giving them all these Oscars for it. However, it lacks in certain other areas unfortunately. First of all, the acting isn't too great. That is not really a problem of the characters though, but rather of the way they were written, which simply offered no room for outstanding performances. This film is all about the way the characters look and not what they do or say. A bit style over substance.

The script is not the greatest achievement in general. It's the usual black-and-white characterization with good vs evil and the characters have no real shades, no facets. This was made up again by the strong work in other departments as it's not too memorable what they say and do, but what they look like. And while I enjoyed most of the film, I cannot see it as an epic or a cult classic as so many people do to this date.

Text Classification

Text Classification Problems

Sentiment: Number of stars between 1 and 10?

There really needs nothing to be said about how good this is for a 1970s movie, especially in terms of technical aspects. The Academy certainly got it right by giving them all these Oscars for it. However, it lacks in certain other areas unfortunately. First of all, the acting isn't too great. That is not really a problem of the characters though, but rather of the way they were written, which simply offered no room for outstanding performances. This film is all about the way the characters look and not what they do or say. A bit style over substance.

The script is not the greatest achievement in general. It's the usual black-and-white characterization with good vs evil and the characters have no real shades, no facets. This was made up again by the strong work in other departments as it's not too memorable what they say and do, but what they look like. And while I enjoyed most of the film, I cannot see it as an epic or a cult classic as so many people do to this date.

How can a program make this decision?

Text Classification

Text Classification Problems

We use classification to decide about span boundaries, span types, text labels, relations between spans, relation types, . . .

Lexical and syntactic

- Tokenization
- Sentence splitting
- Paragraph detection
- Stemming
- Lemmatization
- Part-of-speech tagging
- Similarity computation
- Spelling correction
- Phrase chunking
- Dependency parsing
- Constituency parsing

. . . and some more

Semantic and pragmatic

- Term extraction
- Numerical entity recognition
- Named entity recognition
- Reference resolution
- Entity relation extraction
- Temporal relation extraction
- Topic detection
- Authorship attribution
- Sentiment analysis
- Discourse parsing
- Spam detection
- Argument mining

. . . and many many more

Text Classification

Text Classification Problems

We use classification to decide about span boundaries, span types, text labels, relations between spans, relation types, . . .

Lexical and syntactic

- Tokenization $\rightarrow$ NLP-III
- Sentence splitting
- Paragraph detection
- Stemming
- Lemmatization $\rightarrow$ NLP-IV
- Part-of-speech tagging $\rightarrow$ NLP-IV
- Similarity computation
- Spelling correction
- Phrase chunking
- Dependency parsing $\rightarrow$ NLP-V
- Constituency parsing $\rightarrow$ NLP-V

. . . and some more

Semantic and pragmatic

- Term extraction
- Numerical entity recognition
- Named entity recognition $\rightarrow$ NLP-IV
- Reference resolution $\rightarrow$ NLP-VII
- Entity relation extraction
- Temporal relation extraction
- Topic detection
- Authorship attribution $\rightarrow$ Lab Class
- Sentiment analysis $\rightarrow$ here
- Discourse parsing $\rightarrow$ NLP-VII (maybe)
- Spam detection
- Argument mining $\rightarrow$ NLP-VII (maybe)

. . . and many many more

Text Classification

Classification Tasks [ML:I 33 ff.]

Definition 1 (Classification Task)

Given some $o \in O$, determine its class $\gamma(o) \in C$.

Setting of the real world:

- O is a set of objects.
  emails, IMDB movie reviews, document pairs

- C is a set of classes.
  spam, positive sentiment, same author

- $\gamma : O \to C$ is the ideal classifier. [NLP:II 92 ff.]
  Semantics: $\gamma(x)$ is a human expert that annotates the objects with the correct classe.

$$\begin{tikzcd}\text{Objects} & \text{Classes} \\ O \arrow[r, "\gamma(o)"] & C\end{tikzcd}$$

Text Classification

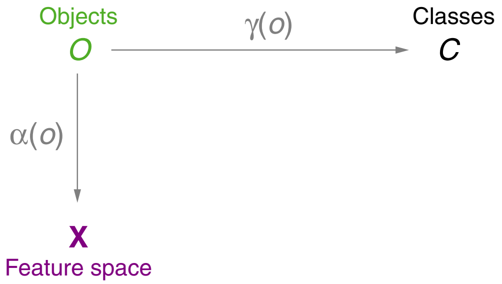
Classification Tasks [ML:I 33 ff.]

Definition 2 (Classification Task)

Given some $o \in O$, determine its class $\gamma(o) \in C$.

Setting of the model world:

- X is a multiset of feature vectors.
  word frequency vectors for all examples in O
- C is a set of classes.
  as before: spam, ...
- $\alpha : O \to X$ is the model formation function.
  Semantics: $\alpha(o)$ finds a feature vector that 'represents' the object.



$$\begin{tikzcd} O \arrow[r, "\gamma(o)"] \arrow[d, "\alpha(o)"'] & C \\ \mathbf{X} & \end{tikzcd}$$

Text Classification

Classification Tasks [ML:I 33 ff.]

Definition 3 (Classification Task)

Given some $o \in O$, determine its class $\gamma(o) \in C$.

NLP Course:

- Acquire O and C from language sources.
- Specify γ. guidelines, annotation studies, . . .
- Define a good $\mathbf{X}$ (and $\alpha(o)$).

$$\begin{tikzcd}
\substack{\text{Objects} \\ O} \arrow[r, "\gamma(o)"] \arrow[d, "\alpha(o)"'] & \substack{\text{Classes} \\ C} \\
\substack{\mathbf{X} \\ \text{Feature space}} \arrow[ur, "y(\mathbf{x})"'] &
\end{tikzcd}$$

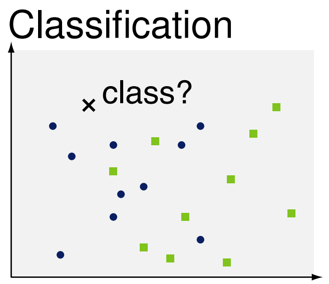
Machine Learning Course:

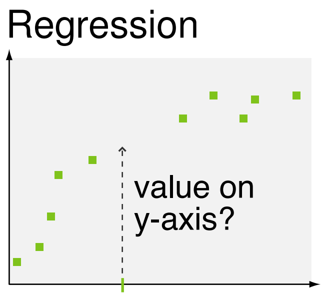
- Formulate a model function $y : \mathbf{X} \to C$
- Maximize the goodness of fit between $(\mathbf{x}, c)$ and $(\mathbf{x}, y(\mathbf{x}))$, $(\mathbf{x}, c) \in D$. D is a multiset of examples.

Remarks:

Classification and Regression in NLP

- In Regression, the task is to assign each example $\mathbf{x}$ a value from a real-valued scale.
- Classification predicts group membership, regression a "missing" value. Classification can be seen as a special case of regression with a threshold. [ML:II 57 ff.]
- There is a duality between the two formulations. While the classification output is determinate, the regression output can be interpreted as a 'intensity' or 'confidence score', e.g. *How positive is the sentiment* or *How certain is the authorship decision?*.





Text Classification

Classification Tasks: Classes C

Binary classification.

- There are exactly 2 classes, 1 has to be selected per example.
- Spam classification. spam/no spam
- Sentiment classification. positive/negative

Multi-class classification.

- There are c classes, 1 has to be selected per example.
- Amazon rating prediction. 1, 2, . . . , 5 stars

Multi-label classification.

- There are c classes, 1, 2, . . . , c have to be selected per example.
- Reuters news topics. trade, grain, ship, . . .
- Book genre classification. drama, comedy, romance, . . .

Text Classification

Classification Tasks: Objects O

Token classification. [NLP:IV 57 ff., NLP:V 85 ff.]

- ❑ Assign a class to each token in a sequence.
- ❑ POS tagging

  `The`$_{\texttt{determiner}}$ `dwarfs`$_{\texttt{noun}}$ `loved`$_{\texttt{verb}}$ `her`$_{\texttt{pronoun}}$ `dearly`$_{\texttt{adverb}}$

Document classification.

- ❑ Assign a class to a long, continuous sequence of tokens.
- ❑ Positional information (order and relation of words) is less important.

Span or sentence classification.

- ❑ Assign a class to a continuous sequence of tokens.
- ❑ Positional information is very important.
- ❑ Natural language understanding [GLUE, Wang et al., 2019]

  Is the Premise entailed in/contradicted by/neutral to the Hypothesis?

```
Premise:  I have never seen a hummingbird not flying.
Hypothesis:  I have never seen a hummingbird.
```

Remarks:

- ❑ Many (non-neural) classification algorithms work for $|C| = 2$ classes only. Multi-class and multi-label classification is handled with multiple binary classifiers (e.g., one-versus-all).
- ❑ Neural networks can learn multi-class classification natively. The number of classes can be controlled though the size of the output vector.

Text Classification

Feature space $\mathbf{X}$

In classification, each example o_j is represented as a feature vector $\mathbf{x} \in \mathbf{X}$.

- ❑ A feature vector is an ordered set of values of the form $\mathbf{x} = (x_1, \ldots, x_m)$, $m \geq 1$, where each feature x_i denotes a measurable property of an input.
  We consider only real-valued features here.
- ❑ Each instance o_j is mapped to a vector $\mathbf{x}^{(j)} = (x_1^{(j)}, \ldots, x_m^{(j)})$ where $x_i^{(j)}$ denotes the value of feature x_i for instance o_j.

$\rightarrow$ The model formation function $\alpha(o) = \mathbf{x}$ determines the representation fidelity, exactness, quality, or simplification. Finding a good α is essential.

Common strategies to build feature spaces:

1. Feature Engineering: the elements/dimension of the feature space $\mathbf{X}$ are selected and evaluated manually; each dimension in the feature vector has a pre-defined meaning. This is common for linear models, decision trees, bayesian learning, . . .
2. Representation Learning: The model learns the representation; the feature dimension have no (obvious) meaning. This is the default for deep learning. Word vectors are often used as initial values in NLP to speed up training.

Text Classification

Feature Engineering

Select and evaluate the dimension of the feature space $\mathbf{X}$ manually.

- ❑ Find a good representation of the problem and examples.
- ❑ Performance depends on good feature design.

Features can be any representation or measure of text.

- ❑ Standard content features represent the text. Word counts, token n-grams, . . .
- ❑ Standard structure features represent the linguistic structure. POS, phrase structure n-grams, . . .
- ❑ Task-specific features are tailored to a specific task, usually based on expert knowledge. Local sentiment, discourse relations, flow patterns, . . .

Text Classification

Content Features

Token n-gram frequencies. [NLP:III 50 ff.]

- The relative frequencies of all token 1, 2, . . . , n-grams in a text.
- Each n-gram will become a feature dimension in X, which will lead to large, sparse feature spaces.
- Limit size: Exclude tokens that appear $\leq k$ times or in $\leq k$% of documents.

Word list occurrences.

- How often words from a word list occur in a document.
- Word lists collect terms that share a concept, for example:
  - Linguistic Dimensions
    e.g. Number words: `one, thirty, million, ...`
  - Linguistic Inquiry and Word Count (LIWC) [Pennebaker et al., 2010]
    e.g. Anxiety: `nervous, afraid, tense, ...`
  - General Inquirer
    e.g. Skill aesthetic: `architecture, ballet, ...`

Text Classification

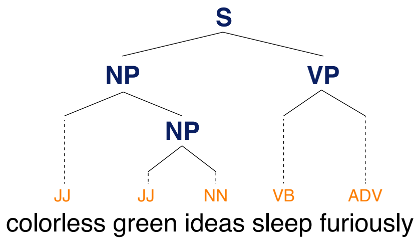
Linguistic Structure Features

Part-of-speech (POS) tag n-grams.

- The relative frequencies of all POS 1, 2, . . . , n-grams in a text.

Phrase Structure Grammar types. [NLP:V]

- The relative frequencies of all PSG types.
- The relative frequencies of all PSG type 2, . . . , n-grams from the serialized grammar structure.



Serialized:

$\texttt{(S (NP JJ (NP JJ NN)}_{NP}\texttt{)}_{NP}\texttt{ (VP VB ADV)}_{VP}\texttt{)}_{S}$

3-grams:

```
(S (NP JJ, (NP JJ (NP, JJ (NP JJ, ...
```

Text Classification

Task-specific features (a selection)

Stylometric features

- The relative frequencies of all character n-grams.
- The relative frequencies of the most frequent function words.
- Lexical statistics. Average numbers of tokens, clauses, and sentences.
- Average length of sentences or paragraphs.
- Readability Score. Flesh-Kincaid

Other text features

- Sentiment polarity of individual words. For sentiment classification
- Starts-with-a-number. For clickbait detection
- . . .

Other non-text features

- Serialized follower-network. For hate-speech detection on twitter
- Time-difference between two messages send. For hate-speech detection on twitter
- . . .

Text Classification

Feature Engineering

Advantages of feature engineering:

- Explainability. Model performance directly indicates good features, which feeds back insight into the task.
- Control. There are no unintended features in the feature space that bias the model.

Drawbacks of feature engineering:

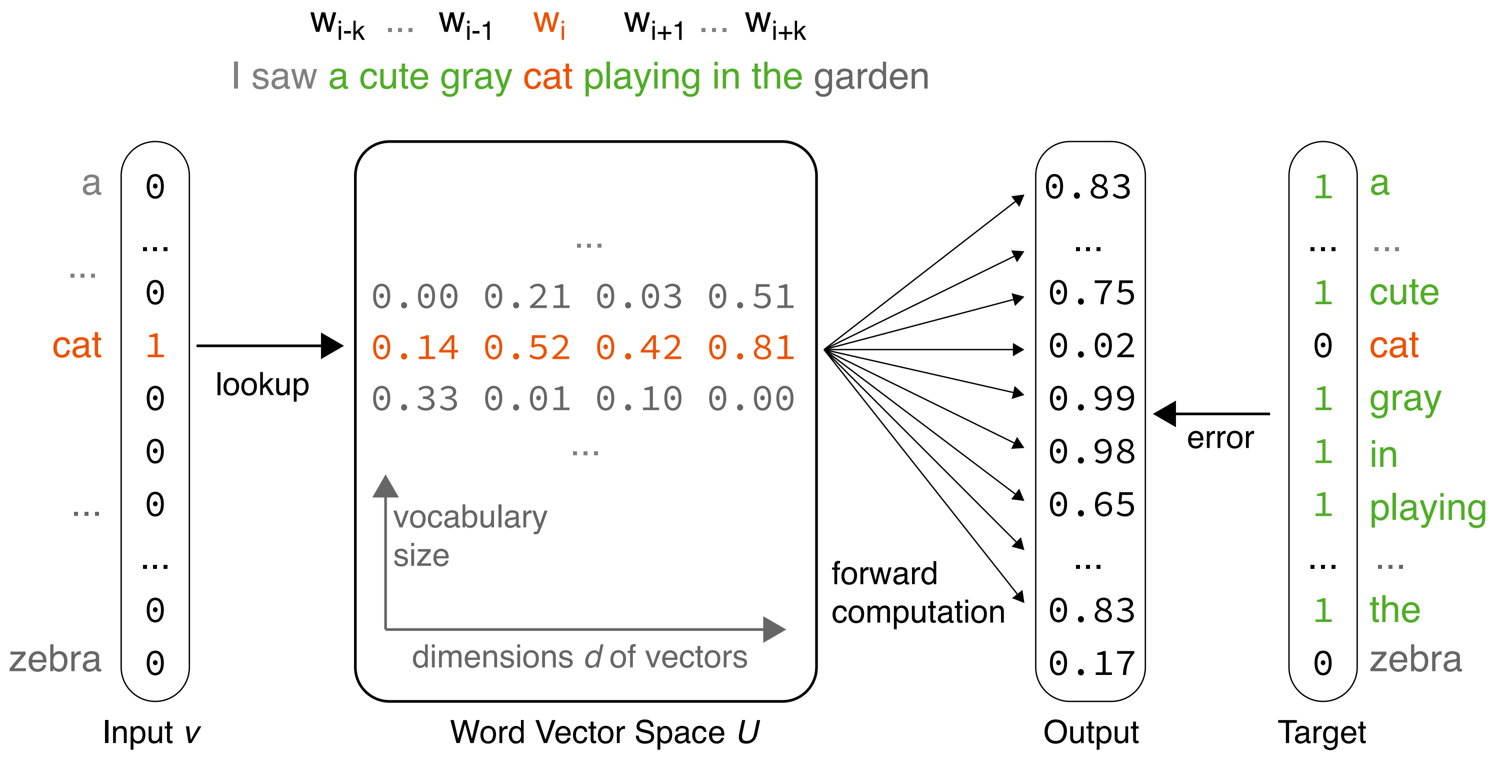
- Expensive evaluation is required to find good representation.
- Feature Space is limited by the developer's understanding of the problem. 'Learning' is not done by the model, but by the designer.
- Feature space X is often very large and sparse. High memory consumption and slow learning.

Text Classification

Representation Learning [NLP:III 66]

Idea: Input the object o verbatim and let a model learn the feature space.

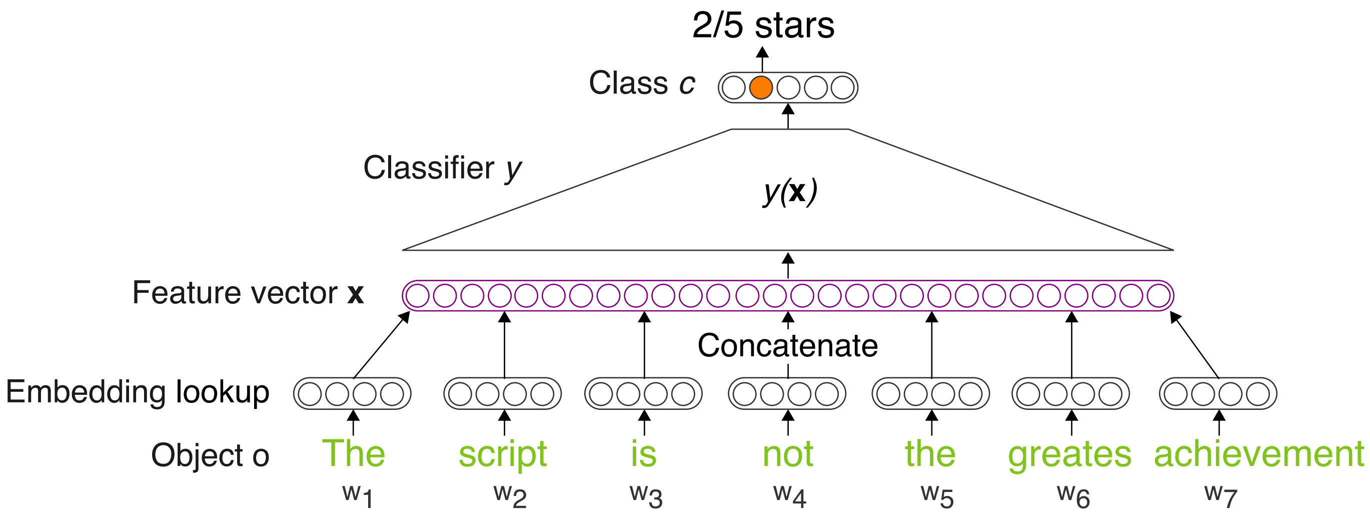
- Learn an embedding (a dense vector) from the verbatim input token.
  Example: Word2Vec



Text Classification
Representation Learning

Idea: Input the object o verbatim and let a model learn the feature space.

- Learn an embedding (a dense vector) from the verbatim input token.
  Example: Word2Vec
- Feature vector: Concatenate all embeddings of the input sequence.
  Other, manual features can still be a- or prepended.



Text Classification

Feature Space Size

Different semantics of the feature spaces $\mathbf{X}$:

- Feature Engineering: Each dimension $x_i \in \mathbf{X}$ is a measure of a property of the input documents. The size of $\mathbf{X}$ is independent of the size of the documents. Example: x_3 is always the count of `aardvark`
- Representation Learning: Each dimension $x_i \in \mathbf{X}$ corresponds to a (latent) dimension k of word w_j in the embedding space. The size of $\mathbf{X}$ varies between inputs. Example: x_3 is the 3rd index of the embedding of w_1

Classification models have a fixed input size! almost always

- Embedding-based feature vectors must be
  - padded (filled with 0 for short inputs), and
  - truncated (cut off after the size limit is reached).
- This puts a limit on the input text length, which is usually not the case with feature engineering.

Text Classification

Feature Space Size

Typical sizes of feature spaces:

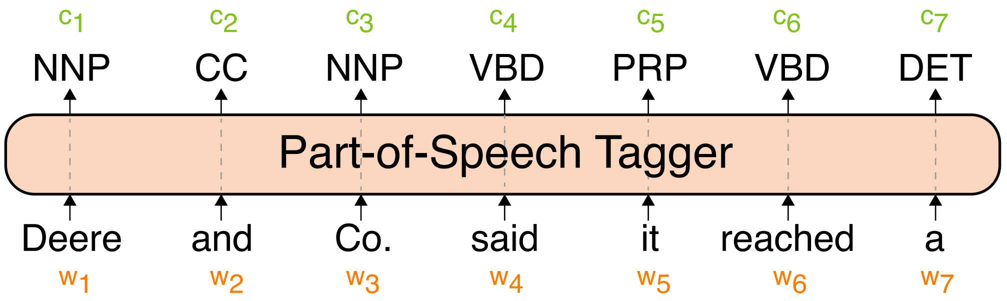
- ❑ BERT has an input vector length of 393,216 (dense). [Devlin et al.]
  - Embedding dimension: 768. bert-base
  - Input sequence length: 512 tokens.
- ❑ Longformer has an input vector length of 3,145,728 (dense). [Beltagy et al.]
  - Embedding dimension: 768.
  - Input sequence length: 4,096 tokens.
- ❑ Engineered spaces with n-gram counts can be (extremely) much larger. From Google n-grams:
  - Unique unigrams: 13,588,391
  - Unique bigrams: 314,843,401
  - Unique trigrams: 977,069,902
- ❑ Engineered spaces can also be very small (1-2 digits), if the features capture the phenomenon very well.

Text Classification

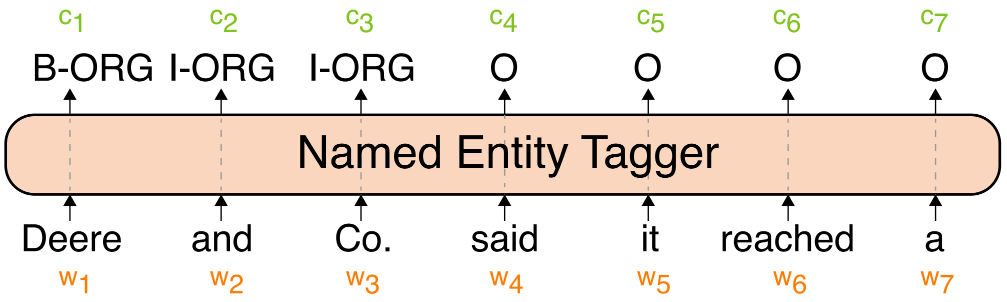
Token Classification

Given an (observed) input sequence of tokens $w_1, \ldots, w_n$, determine the most likely output sequence $c_1, \ldots, c_n$, where c_i is the class of w_i (tagging).

POS-tagging: classify each word with its word class.



Named Entity Recognition: classify each word with **B**egin **I**nside **O**utside.

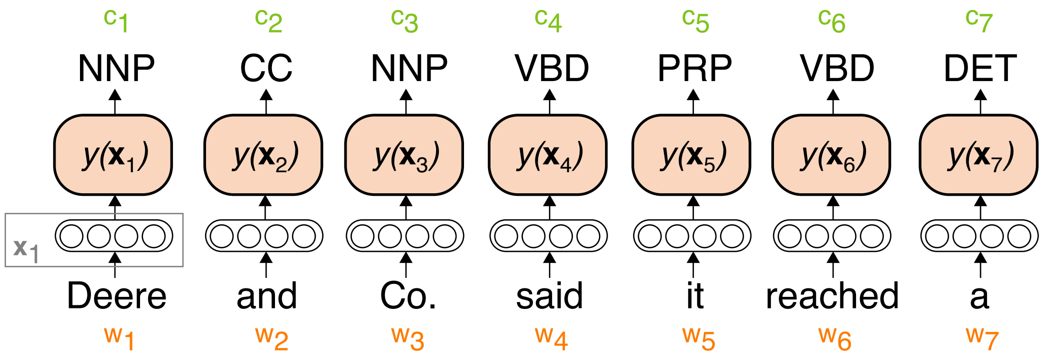


Text Classification

Token Classification

Given an (observed) input sequence of tokens $w_1, \ldots, w_n$, determine the most likely output sequence $c_1, \ldots, c_n$, where c_i is the class of w_i (tagging).

Idea: Classify $c_i = y(\mathbf{x}_i)$ based on a representation $\mathbf{x}_i$ of each individual input token w_i:



c1 c2 c3 c4 c5 c6 c7
NNP CC NNP VBD PRP VBD DET
y(x1) y(x2) y(x3) y(x4) y(x5) y(x6) y(x7)
x1
Deere and Co. said it reached a
w1 w2 w3 w4 w5 w6 w7

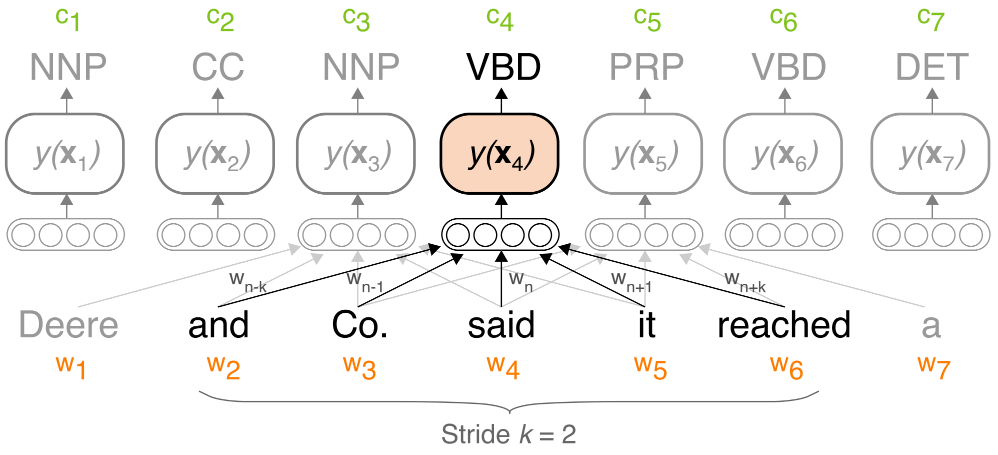
- $y(\mathbf{x}_i)$ could simply be a maximum likelihood estimator. Works well for POS-tagging but poorly for Named Entity Recognition. Why?
- $\mathbf{x}_i$ can be the vocabulary index of w_i, a contextualized word vector, ...

Text Classification

Token Classification

Given an (observed) input sequence of tokens $w_1, \ldots, w_n$, determine the most likely output sequence $c_1, \ldots, c_n$, where c_i is the class of w_i (tagging).

Idea: Classify $c_i = y(\mathbf{x}_i)$ based on a representation $\mathbf{x}_i$ of the span $w_{i-k}, \ldots, w_{i+k}$ with stride k.



c1 c2 c3 c4 c5 c6 c7
NNP CC NNP VBD PRP VBD DET
y(x1) y(x2) y(x3) y(x4) y(x5) y(x6) y(x7)
wn-k wn-1 wn wn+1 wn+k
Deere and Co. said it reached a
w1 w2 w3 w4 w5 w6 w7
Stride k = 2

Text Classification

Token Classification

Given an (observed) input sequence of tokens $w_1, \ldots, w_n$, determine the most likely output sequence $c_1, \ldots, c_n$, where c_i is the class of w_i (tagging).

Idea: Classify $c_i = y(\mathbf{x}_i)$ based on a representation $\mathbf{x}_i$ of the span $w_{i-k}, \ldots, w_{i+k}$ with stride k.

Typical features in $\mathbf{x}_i$ are :

1. For $w_{i-k}, \ldots, w_{i+k}$ the vocabulary indices, pre- and suffixes, capitalization, or occurances in word lists.
2. For $w_{i-k}, \ldots, w_{i-1}$ the already determined tags.

Tagging with span classification:

1. Pad all sequence on both sides with a special token.

   `[PAD] [PAD]` Dere and Co. said ... `[PAD] [PAD]`
2. Model the training examples $D = \{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_n, c_n)\}$ for all sequences in training dataset.
3. Train the classifier $y(\mathbf{x})$.
4. Predict c_i identically by padding and then iterating over the words.

Remarks:

- Note the notation: In general sequence processing (like machine translation or language modelling), w is used for the input and y for the output. In tagging, the output is often denominated with l. We use c in the examples to be consistent to machine learning notation: c denotes classes, $\mathbf{x}$ feature vectors, and y the classifier.
- Span-based token classification is also often used for transition-based syntax parsers like arc-standard.
- Common special tokens are `[PAD]`, `[UNK]`, `[MASK]`, `[SEP]`, and `[EOS]`. These are usually treated as individual words by the tokenizer, they are not preprocessed, and they have their own vocabulary indices.
- Its possible to offset the stride to include more left-side than right-side context, or vice versa.
- The classifier $y(\mathbf{x})$ can also be a set of hand-crafted rules.

Text Classification

Common Classification Algorithms [ML:I 76]

- Naïve Bayes.
  Predicts classes based on conditional probabilities of feature values.
- Support vector machines.
  Maximizes the margin between examples and the decision boundary.
- Decision tree.
  Sequentially compares examples on single features, selected via information gain.
- Random forest.
  Majority voting based on several decision trees.
- Neural networks.
  Learn complex function on feature combinations.

. . . and many more

Text Classification

Evaluation [NLP:II 10 ff., ML:II 112 ff.]

Evaluating Model Effectiveness [Joachims, 2002]

- Corpus. Reuters-21578, 90 topics, 9603 training and 3299 test texts.
- Features. More than 10,000 word-based features.
- Classification. Four baseline algorithms and different SVM variations.
- Optimization. Hyperparameters (incl. #features) tuned for all algorithms.

Effectiveness (F_1-score)

Learning	**10 Most Frequent Topics**										**Micro**
Algorithm	earn	acq	mny-fx	grain	crude	trade	intrst.	ship	wheat	corn	**F_1**
Naïve Bayes	96.0	90.7	59.6	69.8	81.2	52.2	57.6	80.9	63.4	45.2	72.3
Rocchio	96.1	92.1	67.6	79.5	81.5	77.4	72.5	83.1	79.4	62.2	79.9
Decision trees	96.1	85.3	69.4	89.1	75.5	59.2	49.1	80.9	85.5	87.7	79.4
k nearest neighbors	97.8	91.8	75.4	82.6	85.8	77.9	**76.7**	79.8	72.9	71.4	82.6
Linear SVM (C 0.5)	98.0	95.5	**78.8**	91.9	**89.4**	**79.2**	75.6	**87.4**	86.6	87.5	86.7
Linear SVM (C 1.0)	**98.2**	**95.6**	78.5	93.1	**89.4**	**79.2**	74.8	86.5	**86.8**	**87.8**	**87.5**
RBF-SVM	98.1	94.7	74.3	**93.4**	88.7	76.6	69.1	85.8	82.4	84.6	86.4

Text Classification

Dataset Preparation

Annotations present in text corpora often do not match the task instances required for supervised classification.

- There may be no negative instances, who are required for training.
  `[Jaguar]`$_{ORG}$ `is named after the animal jaguar.`

- Annotations may have to be mapped to other task instances.
  1–2 Stars → *negative*, 3 Stars → *neutral*, 4–5 Stars → *positive*

- Some classes may be more or less common than others. For learning, a balanced distribution of the target variable is sometimes preferable.
  → Oversample rare classes, undersample common classes.

Text Classification

Dataset Preparation: Negative Instances

Why "negative" instances?

- In many classification tasks, one class is in the focus.
- Other classes may not be annotated, or are more specific than needed.

  False token boundaries, spans that are *not* entities, different neutral sentiments, . . .

Defining negative instances

- What is seen as a "negative" instance is a design decision.
- The decision should be based on what a classifier should be used for.
- Trivial cases may distract classifiers from learning relevant differences.

Example: Negative instances in person entity recognition

"$[\text{tim}]_{PER}$ works in $[\text{cupertino}]_{LOC}$. $[\text{san fran}]_{LOC}$ is his home. as a cook, he cooks all day."

- All other named entities?
- All other content words?
- All other noun phrases?

Text Classification

Dataset Preparation: Negative Instances

Why "negative" instances?

- In many classification tasks, one class is in the focus.
- Other classes may not be annotated, or are more specific than needed.

  False token boundaries, spans that are *not* entities, different neutral sentiments, . . .

Defining negative instances

- What is seen as a "negative" instance is a design decision.
- The decision should be based on what a classifier should be used for.
- Trivial cases may distract classifiers from learning relevant differences.

Example: Negative instances in person entity recognition

"$[\text{tim}]_{PER}$ works in $[\text{cupertino}]_{LOC}$. $[\text{san fran}]_{LOC}$ is his home. as a cook, he cooks all day."

- All other named entities? $\rightarrow$ Can only distinguish entity *types* then.
- All other content words? $\rightarrow$ Verbs will never be person names (somewhat trivial).
- All other noun phrases? $\rightarrow$ Seems reasonable.

Text Classification

Dataset Preparation: Mapping of Target Variable Values

What is the mapping of target variable values?

- The alteration of the target classes or values in a given corpus/dataset.
- May require expert knowledge about the target variable.

Why mapping?

- Corpus annotations may be more fine-grained than needed.

  Sentiment scores: $[1, 2] \rightarrow$ “negative”, $]2, 4[\rightarrow$ “neutral”, $[4, 5] \rightarrow$ “positive”

- Some values or differences may not be relevant in a given application.

  Polarities: “negative” $\rightarrow$ “negative”, ignore “neutral”, “positive” $\rightarrow$ “positive”

- The ranges of the “same” target variable may not match across corpora.

  Sentiment scores: $\{1, 2\} \rightarrow 0$, $\{3\} \rightarrow 1$, $\{4, 5\} \rightarrow 2$

- The category “names” may just not be those desired (purely “aesthetical”).

  Polarities: “bad” $\rightarrow$ “negative”, “medium” $\rightarrow$ “neutral”, “good” $\rightarrow$ “positive”

Text Classification

Dataset Preparation: Balancing Datasets

What is dataset balancing?

- The alteration of the distribution of a dataset with respect to some target variable, such that the distribution is uniform afterwards.
- Balancing is more common for classification than for regression, since in regression there is often no fixed set of target values to balance.

  A solution for regression is *binning* (i.e., to balance the distribution for certain intervals).

When to balance?

- Balancing a training set prevents machine learning from being biased towards majority classes (or values).
- Validation and test sets should usually not be balanced, since analysis algorithms are usually evaluated on *representative* distributions.

How to balance?

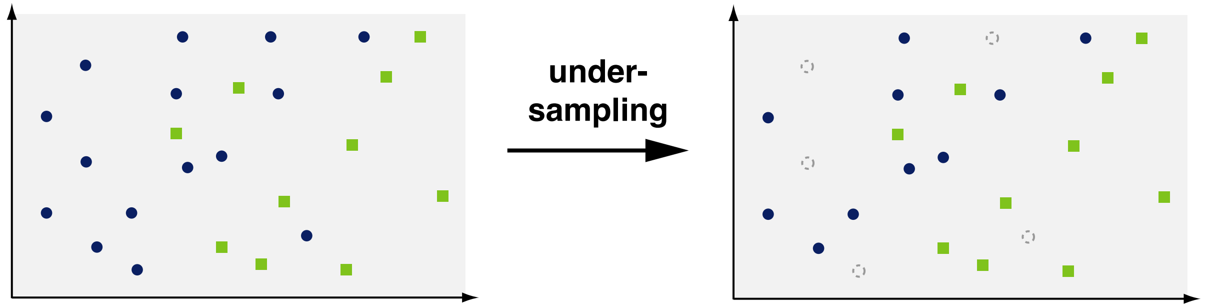
- Undersampling. Removal of instances from majority classes.
- Oversampling. Addition of instances from minority classes.

Text Classification

Dataset Preparation: Balancing Datasets

How to balance with undersampling?

- Removing instances of all non-minority classes until all classes have the size of the minority class.
- Instances to be removed are usually chosen randomly.



Pros and cons

- Pro. All remaining data is real.
- Pro. Downsizing of a dataset makes training less time-intensive.
- Con. Instances that may be helpful in learning are discarded (i.e., potentially relevant information is lost).

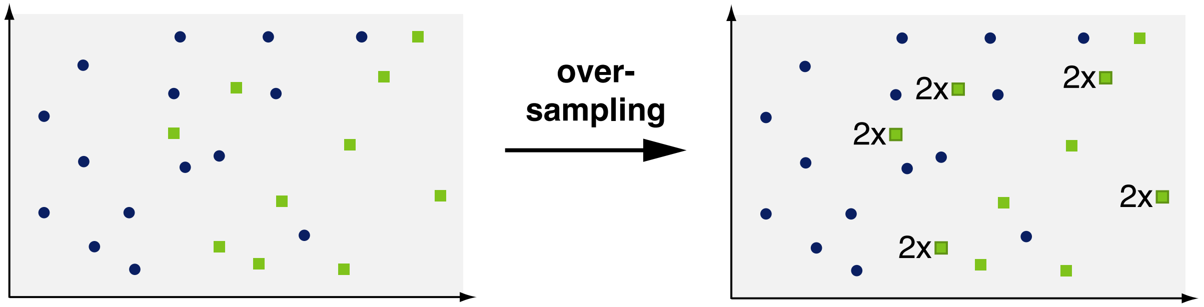
Text Classification

Dataset Preparation: Balancing Datasets

How to balance with oversampling?

- Adding instances of all minority classes until all classes have the size of the majority class.
- Usually, the instances to be added are random duplicates.

  Where reasonable, an alternative is to create artificial instances using interpolation.



Pros and cons

- Pro. No instance is discarded (i.e., all information is preserved).
- Con. Upsizing of a dataset makes training more time-intensive.
- Con. The importance of certain instances is artificially boosted, which may make features discriminative that are actually noise.

Text Classification

Dataset Preparation: Balancing Datasets

Undersampling vs. Oversampling
Example: A sentiment training set

- 1000 positive, 500 negative, 100 neutral instances.
- After undersampling?
- After oversampling?

Dataset Preparations

Undersampling vs. Oversampling

Example: A sentiment training set

- 1000 positive, 500 negative, 100 neutral instances.
- After undersampling? $\rightarrow$ 100+100+100 instances
- After oversampling? $\rightarrow$ 1000+1000+1000 instances

What to use?

- When more than enough data is available, undersampling is preferable.
- When the class imbalance is low, oversampling is rather unproblematic.
- When the class imbalance is high, no really good choice exists.

Alternatives to balancing?

- Many machine learning optimization procedures can penalize wrong predictions of minority instances more than majority instances.
- Conceptually, this is the more sound way of preventing the bias.
- Practically, it makes the learning process more complex, which is why balancing is often used.

Text Classification

Case Study: Review Sentiment Analysis

Sentiment classification of reviews

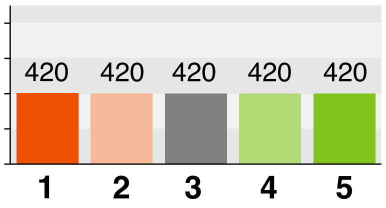
- Classification of the nominal sentiment polarity or score of a customer review on a product, service, or work of art.

Data

- 2100 English hotel reviews from TripAdvisor.

  900 training, 600 validation, and 600 test reviews.
- Each review has a sentiment score from {1, . . . , 5}.

ArguAnaTripAdvisor corpus



Tasks

- 3-class sentiment. 1–2 mapped to negative, 3 to neutral, 4–5 to positive.

  Training set balanced with random undersampling.
- 5-class sentiment. Each score interpreted as one (nominal) class.

Approach

- Algorithm. Linear SVM with one-versus-all multi-class handling.

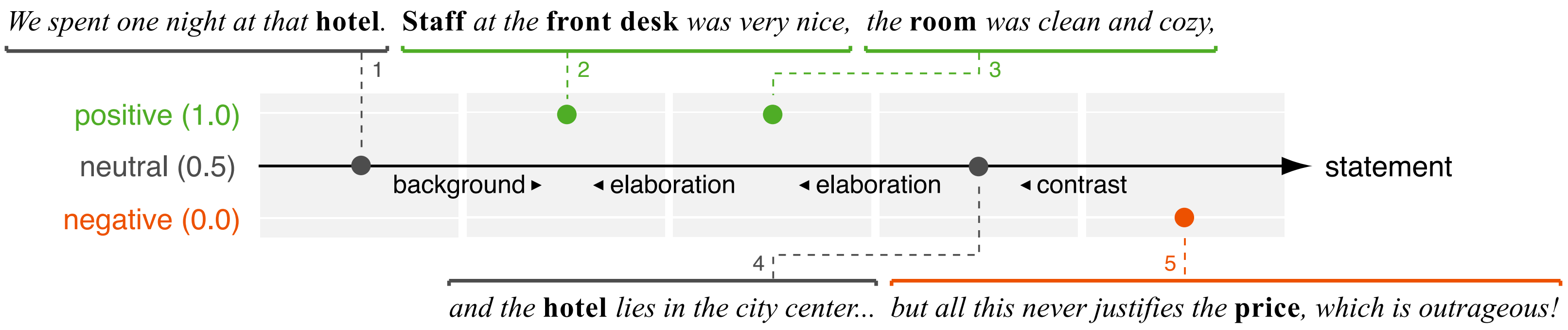
  Cost hyperparameter tuned on validation sets.
- Features. Combination of several standard and specific feature types.

Text Classification

Review Sentiment Analysis: Review Argumentation

Example hotel review

“We spent one night at that hotel. Staff at the front desk was very nice, the room was clean and cozy, and the hotel lies in the city center. . . but all this never justifies the price, which is outrageous!”


We spent one night at that hotel.
Staff at the front desk was very nice,
the room was clean and cozy,
1
2
3
positive (1.0)
neutral (0.5)
negative (0.0)
background ▸
◂ elaboration
◂ elaboration
◂ contrast
statement
4
5
and the hotel lies in the city center...
but all this never justifies the price, which is outrageous!

A shallow model of review argumentation

- A review can be seen as a flow of local sentiments on domain concepts that are connected by discourse relations.

  The domain concepts (aka aspects) are omitted in the following.

Text Classification

Review Sentiment Analysis: Evaluation of Standard Features

Evaluation

- One linear SVM for each feature type in isolation and for their combination.
- Training on training set, tuning on validation set, application on test set.

Discussion

- Token unigrams best, but some other types close.
- Combination does not outperform single types.
- 60.8% accuracy does not seem satisfying.

Effectiveness results (accuracy)

Category	Feature type	# Features	3 classes
Content	Token unigrams	426	**60.8%**
	Token bigrams	112	49.5%
	Token trigrams	64	24.5%
	Core vocabulary	83	41.7%
	Sentiment scores	6	59.3%
	Sentiment words	123	60.5%
Style	POS unigrams	48	51.3%
	POS bigrams	70	49.0%
	POS trigrams	118	45.5%
	Phrase unigrams	13	48.8%
	Phrase bigrams	43	52.5%
	Phrase trigrams	122	50.8%
	Function words	100	57.3%
	Character trigrams	200	48.7%
	Lexical statistics	6	42.8%
Combination of features		**1534**	**60.8%**

Text Classification

Review Sentiment Analysis: Specific Feature Types

Local sentiment distribution

- The relative frequencies of positive, neutral, and negative local sentiment as well as of changes of local sentiments.

  positive 0.4 neutral 0.4 negative 0.2 (neutral, positive) 0.25 . . .

- The average local sentiment value from 0.0 (negative) to 1.0 (positive).

  average sentiment 0.6

- The interpolated local sentiment at each normalized position in the text.

  example normalization length 9: (0.5, 0.75, 1.0, 1.0, 1.0, 0.75, 0.5, 0.25, 0.0)

Discourse relation distribution

- The relative frequencies of discourse relation types in the text.

  background 0.25 elaboration 0.5 contrast 0.25 (all others 0.0)

- The relative frequencies of combinations of types and local sentiments.

  background(neutral, positive) 0.25 elaboration(positive, positive) 0.25 . . .

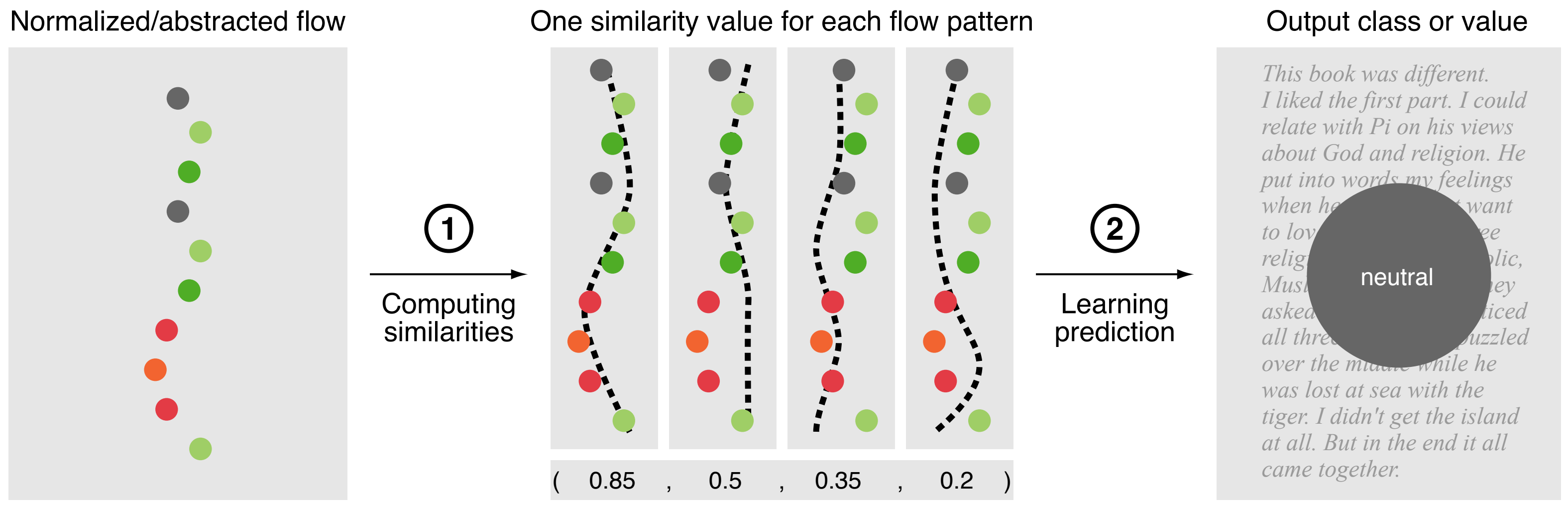
Text Classification

Review Sentiment Analysis: Specific Feature Types

Sentiment flow patterns

- The similarity of the normalized flow of the text to each flow pattern.

  Details on the patterns in the lecture part on clustering.



This book was different. I liked the first part. I could relate with Pi on his views about God and religion. He put into words my feelings when he … want to lov… ee relig… lic, Mus… ey asked… iced all thre… uzzled over the mi… while he was lost at sea with the tiger. I didn't get the island at all. But in the end it all came together.

Content and style features

- Content. Token n-grams, sentiment scores.
- Style. Part-of-speech n-grams, character trigrams, lexical statistics.

  Subset of the standard features above, here as baseline features.

Text Classification

Review Sentiment Analysis: Evaluation of the Specific Feature Types

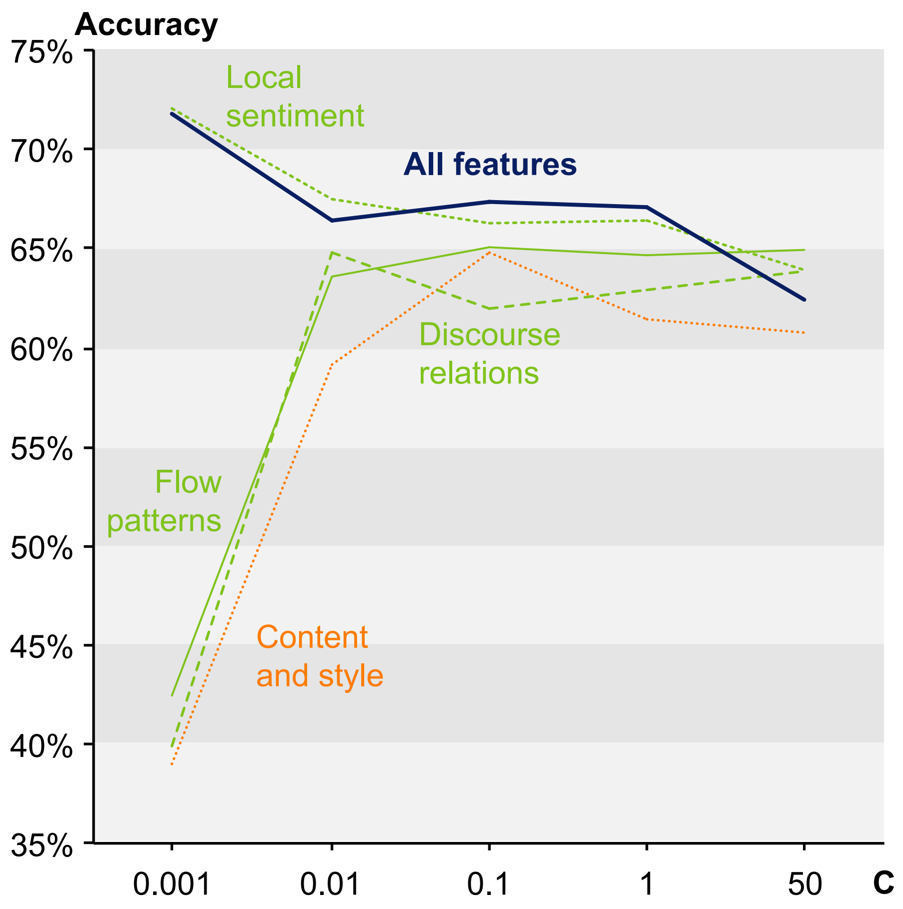
Evaluation

- One linear SVM for each feature type in isolation and for their combination.
- Training on training set, tuning on validation set, application on test set.
- Both 3-class and 5-class.

Cost hyperparameter tuning

- C values tested for SVM. 0.001, 0.01, 0.1, 1.0, 50.0
- Best C used on test set.
- Results shown here for the 3-class task only.

Validation accuracy depending on C



Text Classification

Review Sentiment Analysis: Results and Discussion for the Specific Features

Effectiveness on test set (accuracy)

Feature type	# Features	3 Classes	5 Classes
Local sentiment distribution	50	69.8%	42.2%
Discourse relation distribution	75	65.3%	40.6%
Sentiment flow patterns	42	63.1%	39.7%
Content and style features	1026	58.9%	43.2%
Combination of features	1193	**71.5%**	**48.1%**
Random baseline		40.0%	20.0%

Discussion

- Content and style features. A bit weaker than in the experiment above, due to slight differences in the experiment setting.
- Sentiment flow patterns. Impact is more visible across domains (later).
- Combination of features. Works out this time, so more complementary.

- The 5-class accuracy seems insufficient.
- Classification misses to model the ordinal relation between classes; regression might be better.

Text Classification

Review Sentiment Analysis: Impact of Preprocessing

Evaluation on ground-truth local sentiment

- The specific features rely on the output of local sentiment classifiers.

  Subjectivity accuracy 78.1%, polarity accuracy 80.4%

- Using the ground-truth local sentiment available in the given data, the impact of errors in this output can be assessed.

Effectiveness results with ground-truth local sentiment

Feature type	# Features	MAE	RMSE
Local sentiment distribution	50	**0.61**	0.77
Discourse relation distribution	75	0.68	0.84
Sentiment flow patterns	42	0.67	0.86
Content and style features	1026	0.90	1.11
Combination of features	1193	**0.61**	**0.75**
Random baseline		1.20	1.41

Discussion

- Notable error reduction, but room for further improvement remains.