Syntactic N-grams as Machine Learning Features for Natural Language Processing

Marvin Gülzow

Table of Contents

TL;DR

- Introduce Syntactic n-grams
- Use them for authorship attribution
- Compare machine learning approaches
  - Support Vector Machines
  - Naive Bayes
  - J48 (decision tree)
- $\Rightarrow$ SVM + SN-Grams work well

Section 1

Basics

N-Grams

Definition

n-gram:

$$w = (w_1, \ldots, w_n) \in \Sigma$$

- n sequential items from a text
- "item": characters, words, phonetic units, linguistic features, ...
- "sequential": Neighborship relation required
- $\Rightarrow$ Text fragments
- $\Rightarrow$ Probatilistic features

[2]

Syntactic N-Gram:

Definition

Syntactic N-Gram: "An n-gram obtained based on the order in which the elements appear in syntactic trees"

- Items: SR-Tag (syntactic-relation tag)
- Neighborship relation: Lie on same path
- Syntactic tree: Parse result according to *formal* grammar
- Issue: *Natural* language processing?
- Stanford NLP suite
- "SN-Grams of SR-tags"

[1], [2]

SN-Grams Example

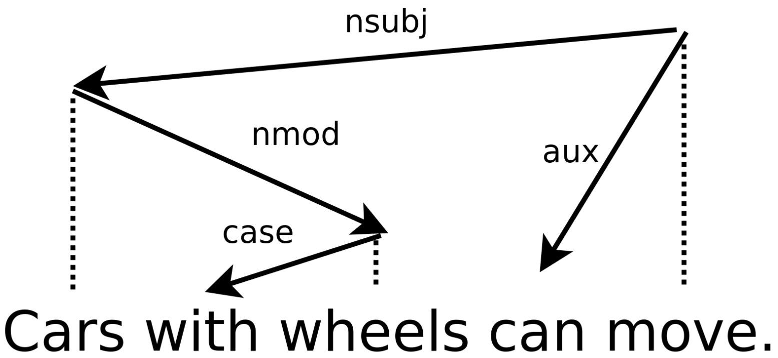
"Cars with wheels can move"

```
-> move/VB (root)
  -> Cars/NNS (nsubj)
    -> wheels/NNS (nmod:with)
      -> with/IN (case)
  -> can/MD (aux)
```

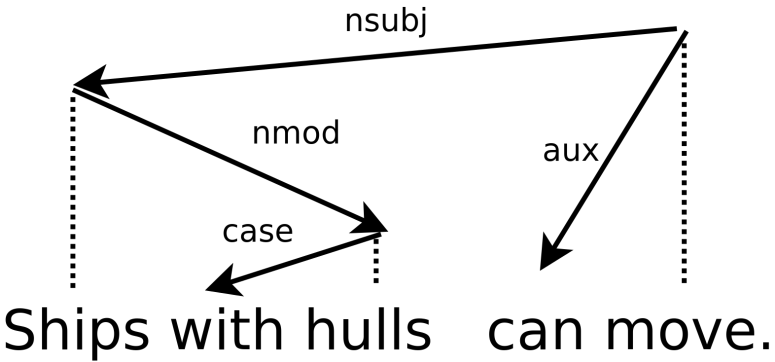
"Ships with hulls can move"

```
-> move/VB (root)
  -> Ships/NNS (nsubj)
    -> hulls/NNS (nmod:with)
      -> with/IN (case)
  -> can/MD (aux)
```

SN-Grams Example



SN-Grams Example



Resulting SN-Grams

- (aux)
- (nsubj, nmod)
- (nmod, case)
- (nsubj, nmod, case)
- $\Rightarrow$ Independent of content.

Syntactic N-Grams

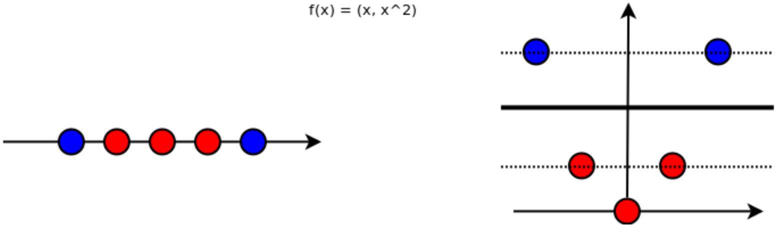
- Advantages
  - “real” neighbors: No arbitrary influence from content
  - Assumption: Captures author’s writing style
- Disadvantages
  - Preprocessing is expensive (only once though).
  - Parser Quality determines results
  - Good parsers not available for every language

Support Vector Machine (SVM)

- Deterministic binary classifier
- **linear** separation of classes
- Separator: Hyperplane
- $\rightarrow$ Gap between classes has maximum width
- Non-linearly separable Data?

[4]

Kernel trick



Kernel trick

- Add dimensions
- Warp data
- $\Rightarrow$ Transformation via kernel-function
- $\Rightarrow$ Restricted to numerical data
- $\Rightarrow$ Multiclass-classification via multiple Binary classification

SVM learning

1. Choose appropriate kernel (human)
2. Project data into target vector space
3. Find optimum separator
    - Maximize distance of each object to separator
    - $\Rightarrow$ Items defining border are **support vectors**

[6]

Support Vector Machine (SVM)

- Advantages
  - Non-linear spearation
  - "Tunable" to noise
  - More robust against biased data
  - Unique, global solution exists
  - $\Rightarrow$ High accuracy
- Disadvantages
  - Only work on numerical data
  - Learned model not interpretable
  - Training in $O(n^2)$

[5]

Support Vector Machine (SVM)

- WEKA/LibSVM
- SVMs work on numerical Data
- We have: Nominal data
- $\Rightarrow$ Map semantic relation to numbers

Naive Bayes

- How many times does an attribute appear in a class?
- $\Rightarrow$ Look at each attribute of item to classify
- $\Rightarrow$ Probabilities determine class
- Each classified object contributes to training set
- Used as a reference for other learners

[4]

Naive Bayes

- Bayes' Theorem

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

- Naïve assumption: all attributes are *independant*
- $$P(E = (a_1, \ldots, a_n)|h) = \prod_{a_i \in E} P(a_i|h)$$
- $$P(a_i|h) = \frac{\#\text{data from class } h \text{ with } A_i = a_i}{\#\text{data from class } h}$$
- Object class $\Rightarrow$ most probable

[4]

Naive Bayes

Advantages

- Easy to implement
- Fast implementation possible
- Learns with each example
- Somewhat accurate
- Standard comparison for other classifiers

Disadvantages

- Attributes are usually **not** independant
- Probabilities may be unavailable

J48

- Descision tree builder
- Entropy based
- $\Rightarrow$ Which attribute yields the highest **information gain**?
- Builds optimum descision tree
- $\Rightarrow$ Human-interpretable model

[6]

J48 - Information Gain

- Given: Labelled dataset
- Find: Attribute which is optimal for discriminating between classes
- Calculate *entropy* of training set T

$$e(T) = -\sum_{i=1}^{k} p_i \cdot \log_2 p_i$$

- Calculate information gain for attribute A

$$IG(T, A) = e(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot e(T_i)$$

- $\Rightarrow$ Tree splits data on this attribute
- Repeat
- Other split critera: Gini-Index, χ^2, Randomly, ...

[6]

J48

Advantages

- Model can be interpreted for other uses
- Fast classification (precomputed model)
- Can fix missing values (parser errors)

Disadvantages

- Require pruning
- Sensitive to noise
- Greedy approach can get stuck

Section 2

Approach

Dataset

- English novels
  - Booth Tarkington (13)
  - George Vaizey (13)
  - Louis Tracy (13)
- 24 for Training, 11 for classification
- Total of 6.1 MB

Algorithm

1. Parse Corpus using StanfordNLP
2. Extract syntactic relations (SR-tags)
3. Construct SN-grams $\Rightarrow$ Profile
4. Classify as usual
5. Establish baseline using other classifiers

Experiments

- N-Grams
  - Word based
  - POS (Part Of Speech)
  - Character based
  - SR-Tags
- Vary n-gram size form 2 to 5
- Profile sizes from 400 to 11000
- Use J48 and NB as baseline

Section 3

Results

Results - In brief

- All classifiers better than 50% accuracy
- SVMs outperform other classifiers
- SR-tags yield better results than other tags
- Bigrams and trigrams better than 4- and 5-grams
- 100% accuracy in some cases

[1], [3]

Results

```
// Show tables from the paper now.
goto PAPER_RESULTS;
```

Section 4

Assessment

Positive

- SN-Grams provably more accurate than other approaches
- Able to reliably identify author in a small pool of possible authors
- Solid theoretical basis (SVM and parsing)
- Hard to hide author's grammatical habits

Negative

- Parsing takes "considerable time" on 39 novels
  $\Rightarrow$ Mentioned in paper, as expected
- Parser has extreme influence on result
- $\Rightarrow$ What about "wierd" texts?
  - Non-natives with the speaking of bad grammatics
  - Fantasy/Scifi "bogus" words
- SVM models not interpretable

Paper quality

Positive:

- Good explanation of SN-Grams
- Thorough comparison of many cases
- Clear results
- New, practical method found

Negative:

- Hard to reproduce:
  - Examples inconsistent
  - No concrete parameters given (Learners!)
  - Tool versions missing
- Small set of candidate authors (3)

Section 5

Own implementation

Own implementation

Not just yet :(

Section 6

Sources

Sources

- G. Sidorov et. al.: *Syntactic N-grams as Machine learning Features for Natual Language Processing*, CIC Mexico, IPN Mexico, University of the Aegean (Greece)
- Stanford Cousera lecture on language modeling (`https://class.coursera.org/nlp/lecture/17`)
- Efstathios Stamatatos, *A Survey of Modern Authorship Attribution Methods*, Dept. of Information and Communication Systems Eng, University of the Aegean
- Michael Berthold and Iris Adae: *SVMs and Rule Learning* Lecture held at the University of Constance, Winter term 2014/15
- Laura Auria and Rouslan A. Moro: *Support Vector Machines (SVM) as a Technique for Solvency Analysis*, DIW Berlin, 2008