SVM ALGORITHMS FOR SENTIMENT ANALYSIS

Advanced Topics in Computer science
Project

Pegoraro Alessandro

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO MATEMATICA
Dipartimento di Matematica "Tullio Levi-Civita"

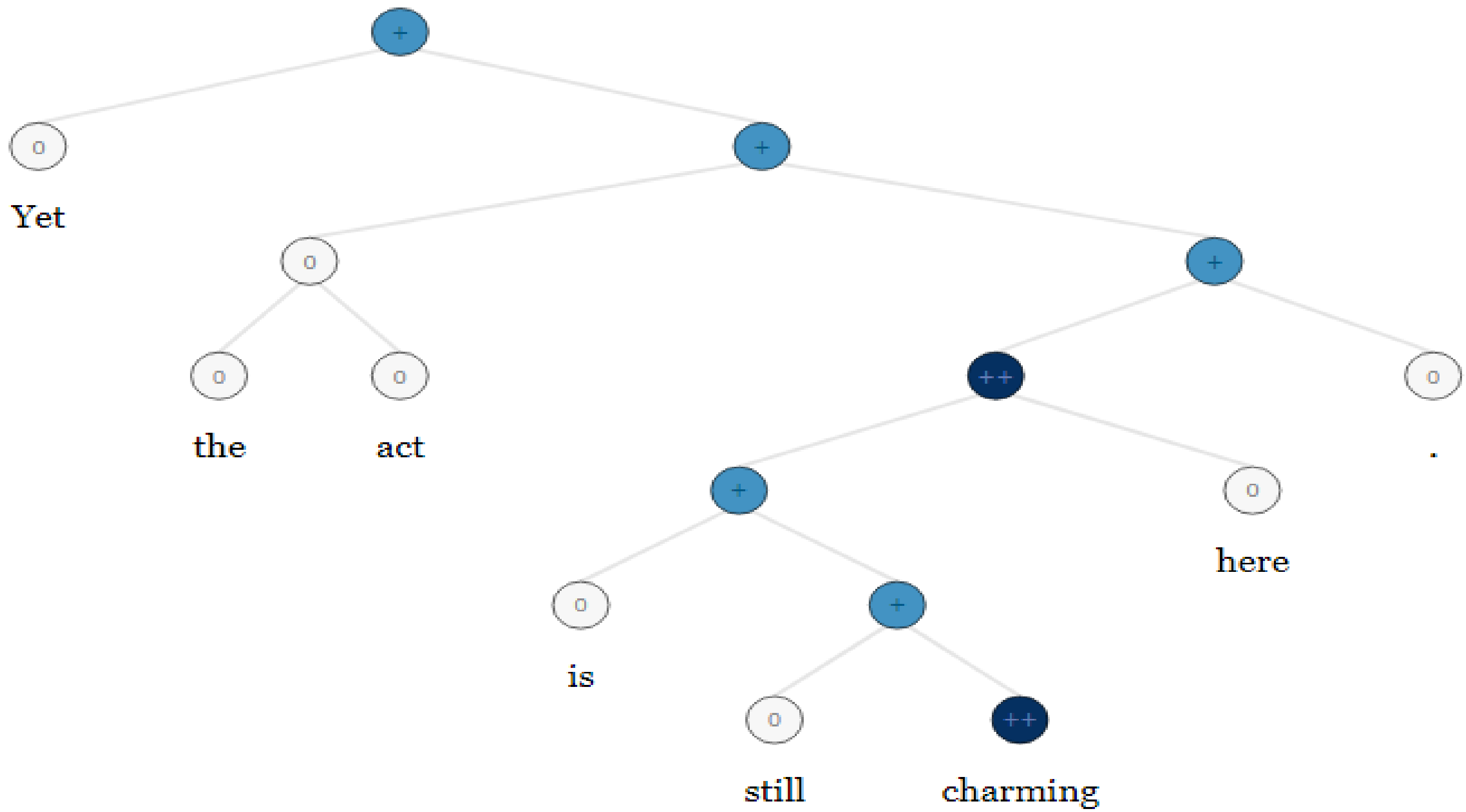
Sentiment analysis

- Classification task:
  - Given a text as input, we want to identify what the subject feels about the object of the text
  - In our specific task the text will be a movie review and the object will be a movie
- Two different versions:
  - Binary (Positive, Negative)
  - Fine-grained with neutral reviews

Dataset

- Stanford Sentiment Treebank SST-5 and SST-2
  - Movie review sentences labelled with 5 classes
  - Each sentence is represented in a binary parse tree
  - Each node represents a phrase and is labeled
  - The root node represent the true sentiment of the review
  - On SST-5 the root label can range between *negative, somewhat negative, neutral, somewhat positive* and *positive*
  - ON SST-2 the neutral tree are discarded, and it becomes: *negative* or *somewhat negative* vs *somewhat positive* or *positive*

Dataset



- String Representation

(3 (2 Yet) (3 (2 (2 the) (2 act))(3 (4 (3 (2 is) (3 (2 still) (4 charming)))(2 here))(2 .))))

Related work

- Current Best Models for SST
  - **RNTN:** *Recursive deep models for semantic compositionality over a sentiment treebank, 2013.* ***SST-5:*** *45.70%* ***SST-2:*** *85.40%*
  - **LSTM:** *Improved Sentence Modeling using Suffix Bidirectional LSTM, 2018.* ***SST-5:*** *56.20%*
  - **RoBERTa:** *Self-explaining structures improve nlp models, 2020.* ***SST-5:*** *59.10%*
  - **RoBERTa:** *SMART: Robust and Efficient Fine-Tuning for Pre-trained Natural Language Models through Principled Regularized Optimization, 2019.* ***SST-2:*** *97.50%*

- Tree kernel for SVM
  - **Tagging Kernel:** *New Ranking Algorithms for Parsing and Tagging, 2002.*
  - **Semantic Role Labeling:** *Semantic Role Labeling via Tree Kernel Joint Inference, 2006.*
  - **Tree Kernel:** *Kernel Methods for Tree Structured Data, 2009.*

Approach

- Traditional SVM preprocessing and feature extraction
  - Bag of Words
  - Tf-idf representation
  - Part of Speech
  - Tree label

  Using the scikit-learn library

- Tree structure exploitation with Tree kernels for SVM
  - Tree preprocessing
  - Subtree kernel
  - SubSet Tree kernel
  - Partial Tree kernel
  - String kernel

  Using the SVM-LIGHT-TK library

Traditional SVM

- **Model1**: Root sentence + Tf-idf representation
- **Model2**: PoS tagging + Bag of word + root's children's label
- **Model3**: PoS tagging + Tf-idf representation + root's children's label

The SVM in the scikit-learn library could only accept array with a fixed number of numerical features

Tree kernel for SVM

- **Subtree kernel ST**
  weighted sum of the number of matching proper subtrees

$$K_{subtree}(T_1, T_2) = \sum_{t_1 \in T_1} \sum_{t_2 \in T_2} C(t_1, t_2)$$

with $\boldsymbol{C(t_1,t_2)}$ sums of all matching features rooted in t_1 and t_2

- **SubSet Tree kernel SST**
  weighted sum of the number of shared subset trees

$$K_{subset}(T_1, T_2) = \sum_{s \in m} h_s(T_1) h_s(T_2)$$

with $\boldsymbol{h_s(T)}$ the number of times the subset tree ***s*** occurs in ***T***

Tree kernel for SVM

- **Partial Tree kernel PT**

  weighted sum of the number of all matching subtrees
  same formulation as **ST**
  different local kernel $\boldsymbol{C(t_1,t_2)}$

$$C(t_1, t_2) = 1 + \sum_{J_1, J_2, |J_1|=|J_2|} \prod_{i=1}^{|J_1|} C(ch_{t_1}[J_{1i}], ch_{t_2}[J_{2i}])$$

with J_{1i} and J_{2i} index associated with the child ch_{t_1} and ch_{t_2} respectively

Tree kernel for SVM

The SVM-LIGHT-TK library provided the following Tree kernels:

- **ST:** SubTree kernel
- **SST:** SubSet Tree kernel
- **SST-BoW:** SubSet Tree kernel + Bag of Word with leaves as features
- **PT:** Partial Tree kernel
- **SSTK kernel:** Fast Partial Tree kernel within first tree level + SubSet Tree kernel for the remaining tree level
- **IBRID:** Partial Tree kernel + no leaves contribution
- **STRING:** String representation comparison

Results

- All but the SubTree kernel outperformed the SVM models in SST-2
- The **Empty** preprocessing did not brought up any improvement in any kernel
- All but the **IBRID** kernel perform best with no preprocessed trees

	SST-2		
Kernel	Normal	PoS	Empty
SSTK	96.21%	95.83%	94.45%
ST	78.20%	75.34%	75.01%
SST	96.21%	95.83%	94.45%
SST-BOW	96.10%	95.44%	93.79%
PT	95.50%	95.33%	94.89%
IBRID	97.20%	97.42%	96.81%
STRING	96.37%	92.20%	92.20%

SVM	SST-5	SST-2
Model1	40.04%	80.61%
Model2	49.50%	91.32%
Model3	52.44%	92.42%

Model	SST-5	SST-2
RNTN	45.70%	85.40%
LSTM	56.20%	#
RoBERTa	59.10%	97.50%

Considerations

- Both the traditional SVM model and the Tree kernels outperform the base models proposed for SST-5 and SST-2 respectively.
- The traditional SVM model does not perform as well as the newest models that use LSTM or TRANSFORMER
- The Tree kernels instead are just shy of .08% in accuracy with respect to the current best model proposed
- Given that SVM-LIGHT-TK does not support multiclass classification natively it could be interesting to develop an one-vs-one or one-vs-rest approach