CS512: Sentiment Analysis On Tweets

Aming, Ni
Presentation Outline

- Short Introduction
- Related Works
- Problem Formulation
- Data Collection and Generation
- Feature Extraction
- Methods and Classifiers
- Evaluation Metrics
- Main Results
- Additional Results (If time permitted)
- Conclusion and Possible Extensions
Short Introduction

1. Sentiment Analysis On Tweets
   a. Ill defined problem: different opinions from different people
   b. Traditional Method: identifying sentiment expressions + determining sentiment types
   c. Noisy data:
      i. More casual: suffer from spelling mistakes, grammar errors, slangs
      ii. More ambiguous: limited to 140 (or 280) characters in length, harder to label

2. Sentiment Analysis Competition:
   a. SemEval: Ongoing series of evaluations on computational semantic analysis systems over the past 11 years
   b. Task 4 is to evaluate Twitter Sentiment Analysis Systems, and has five subtasks.
   c. Use 2017 SemEval as our baseline
      i. Had a total of 48 teams participating on Task 4.
      ii. We focus on subtask A, B and C for English text only.
Related Works

1. Previous Works Before SemEval-2017
   a. Datasets: Include using manually labeled datasets, automatically labeled datasets, or mixed of both
   b. Preprocessing:
      i. Replacing a set of words with equivalence class tokens
      ii. Removing irrelevant data like stopwords and retweets
      iii. Stemming, Lowering, Tokenization...
   c. Feature Extraction:
      i. Common techniques: Lexical analysis, syntactic analysis and semantic analysis
      ii. Examples: ngrams, part of speech tagging, entity typing.
   d. Classifiers: SVM, Naive Bayes, Maximum Entropy, Conditional Random Field...
1. Major Techniques in SemEval-2017
   a. 20 teams used deep learning and neural network methods
   b. A few use SVM with various embedding methods
   c. Others: Maximum Entropy, Logistic Regression, Random Forest, Naive Bayes...
   d. Most common external datasets: sentiment140 for word embedding, or Twitter API to download more tweets.

2. Top two performing teams:

<table>
<thead>
<tr>
<th>Team</th>
<th>Techniques &amp; Ideas</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB_twtr</td>
<td>(1) url to “url” (2) emotions to “smile”, “sadness” (3) replaced words like “soooo” with “soo” (4) lowercased (5) trained embedding on 100 million unlabeled tweets using Twitter API (6) Ensemble: CNNs + LSTMs</td>
</tr>
<tr>
<td>DataStories</td>
<td>(1) Task A: LSTM+Att (2) Task B and C: BiLSTM + Concat+Att+context</td>
</tr>
</tbody>
</table>

Table 3: Main Ideas and Techniques in Top 2 Teams
Problem Formulation

1. Formulate all three subtasks in SemEval-2017 Task as three multiclass classification problems.
2. Task A: Given a tweet, decide whether it expresses POSITIVE, NEUTRAL, or NEGATIVE sentiment.
3. Task B: Given a tweet and a topic, decide whether it expresses POSITIVE, or NEGATIVE sentiment.
4. Task C: Given a tweet and a topic, decide whether it expresses STRONGLY POSITIVE, POSITIVE, NEUTRAL, NEGATIVE, or STRONGLY NEGATIVE sentiment.
1. **Original SemEval-2017 datasets:**
   a. Training dataset: are mixed of (1) newly collected datasets over the period of September-November 2016 (2) datasets from previous SemEval Competitions.
   c. Topics for test datasets are different from those in the training datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Subtask</th>
<th>Topics</th>
<th>Positive 2</th>
<th>Neutral 0</th>
<th>Negative -1</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>A</td>
<td>N/A</td>
<td>19902</td>
<td>22591</td>
<td>7840</td>
<td>50333</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>373</td>
<td>14951</td>
<td>1544</td>
<td>4013</td>
<td>20508</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>200</td>
<td>1020</td>
<td>12922</td>
<td>12993</td>
<td>30632</td>
</tr>
<tr>
<td>Test</td>
<td>A</td>
<td>N/A</td>
<td>2375</td>
<td>5937</td>
<td>3972</td>
<td>12284</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>125</td>
<td>2463</td>
<td>0</td>
<td>3722</td>
<td>6185</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>125</td>
<td>131</td>
<td>2332</td>
<td>6194</td>
<td>12379</td>
</tr>
</tbody>
</table>

Table 4: Statistics on the English training and testing datasets.
1. **Sentiment140 dataset**
   a. mainly for learning word embedding
   b. Has about 1.6 million tweets including only POSITIVE and NEGATIVE labels
   c. Is automatically labeled by using emoticon heuristics.
      i. Example: Happy Emoticon means Positive

2. **Sander dataset:**
   a. Has POSITIVE, NEUTRAL, and NEGATIVE labels over four Topics.
   b. Randomly sample STRONGLY POSITIVE/NEGATIVE from POSITIVE/NEGATIVE using uniform distribution.

<table>
<thead>
<tr>
<th>Topic</th>
<th># Positive</th>
<th># Neutral</th>
<th># Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>191</td>
<td>581</td>
<td>377</td>
</tr>
<tr>
<td>Google</td>
<td>218</td>
<td>604</td>
<td>61</td>
</tr>
<tr>
<td>Microsoft</td>
<td>93</td>
<td>671</td>
<td>138</td>
</tr>
<tr>
<td>Twitter</td>
<td>68</td>
<td>647</td>
<td>78</td>
</tr>
</tbody>
</table>

Table 6: Statistics For Sander’s dataset.
Data Collection and Generation Cont..

1. Syntactic Dataset Using Markov Chain Model
   a. Available labeled tweets are not sufficient to train complex models.
   b. Augmentation: (1) SemEval-2017 training datasets with Sander’s dataset, (2) Randomly add/replace positive/negative words from word lists to generate new sentences. (3) Train a markov chain model on the combined datasets and generate syntactic tweets.

<table>
<thead>
<tr>
<th>Subtask</th>
<th>Positive</th>
<th>Neutral</th>
<th>Negative</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>Task A</td>
<td>443588</td>
<td>442290</td>
<td>363684</td>
<td>1.25M</td>
</tr>
<tr>
<td>Task B</td>
<td>414690</td>
<td>6982</td>
<td>265197</td>
<td>0.69M</td>
</tr>
<tr>
<td>Task C</td>
<td>79550</td>
<td>376186</td>
<td>368460</td>
<td>241288</td>
</tr>
</tbody>
</table>

Table 5: Statistics For Syntactic datasets.
1. Some of the syntactic tweets from Task A positive pool.

<table>
<thead>
<tr>
<th>holy shit what a good place to work</th>
</tr>
</thead>
<tbody>
<tr>
<td>holy shit what a good time. i like that!</td>
</tr>
<tr>
<td>holy shit what a good idea for a linemans feast! at dunkin tomorrow!</td>
</tr>
<tr>
<td>holy shit what a good 1st-party launch lineup. yay tomorrow i’ll be out of work before rugby filled weekend. hurry up saturday!</td>
</tr>
<tr>
<td>holy shit what a good weekend. if you’re belieber!.. url @premiereworld could you play nirvana full blast in the movies sept 9.</td>
</tr>
<tr>
<td>holy shit what a good time..happy independence day #jamaica c’mon rita i know you’re gonna go out tomorrow morning. wisconsin’s going red!</td>
</tr>
</tbody>
</table>

Table 7: An example of syntactic tweets on task A positive pool
Feature Extraction

1. Preprocessing step is very similar to the top ranker BB_twtr:
   a. Lowercase all tweets
   b. Replace url to “url”
   c. Replace words like “sooooo” with “soo”
   d. Replace positive emoticons like “:)”, “:-)”, “:D” with the word “POSITIVE”
   e. Replace negative emoticons like “:(“, “:-(“, “=(-“ with the word “NEGATIVE”
   f. ...

2. Train word embedding on the combined corpus of sentiment140, sander, syntactic tweets and given training tweets.
Methods and Classifiers

1. Use SVM with average word embedding to represent sentence embedding as another simple baseline. (liblinear svm with square loss and l2 penalty)

2. Ensemble Method: CNN + BiLSTM
   a. For each basic classifier, we randomly sample k dimensions from the full word embedding dimensions.
   b. For prediction, we add up all the normalized probability and take argmax.
   c. Dimension sampling is similar to the feature sampling in Random Forest, and Prediction is Softmax.
Workflow

Figure 1: Ensemble Algorithm Workflow
Evaluation Metrics

1. We use the same performance metrics as in SemEval-2017
2. For Task A, the primary measure is average recall, accuracy and F1 score are used as secondary measures.
3. For Task B, the primary measure is also average recall, accuracy and F1 score are used as secondary measures.
4. For Task C, the primary measure is macro-average mean absolute error, mean absolute error is used as secondary measure.
### Main Results

**Table 8: Performance on subtask A**

<table>
<thead>
<tr>
<th>Method</th>
<th>AvgRec</th>
<th>$F_1^{PN}$</th>
<th>Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB_twtr</td>
<td>0.681</td>
<td>0.677</td>
<td>0.651</td>
</tr>
<tr>
<td>DataStories</td>
<td>0.681</td>
<td>0.685</td>
<td>0.658</td>
</tr>
<tr>
<td>SVM</td>
<td>0.652</td>
<td>0.597</td>
<td>0.603</td>
</tr>
<tr>
<td>Our Method</td>
<td>0.674</td>
<td>0.644</td>
<td>0.667</td>
</tr>
</tbody>
</table>

**Table 9: Performance on subtask B**

<table>
<thead>
<tr>
<th>Method</th>
<th>AvgRec</th>
<th>$F_1$</th>
<th>Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB_twtr</td>
<td>0.882</td>
<td>0.890</td>
<td>0.897</td>
</tr>
<tr>
<td>DataStories</td>
<td>0.856</td>
<td>0.861</td>
<td>0.869</td>
</tr>
<tr>
<td>SVM</td>
<td>0.839</td>
<td>0.846</td>
<td>0.865</td>
</tr>
<tr>
<td>Our Method</td>
<td>0.891</td>
<td>0.868</td>
<td>0.874</td>
</tr>
</tbody>
</table>

**Table 10: Performance on subtask C**

<table>
<thead>
<tr>
<th>Method</th>
<th>$MAE^M$</th>
<th>$MAE^\mu$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB_twtr</td>
<td>0.481</td>
<td>0.554</td>
</tr>
<tr>
<td>DataStories</td>
<td>0.555</td>
<td>0.543</td>
</tr>
<tr>
<td>SVM</td>
<td>0.725</td>
<td>0.750</td>
</tr>
<tr>
<td>Our Method</td>
<td>0.719</td>
<td>0.763</td>
</tr>
</tbody>
</table>
Additional Results
Does Word Embedding Help?

Figure 2: Effect of Embedding on Neural Net Models.
We set the maximum tweet length to 140, CNN kernel size to 3, batchsize to 128, and number of epoch to be 1.
Figure 3: Effect of Embedding Size on subtask A, B, and C. We set maximum tweet length to 140, CNN kernel size to 3, batchsize to 128, and run on 2 epochs.
Additional Results
Does Learning Entity Typing Help?

Figure 4: Effect of Typing on Three subtask A, B, and C.
We set maximum tweet length to 140, CNN kernel size to 3, batchsize to 128, and run on 2 epochs.
Additional Results
Does Syntactic Dataset Help?

Figure 5: Effect of Syntactic Data on Three subtasks A, B, and C.
We set maximum tweet length to 140, CNN kernel size to 3, batchsize to 128, word embedding to 150 and run on 1 epochs.
Does Ensemble With Dimension Sampling Help?

Figure 6: Effect of Ensemble on Three subtasks A, B, and C. We set maximum tweet length to 140, CNN kernel size to 3, batchsize to 128, word embedding to 120 and run on 1 epochs.
More Results
Effect of Kernel Size on CNN

Figure 7: Effect of Kernel Size in CNN.
We set word embedding size to 150, maximum tweet length to 140, stride size to 1, batch size to 128, and run on 1 epoch.
Figure 8: Effect of Kernel Size in CNN.
We set word embedding size to 150, maximum tweet length to 140, kernel size to 2, batch size to 128, and run on 1 epoch.
Figure 9: Effect of Architectures in CNN.
We set word embedding size to 150, maximum tweet length to 140, kernel size to 2, stride size to 1, batch size to 128, and run on 1 epoch.
Figure 10: Effect of # units in BiLSTM.
We set word embedding size to 150, maximum tweet length to 140, batch size to 128 and run on 1 epoch.
More Results

Effect of Batchsize on CNN and LSTM

Figure 11: Effect of batchsize in Neural Nets.
We set word embedding size to 150, maximum tweet length to 140, kernel size to 2, stride size to 1 and run on 1 epoch.
More Results

Effect of \# Epoch on CNN and LSTM

Figure 12: Effect of \# Epoch in Neural Nets. We set word embedding size to 150, maximum tweet length to 140, kernel size to 2, stride size to 1 and batch size to 128.
Conclusion and Possible Extensions

2. Syntactic tweets may only help in 2-way classification tasks.
3. Good features are more important than classifiers.
4. Possible extensions: Use a better method to generate syntactic tweets