Evaluation metrics matter: predicting sentiment from financial news headlines.

Andrew Moore and Paul Rayson

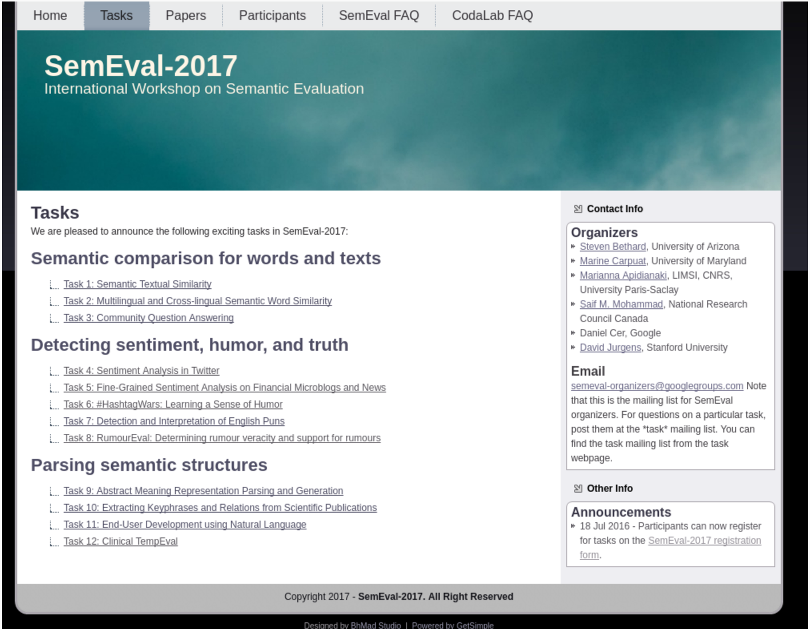
February 27, 2018

School of Computing and Communications, Lancaster University.

Table of contents

Introduction

What is SemEval



Home | Tasks | Papers | Participants | SemEval FAQ | CodaLab FAQ

SemEval-2017

International Workshop on Semantic Evaluation

Tasks

We are pleased to announce the following exciting tasks in SemEval-2017:

Semantic comparison for words and texts

- Task 1: Semantic Textual Similarity
- Task 2: Multilingual and Cross-lingual Semantic Word Similarity
- Task 3: Community Question Answering

Detecting sentiment, humor, and truth

- Task 4: Sentiment Analysis in Twitter
- Task 5: Fine-Grained Sentiment Analysis on Financial Microblogs and News
- Task 6: #HashtagWars: Learning a Sense of Humor
- Task 7: Detection and Interpretation of English Puns
- Task 8: RumourEval: Determining rumour veracity and support for rumours

Parsing semantic structures

- Task 9: Abstract Meaning Representation Parsing and Generation
- Task 10: Extracting Keyphrases and Relations from Scientific Publications
- Task 11: End-User Development using Natural Language
- Task 12: Clinical TempEval

Contact Info

Organizers

- Steven Bethard, University of Arizona
- Marine Carpuat, University of Maryland
- Marianna Apidianaki, LIMSI, CNRS, University Paris-Saclay
- Saif M. Mohammad, National Research Council Canada
- Daniel Cer, Google
- David Jurgens, Stanford University

Email

semeval-organizers@googlegroups.com Note that this is the mailing list for SemEval organizers. For questions on a particular task, post them at the *task* mailing list. You can find the task mailing list from the task webpage.

Other Info

Announcements

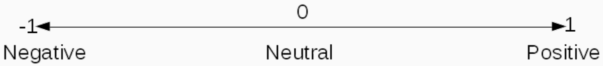
- 18 Jul 2016 - Participants can now register for tasks on the SemEval-2017 registration form.

The task

Example sentence

'Why AstraZeneca plc & Dixons Carphone PLC Are Red-Hot Growth Stars!'

Sentiment scale



Data

Training data: 1142 samples, 960 headlines/sentences.
Testing data: 491 samples, 461 headlines/sentences.

Evaluation metric

Cosine Similarity (CS) [1]

$$\frac{\sum_{i=1}^{K} A_i B_i}{\sqrt{\sum_{i=1}^{K} A_i^2}\sqrt{\sum_{i=1}^{K} B_i^2}} \tag{1}$$

Example

A = Predicted sentiment = [0.5, -0.2]
B = True sentiment = [0.4, 0.1]
Cosine similarity = 0.189

[1] Taken from Wikipedia `https://en.wikipedia.org/wiki/Cosine_similarity`

Approach

Additional data used

Word2Vec model

Used 189, 206 financial articles (e.g. Financial Times) that were manually downloaded from Factiva[2] to create a Word2Vec model [5][3].

These were created using Gensim[4].

[2] https://global.factiva.com/factivalogin/login.asp?productname=global

[3] https://github.com/apmoore1/semeval/tree/master/models/word2vec_models

[4] https://radimrehurek.com/gensim/models/word2vec.html

Support Vector Regression (SVR) [2]

Features and settings that we changed

1. Tokenisation - Whitespace or Unitok[5]
2. N-grams - uni-grams, bi-grams and both.
3. SVR settings - penalty parameter C and epsilon parameter.
4. Target aspect.
5. Word Replacements.

[5] `http://corpus.tools/wiki/Unitok`

Word Replacements

Example Sentence

'AstraZeneca PLC had an improved performance where as Dixons performed poorly'

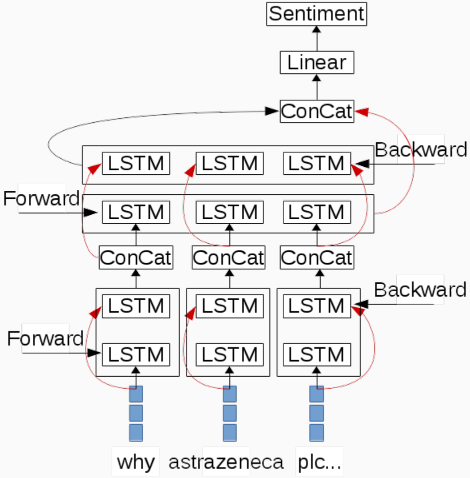
'companyname had an posword performance where as companyname performed negword'

Word Replacements

Company example N=10 company = 'tesco'

sainsbury	0.6729	**primark**	0.4811
asda	0.5999	**grocer**	0.4792
morrisons	0.5188	**unilever**	0.4764
supermarkets	0.5089	**wal-mart**	0.4750
kingfisher	0.4956	**waitrose**	0.4713

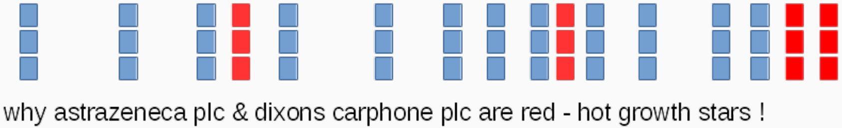
Bi-directional Long Short-Term Memory BLSTM [3][4]



BLSTM Sentence representation

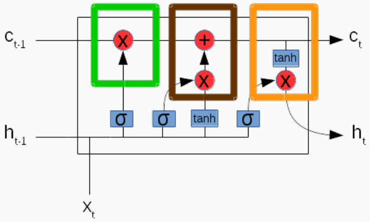
1. Sentences are fixed length.
2. All words are represented as vectors.

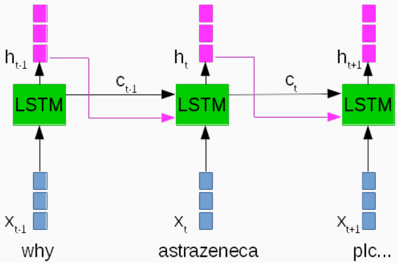
Example



BLSTM LSTM network[6]

LSTM network





Properties

1. **Forgot gate.**
2. **Input gate.**
3. **Output gate.**

[6] Image idea taken from: `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

BLSTM LSTM network

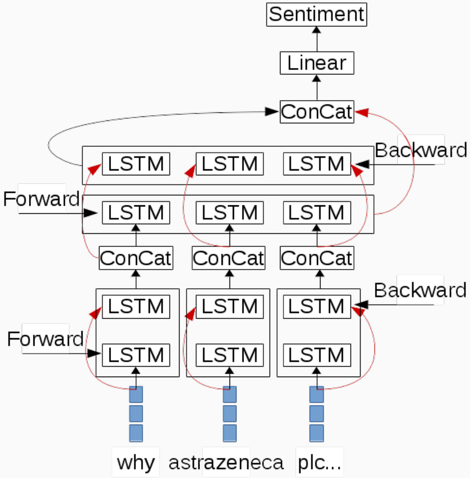
The advantages of LSTMs

1. Good at learning sequential data.
2. Able to learn long term dependencies.

LSTM related work

1. Google have improved their translation system using LSTMs[7]
2. Chiu and Nichols improved Named Entity Recognition[1].

BLSTM architecture explained

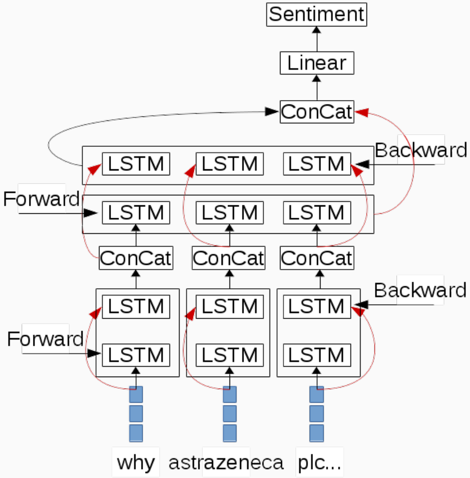


Loss function

Mean Square Error (MSE)

$$\frac{1}{Y}\sum_{i=1}^{Y}(\hat{y}_i - y)^2 \qquad (2)$$

Two BLSTM models



Standard Model (SLSTM)

- Drop out between layers and connections.
- 25 times trained over the data (epoch of 25).

Early stopping model (ELSTM)

- Drop out between layers only.
- Early stopping used to determine the epoch.

Findings and Results

SVR best features

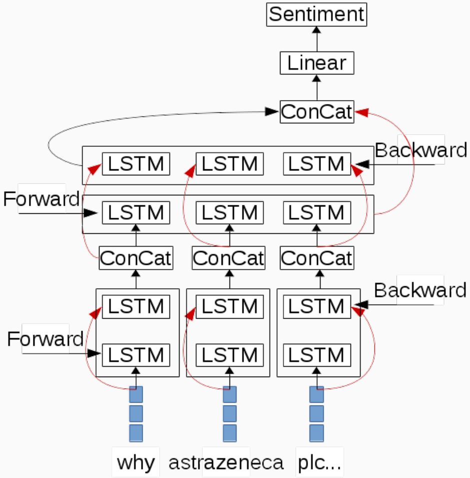
Features

- Using uni-grams and bi-grams to be the best.
- Using a tokeniser always better. Affects bi-gram results the most.
- SVR parameter settings important 8% difference between using C=0.1 and C=0.01.
- Incorporating the target aspect increased performance.
- Using all word replacements. N=10 for pos and neg words and N=0 for company.

Results

SVR	SLSTM	ELSTM
60.21%	73.20%	73.27%

Future Work



1. Incorporate aspects into the BLSTM's shown to be useful by Wang et al. [6].
2. Improve BLSTM's by using an attention model Wang et al. [6].

Future Work

1. Incorporate aspects into the BLSTM's shown to be useful by Wang et al. [6].
2. Improve BLSTM's by using an attention model Wang et al. [6].

Why evaluation metrics matter

The task

‘given a text instance predict the sentiment score for each of the companies/stocks mentioned’[7]

[7] http://alt.qcri.org/semeval2017/task5/

The three different metrics

Cosine Similarity (CS) Metric 1

$$\frac{\sum_{i=1}^{K} y_i \hat{y}_i}{\sqrt{\sum_{i=1}^{K} y_i^2}\sqrt{\sum_{i=1}^{K} \hat{y}_i^2}} \tag{3}$$

Metric 2

$$\frac{\sum_{n=1}^{N} \mathrm{CS}(\hat{y}_n, y_n)}{N} \tag{4}$$

Metric 3

$$\frac{\sum_{n=1}^{N} \begin{cases} len(\hat{y}_n) * \mathrm{CS}(\hat{y}_n, y_n), & \text{if } len(\hat{y}_n) > 1 \\ 1 - |y - \hat{y}_n|, & \text{if } \frac{\hat{y}_n}{y} \geq 0 \end{cases}}{K} \tag{5}$$

K = Total number of samples.

N = Total number of sentences.

The differences in metrics[8]

PS	TS	Metric 1	Metric 2	Metric 3	No. Sentences
[[0.2],[0.5]]	[[-0.4],[-0.1]]	-0.585	-1	0	2
[[0.9],[0.2]]	[[0.8],[0.3]]	0.99	1	0.9	2
[[0.2, 0.3]]	[[-0.1, -0.2]]	-0.992	-0.496	-0.992	1

PS = Predicted Sentiment
TS = True Sentiment

All of the above are two samples.

[8] Code for this slide `https://github.com/apmoore1/semeval/blob/master/examples/metric_examples.py`

Different metrics different results [9]

	Metric		
Model	1	2	3
SVR	62.14	54.59	62.34
SLSTM	72.89	61.55	68.64
ELSTM	73.20	61.98	69.24

[9] code this slide `https://github.com/apmoore1/semeval/blob/master/examples/run.py`

Metrics should reflect the problem

Problem

To identify ‘bullish (optimistic; believing that the stock price will increase) and bearish (pessimistic; believing that the stock price will decline) sentiment associated with companies and stocks.’[10]

Main reason against metric 1

That scores with opposite sentiment should not be rewarded in any way.

[10] http://alt.qcri.org/semeval2017/task5/

Recomended blog posts for word vectors

1. https://colah.github.io/posts/2014-07-NLP-RNNs-Representations/
2. http://sebastianruder.com/word-embeddings-1/

Recomended blog posts for RNN/LSTM

1. https://deeplearning4j.org/lstm - Good place to start.
2. https://colah.github.io/posts/2015-08-Understanding-LSTMs/ - Good place to understand LSTM.
3. https://karpathy.github.io/2015/05/21/rnn-effectiveness/ on the applications of RNN's.
4. https://skillsmatter.com/skillscasts/6611-visualizing-and-understanding-recurrent-networks video on RNN's.[11]
5. https://nbviewer.ipython.org/gist/yoavg/d76121dfde2618422139 usefulness of RNN's.

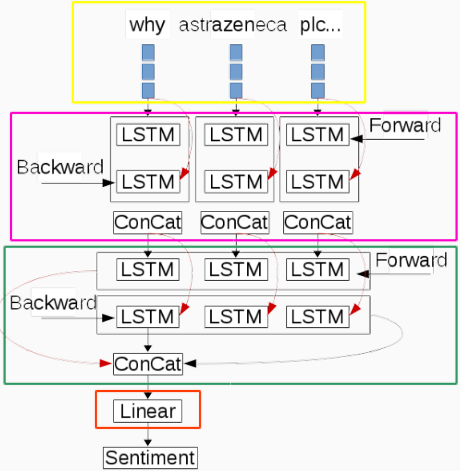
[11] 14.44 mins tips on how to train RNN/LSTM architectures.

Other related resources

1. Recommended book - `http://www.deeplearningbook.org/`
2. Oxford Deep learning course - `https://github.com/oxford-cs-deepnlp-2017/lectures`
3. Stanford courses
    3.1 Machine Learning - CS229
    3.2 NLP with deep learning - CS224n
    3.3 CNN for visual recognition - CS231n

Drawings to code

```
max_length    = self._set_max_length(train_texts)
vector_length = self._word2vec_model.vector_size

train_vectors = self._text2vector(train_texts)

model = Sequential()
model.add(Dropout(0.5, input_shape=(max_length, vector_length)))
# Output of this layer is of max_length by max_length * 2 dimension
# instead of max_length, vector_length
model.add(Bidirectional(LSTM(max_length, activation='softsign',
                             return_sequences=True)))
model.add(Dropout(0.5))
model.add(Bidirectional(LSTM(max_length, activation='softsign')))
model.add(Dropout(0.5))
model.add(Dense(1))
model.add(Activation('linear'))

model.compile(loss='mse',
              optimizer='rmsprop',
              metrics=['cosine_proximity'],
              clipvalue=5)


early_stopping = EarlyStopping(monitor='val_loss', patience=10)

model.fit(train_vectors, sentiment_values, validation_split=0.1,
        callbacks=[early_stopping] , nb_epoch=100)
```



Python libraries used

1. Scikit-learn for the SVR - `http://scikit-learn.org/stable/`
2. Keras for the BLSTMs - `https://keras.io/`

Summary

1. BLSTM outperform SVRs with minimal feature engineering.
2. Define your evaluation metric with regards to your real world problem.
3. Ensure that you know your evaluation metric before creating your system.

Questions?

a.moore@lancaster.ac.uk @apmoore94

All the code can be found here[12]

Presentation can be found here [13]

[12] https://github.com/apmoore1/semeval

[13] https://github.com/apmoore1/semeval/blob/master/presentation/slides.pdf

References I

J. Chiu and E. Nichols.
Named entity recognition with bidirectional lstm-cnns.
Transactions of the Association of Computational Linguistics, 4:357–370, 2016.

H. Drucker, C. J. Burges, L. Kaufman, A. Smola, V. Vapnik, et al.
Support vector regression machines.
Advances in neural information processing systems, 9:155–161, 1997.

A. Graves and J. Schmidhuber.
Framewise phoneme classification with bidirectional lstm and other neural network architectures.
Neural Networks, 18(5):602–610, 2005.

References II

S. Hochreiter and J. Schmidhuber.
Long short-term memory.
Neural computation, 9(8):1735–1780, 1997.

T. Mikolov, K. Chen, G. Corrado, and J. Dean.
Efficient estimation of word representations in vector space.
arXiv preprint arXiv:1301.3781, 2013.

Y. Wang, M. Huang, x. zhu, and L. Zhao.
Attention-based lstm for aspect-level sentiment classification.
In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615. Association for Computational Linguistics, 2016.

References III

Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al.
Google's neural machine translation system: Bridging the gap between human and machine translation.
arXiv preprint arXiv:1609.08144, 2016.