bunji at SemEval-2016 Task 5: Neural and Syntactic Models of Entity-Attribute Relationship for Aspect-based Sentiment Analysis

Toshihiko Yanase, Kohsuke Yanai, Misa Sato, Toshinori Miyoshi, Yoshiki Niwa
Research & Development Group, Hitachi, Ltd.
{toshihiko.yanase.gm, kohsuke.yanai.cs, misa.sato.mw, toshinori.miyoshi.pd, yoshiki.niwa.tx}@hitachi.com

Abstract

This paper describes a sentiment analysis system developed by the bunji team in SemEval-2016 Task 5. In this task, we estimate the sentimental polarity of a given entity-attribute (E#A) pair in a sentence. Our approach is to estimate the relationship between target entities and sentimental expressions. We use two different methods to estimate the relationship. The first one is based on a neural attention model that learns relations between tokens and E#A pairs through backpropagation. The second one is based on a rule-based system that examines several verb-centric relations related to E#A pairs. We confirmed the effectiveness of the proposed methods in a target estimation task and a polarity estimation task in the restaurant domain, while our overall ranks were modest.

1 Introduction

Sentiment analysis is an important technology for understanding users' intentions from review texts. Such technologies are also useful for argumentation mining because it is necessary for readers to capture targets of interest and their polarities (Sato et al., 2015). Shared tasks of aspect-based sentiment analysis (ABSA) in SemEval provide a test bed for fine-grained analysis of sentiment polarities (Pontiki et al., 2014; Pontiki et al., 2015; Pontiki et al., 2016).

We participate in all four slots (Slot 1, 2, 1 & 2, and 3) of the restaurant domain and laptop domain in English. We focus on two types of models to capture the entity-attribute relationship, especially in Slot 2 and Slot 3. The first one is a neural network based model. The second one is a rule-based approach.

Now, we explain the problem settings of the slots and our approaches. The following is an example of sentences that provide positive opinions to the FOOD#QUALITY aspect: Pizza here is good.

Slot 1 is an extraction of all aspects mentioned in a sentence. In this example, the goal is to choose FOOD#QUALITY among many other aspects. We formulate the problem as a multi-label classification problem and use a neural network-based model. Slot 2 is an extraction of opinion target expressions. The expected output is "Pizza" in the above example. We use a pattern matching based approach and focus on gathering resources such as dictionaries. For Slot 1 & 2, we simply combine the prediction results of Slot 1 and Slot 2. Slot 3 is an estimation of sentiment polarities. In this example, we estimate the polarity of this sentence from the aspect of FOOD#QUALITY. For Slot 3, we take two approaches. The first approach is a neural attention model (Luong et al., 2015) that considers the entity attention (FOOD) and attribute attention (QUALITY) of each token. The second approach is a pattern matching-based model that examines the relationship between "Pizza" and "good" that is also used in Slot 2.

The remainder of this paper is structured as follows: In Section 2, we describe our system of phase A. In Section 3, we explain our system of phase B. Finally, Section 4 summarizes our work.

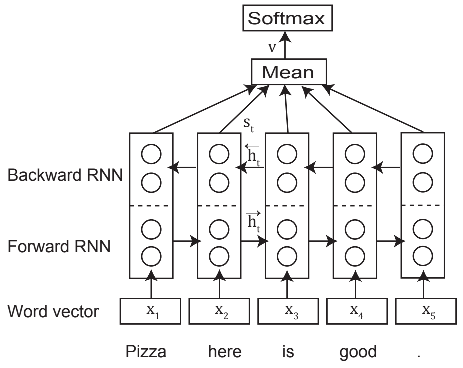

Figure 1: Structure of a neural network for Slot 1.

2 System Description of Phase A

2.1 Slot 1: Aspect Category

We formulate Slot 1 as a multi-label classification problem. In this problem, an entity-attribute pair is considered as a label. We use a neural model to solve this problem. The model is illustrated in Figure 1. Given a sequence of word vectors $X = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)$, this model calculates a vector $\mathbf{y}$ whose element represents probability of each label as:

$$\mathbf{y} = f(X). \tag{1}$$

At first, we apply Stanford Core NLP (Manning et al., 2014) to each document to obtain word sequences. Then, we use word embedding generated by Skip-gram with Negative Sampling (Mikolov et al., 2013) to convert words into word vectors. Three hundred dimensional vectors trained with Google News Corpus [1] are used in Slot 1.

Then, a word vector sequence X is inputted to a recurrent neural network (RNN). The RNN calculates an output vector $\mathbf{s}_t$ for each $\mathbf{x}_t$ as:

$$\mathbf{s}_t, \mathbf{h}_t = g(\mathbf{x}_t, \mathbf{h}_{t-1}), \tag{2}$$

where $\mathbf{h}_t$ denotes a hidden state of the RNN at position t. We use Long Short-Term Memory (LSTM) (Sak et al., 2014) and Gated Recurrent Unit (GRU) (Cho et al., 2014) as implementations of RNN units.

We use a bi-directional RNN (BiRNN) (Schuster and Paliwal, 1997) in order to consider both forward context and backward context. A BiRNN consists of a forward RNN that processes tokens from head to tail and a backward RNN that processes tokens from tail to head. We concatenate forward output $\overrightarrow{\mathbf{s}_t}$ and backward output $\overleftarrow{\mathbf{s}_t}$ into $\mathbf{s}_t$.

The sentence vector $\mathbf{v}$ is then computed as a mean of RNN outputs $\mathbf{s}_t$:

$$\mathbf{v} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{s}_t. \tag{3}$$

Finally, the probabilities in $\mathbf{y}$ are calculated by using a single layered perceptron:

$$\mathbf{y} = \mathrm{softmax}(\tanh(W\mathbf{v} + \mathbf{b})), \tag{4}$$

where W, $\mathbf{b}$ denote a weight matrix and a bias vector, respectively.

We determine that a sentence contains the i-th aspect if its output y_i is greater than a threshold θ. The threshold θ is determined by using development data that is randomly sampled from training data.

We modify aspect names to a suitable format for our neural model. Low-frequency aspects in training datasets are replaced by a new aspect "OTHER". The most common 10 aspects are preserved in the restaurant domain; the most common 16 aspects are preserved in the laptop domain. "NONE" labels are assigned to sentences that do not have any labels. The probability $\mathbf{y_i}$ in an example of a training dataset is defined as $y_i = 1/k$ when a target sentence has the i-th aspect and a total of k aspects, otherwise $y_i = 0$.

We train the model by using backpropagation. The loss is calculated by using cross entropy. We use a minibatch stochastic gradient descent (SGD) algorithm together with an AdaGrad optimizer (Duchi et al., 2011). We add Dropout (Srivastava et al., 2014) layers to the input and output of the RNN. We clip the gradient norm when it exceeds 5.0 to improve the stability of training. The model parameters and θ are trained by the training dataset of the ABSA 2015, and the hyperparameters are tuned by test dataset of the ABSA 2015. We use random sampling to tune the hyperparameters. The best settings are shown in Table 1. We implement our neural systems by using Tensorflow (Abadi et al., 2015).

2.2 Slot 2: Opinion Target Expression

In Slot 2, we extract text spans corresponding to target entities. The procedure of our proposed method

[1] Word embedding is available at https://code.google.com/archive/p/word2vec/

Parameter	REST	LAPT
Dropout p_k	0.8	0.3
Learning rate	0.945	0.374
hidden unit size	128	64
minibatch size	20	20
max epochs	100	840
cell	LSTM	GRU
max token num	40	50
threshold	0.048	0.11

Table 1: Hyperparameter setting for Slot 1. p_k denotes a ratio to keep values in a Dropout layer.

is as follows:

1. Creating dictionaries of food names and drink names by extracting targets in a training dataset,
2. Collecting food names and drink names in Knowledge Base and adding them to dictionaries,
3. Applying dictionary matching to sentences in a test dataset,
4. Extracting restaurant names by using syntactic rules, and
5. Checking relationship between targets extracted by step 3 and step 4 and attribute expressions.

Three key features of our method are the dictionary creation in step 2, the syntactic rules in step 4, and the estimation of the entity-attribute relationship in step 5.

Dictionary Creation

Coverage of dictionaries is crucial to improve recall metrics. In the training dataset, we observe various instances of FOOD entities such as bread, focaccia and gazpacho. Therefore, we try to import world knowledge written in Knowledge Base. We use DBpedia [2] as Knowledge Base to expand the dictionaries. We write a SPARQL query to retrieve labels (rdfs:label) of entities as dictionary entries. First we prepare a list of target types. For examples, we use http://dbpedia.org/ontology/Food and http://dbpedia.org/ontology/Fish as types of FOOD entities. We also prepare a list of types to be ignored such as “dbo:Beverage”. Names of DRINK are also retrieved in the same manner as FOOD.

[2]http://wiki.dbpedia.org/

Domain	Team	Precision	Recall	F1
REST	bunji	48.86	78.20	60.14
	baseline	54.19	67.03	59.93
	Ranked 1st	72.45	73.62	73.03
LAPT	bunji	44.09	35.92	39.59
	baseline	45.92	31.66	37.48
	Ranked 1st	56.85	47.81	51.94

Table 2: Official results of Subtask 1 Slot 1

Domain	Team	Precision	Recall	F1
REST	bunji	62.61	67.32	64.88
	baseline	51.42	38.56	44.07
	Ranked 1st	75.49	69.44	72.34

Table 3: Official results of Subtask 1 Slot 2

Restaurant Name Extraction

We use syntactic rules to extract restaurant names. We define a set of verb-centric rules such as “A1 visited A2” where A1 is a subject, and A2 is an object. A2 is likely to be restaurant names. We manually create 15 rules from training data.

Entity-Attribute Relationship Estimation

We observe entities not related to sentimental expressions in dictionary-match results, which decrease precision scores. Therefore, we filter entities related to sentiment expressions. We use the same method as that in Slot 3.

2.3 Results

Table 2 shows the results of Slot 1. Our system marked the highest recall score among all of the teams in the restaurant domain, while our precision score is lower than that of the baseline system. This is partly because of the determination of threshold values that may be overfitted to the development sets. One possible solution is to use cross validation to estimate more reliable threshold values.

Table 3 shows the results of Slot 2. We can observe improvement of both the precision score and the recall score from those of the baseline system. The recall score is comparable to that of the ranked 1st team, while there is much room for improvement of the precision scores.

Table 4 shows the results of Slot 1 & 2. We can observe the similar tendency to Slot 1’s results because we simply merged the results of Slot 1 and Slot 2, and Slot 1 performs worse than Slot 2.

Domain	Team	Precision	Recall	F1
REST	bunji	35.41	49.01	41.11
	baseline	36.56	39.12	37.80
	Ranked 1st	52.95	52.27	52.61

Table 4: Official results of Subtask 1 Slot 1 & 2

Domain	Team	Precision	Recall	F1
REST	bunji	72.71	88.37	79.78
	baseline	90.65	69.55	78.71
	Ranked 1st	87.00	81.19	83.99
LAPT	bunji	66.84	46.32	54.72
	baseline	86.55	37.86	52.68
	Ranked 1st	72.49	51.83	60.45

Table 5: Official results of Subtask 2 Slot 1

Table 5 shows the results of Slot 1 of Subtask 2. We merge the sentence-wise results into document-wise results.

3 System Description of Phase B

3.1 Slot 3: Sentiment Polarity

Neural Approach

Our method is inspired by a Deep Learning method proposed by Wang and Liu (Wang and Liu, 2015). They used estimated probabilities of Slot 1 as weights of a target entity-attribute pair, and then they inputted weighted tokens to a convolutional neural network. Instead of probabilities of Slot 1, we directly calculate entity attention and attribute attention at each token by using a neural attention model (Luong et al., 2015). The model is illustrated in Figure 2. We calculate a vector $\mathbf{y}_p$ that represents probabilities of polarities (positive, negative, and neutral) as:

$$\mathbf{y}_p = f(X, \mathbf{v}_e, \mathbf{v}_a), \tag{5}$$

where $\mathbf{v}_e$ and $\mathbf{v}_a$ denote vectors corresponding to a target entity and a target attribute. At first we calculate RNN outputs $\mathbf{s}_t$ with Eq. 2 similarly to Slot 1. Then, attention weights for both entity and attribute are computed at attention layers. An entity-attention layer calculates weights ε_t at position t. At each position, e_t is computed to measure the relationship between $\mathbf{s}_t$ and $\mathbf{v}_e$:

$$e_t = \mathbf{v}_e^{\mathrm{T}} W_e \mathbf{s}_t. \tag{6}$$

Then, we transform the scale of e_t and obtain an entity-attention weight ε_t as:

$$\varepsilon_t = \frac{\exp(e_t)}{\sum_j \exp(e_j)}. \tag{7}$$

Similarly, the attribute attention layer has weights α_t at position t as follows:

$$a_t = \mathbf{v}_a^{\mathrm{T}} W_a \mathbf{s}_t, \tag{8}$$

$$\alpha_t = \frac{\exp(a_t)}{\sum_j \exp(a_j)}. \tag{9}$$

Then, we calculate a sentence vector $\mathbf{r}$ that is a weighted sum of RNN output with entity attention weights and attribute attention weights as:

$$\mathbf{r} = \sum_t (\alpha_t \mathbf{s}_t || \varepsilon_t \mathbf{s}_t), \tag{10}$$

where $||$ denotes a concatenation operator that creates a vector in $\mathbb{R}^{2d}$ from two vectors in $\mathbb{R}^d$.

Finally, we calculate $\mathbf{y}_p$ by using a single layered perceptron:

$$\mathbf{y}_p = \mathrm{softmax}(\tanh(W_p \mathbf{r} + \mathbf{b}_p)). \tag{11}$$

We train the Slot 3 model by using backpropagation. We use a minibatch stochastic gradient descent (SGD) algorithm together with the ADAM optimizer (Kingma and Ba, 2015). Hyperparameters are tuned similarly to Slot 1. The hyperparameter settings in Slot 3 are shown in Table 6. We add Dropout (Srivastava et al., 2014) layers to the input and output of the RNN. We also apply L2-regularization to two attention layers and a softmax

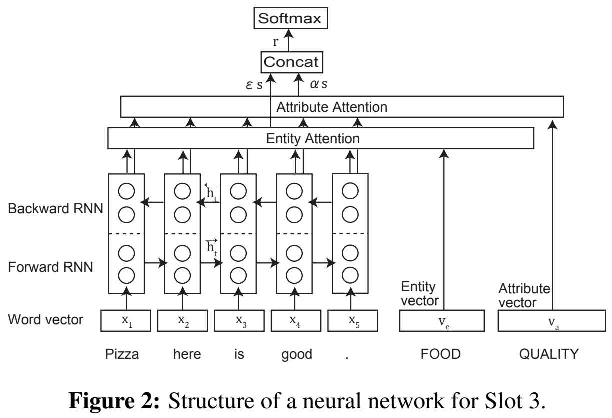

Figure 2: Structure of a neural network for Slot 3.

Parameter	REST (U)	REST (C)	LAPT (C)
Dropout p_k	0.9	0.9	0.9
Learning rate	1.7×10^{-3}	1.4×10^{-3}	5.8×10^{-4}
RNN state size	64	128	64
minibatch size	16	32	16
max epochs	12	19	6
L2 coef	1.9×10^{-4}	1.9×10^{-4}	3.3×10^{-4}

Table 6: Hyperparameter setting for Slot 3. p_k denotes a ratio to keep values in a Dropout layer.

layer. The attention unit size is 300. In an unconstrained setting, we use the same word embedding that has 300 dimensional vectors as Slot 1. In a constrained setting, we use 128-dimensional vectors that are initialized by uniform distribution. We clip the gradient norm when it exceeds 5.0 to improve stability of training. We set the maximum token length as 40. Initial values of entity vectors are created by averaging word vectors in sentences that have target entities. Attribute vectors are also initialized in the same manner as the entity vectors.

Relation-Features Approach

This approach trains a linear classifier using relations of a given entity and a given attribute as features. In the first step, we annotate the following 11 annotations of relations to all documents:

believe Showing someone's belief such as "X likes Y" and "X avoids Y",

significant Showing X's significance such as "X is impressive" and "X is terrible",

require Showing requirement such as "X needs Y",

equivalent Showing X is equivalent to Y, such as "X viewed Y" and "X regarded Y",

include Showing inclusion or possession such as "X has Y" and "X equips Y".

contrast Comparing X with Y such as "Y is ... than X" and "Y is ... compared to X",

affect Showing X affects Y such as "X increases Y" and "X causes Y",

state Showing statement such as "X doubts that Y",

negation Showing negations such as "not X" and "no X",

shift Reversing X's polarity such as "X ban" and "X shortage", and

absolutize Fixing polarity of X such as "X problem" and "X risk".

These annotations were originally developed for an argument-generation system (Sato et al., 2015).

In the second step, we identify an entity expression and an attribute expression that correspond

Domain	Team	C/U	Accuracy
REST	bunji(neural)	U	81.02
	bunji(neural)	C	76.25
	baseline	C	76.48
	Ranked 1st	U	88.13
LAPT	bunji(rel)	U	70.16
	bunji(neural)	C	70.29
	baseline	C	70.04
	Ranked 1st	U	82.77

Table 7: Official results of Subtask 1 Slot 3. (neural) and (rel) denote the neural approach and the relation-feature approach, respectively.

to a given entity-attribute pair. We use a simple dictionary-matching approach. In the restaurant domain, we use a given target annotation as an entity expression. In the laptop domain, we prepare a list of entities extracted from a training dataset. For both domains, we create an attribute dictionary. Entries of the dictionary are manually extracted from training datasets. Then, we assign a sentimental polarity (positive, negative, or neutral) to each entry.

In the third step, we create features for a linear classifier. Those features are generated by combining annotations to capture various relations of a target entity-attribute pair. For example, we examine whether an affect annotation is negated or not and whether a target entity is a subject of an affect annotation or an object.

Finally, we classify a sentimental polarity by using a linear classifier. We use a linear SVM in scikit-learn (Pedregosa et al., 2011) as an implementation of the classifier, on the parameter C = 0.1, loss = squared_epsilon_insensitive, and penalty = l2.

3.2 Results

Table 7 shows the results for Slot 3. We select a suitable method from the neural method and the rule-based method for each domain by comparing scores in the ABSA15 dataset. In the restaurant domain, we can observe that the proposed method improves the accuracy by 10 percentage points compared with the baseline system.

We merged sentence-wise estimation and created document-wise estimation. We gathered polarities of an entity-attribute pair. If a result was both positive and negative, then we judged it as conflicting. Table 8 shows the results for Subtask 2. We can see

Domain	Team	C/U	Accuracy
REST	bunji(neural)	U	70.54
	bunji(neural)	C	66.58
	baseline	C	74.26
	Ranked 1st	U	81.93
LAPT	bunji(rel)	U	60.00
	bunji(neural)	C	62.20
	baseline	C	73.03
	Ranked 1st	U	75.05

Table 8: Official results of Subtask 2 Slot 3. (neural) and (rel) denote the neural approach and the relation-feature approach, respectively.

a similar tendency to the results in Subtask 1.

4 Conclusions

In this paper, we described the participation of the bunji team in SemEval-2016. We used both a neural approach and a rule-based approach to model an entity-attribute relationship. We confirmed the effectiveness of the proposed methods in a target estimation task and a polarity estimation task in the restaurant domain, while our overall ranks were modest.

As a future work, we plan to investigate network structures that are simple enough to be trained with a relatively small dataset. For the rule-based system, we plan to add more rules to improve precision scores.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *The International Conference on Learning Representations (ICLR)*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Proceedings of Workshop at International Conference on Learning Representations.*

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495.

Maria Pontiki, Dimitrios Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeny Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. 2016. SemEval-2016 task

5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16.

Hasim Sak, Andrew W. Senior, and Françoise Beaufays. 2014. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *INTERSPEECH*, abs/1402.1128:338–342.

Misa Sato, Kohsuke Yanai, Toshinori Miyoshi, Toshihiko Yanase, Makoto Iwayama, Qinghua Sun, and Yoshiki Niwa. 2015. End-to-end argument generation system in debating. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 109–114.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, 45(11):2673–2681.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Bo Wang and Min Liu. 2015. Deep learning for aspect-based sentiment analysis. *Reports for CS224d, Stanford University*.