18th International Conference on Information Fusion
Washington, DC - July 6-9, 2015

Troll Detection by Domain-Adapting Sentiment Analysis

Chun Wei Seah; Hai Leong Chieu; Kian Ming A. Chai; Loo-Nin Teow; Lee Wei Yeong
DSO National Laboratories
20 Science Park Drive, Singapore
Email: {SChunWei; CHaiLeon; CKianMin; TLooNin; YLeeWei}@dso.org.sg

***Abstract*—A *troll* is a user intent on sowing discord on the internet. We propose an approach to detect such users from the sentiment of the textual content in online forums. Since trolls typically express negative sentiments in their posts, we derive features from sentiment analysis, and use SVM^{rank} to do binary and ordinal classification of trolls. With a small labeled training set of 20 users, we achieved 60% and 58% generalized receiver operating characteristic (ROC) for binary and ordinal troll classification on our forum data respectively. In our experiments, we used features derived from a recursive neural tensor network sentiment analysis model trained on a movie reviews data set written in standard English. However, our forum data set contains messages in a wide spectrum of topics, and are often written in Colloquial Singapore English. We applied domain adaptation techniques to the sentiment analysis model using un-annotated forum data, and achieved a final result of 78% and 69% generalized ROC for binary and ordinal troll classification respectively.**

***Keywords*—Troll Detection; Domain-Adapting; Sentiment Analysis; Natural Language Processing; Data Mining; Opinion Mining**

I. Introduction

Trolls are internet users whose intent is to sow discord on social platforms, such as forums, on the internet, by posting messages to provoke others into an emotional response. In this paper, we characterise trolls to be users who post messages that are repetitive, destructive and deceptive. Since trolls in forums generally post messages that convey a negative sentiment, we leverage on existing sentiment analysis work in the literature to build a troll detection system. Sentiment analysis is a well-studied problem in the natural language processing community, often with a focus on product reviews such as movie and book reviews. To address the problem of troll detection, we derive features from a sentiment analysis model before applying SVM^{rank} [2], a supervised machine learning approach that is suitable for ordinal classification problems. With this approach, we are able to leverage on the rich sentiment analysis work and data sets, e.g., [1, 3–5], so that we do not need to annotate a large training set for troll detection.

We use a state-of-the-art sentiment analysis model trained on annotated English movie reviews [1]. This model is based on a recursive neural tensor network (RNTN) where sentences are represented as parse trees. In the RNTN, words or phrases are represented as sub-trees of the parse tree of a sentence and are projected into a semantic vector space. The semantic representation of the parse-tree is recursively computed using a tensor-based function that composes the semantic vector representations of its constituents. Figure 1 shows an example used in [1] to illustrate how the RNTN predicts the sentiment of a sentence and it's negation using compositional semantics, where the negative sentiment due to the negation is correctly propagated to the roots of the parse tree.

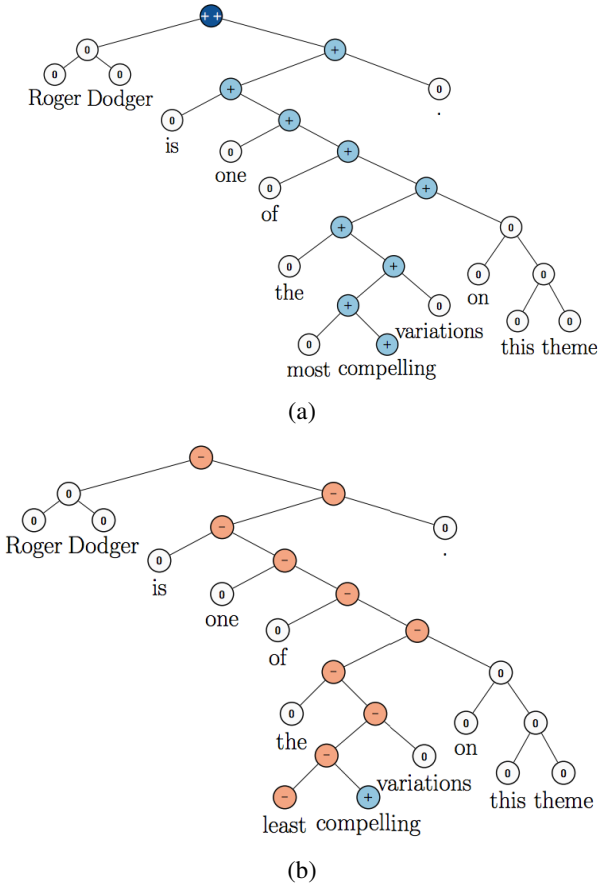

Fig. 1. A movie review sentence and its negation used in [1] to illustrate how the recurrent neural tensor network correctly predicts the sentiments using the compositional model trained on movie reviews. Each node in the parse tree is given one of five sentiments: very negative (−−), negative (−), neural (0), positive (+) and very positive (++). The final predicted sentiment of the sentence is the sentiment predicted at the root of the parse-tree. Note, the only word difference between Fig. a and Fig. b is the word "most" changed to "least", and the resulting final predicted sentiment has changed correctly from very positive to negative.

However, the RNTN was trained on an annotated movie review corpus, which was built by labelling 215,154 phrases on an ordinal scale of sentiments[1]. In the RNTN, words and phrases represented by parse-trees are projected onto a semantic vector space. In particular, the RNTN projects all out-of-vocabulary (OOV) words onto a single-vector, and the different OOV words are no longer distinguishable. The net effect is that the OOV words have little influence on the sentiment analysis predictions. This poses a problem when the model is used for a domain different from English movie reviews, especially when the domain has many OOV words that are important in conveying sentiments. Indeed, this is the case for our data set, which is collected from a Singapore forum that discusses a wide range of topics, often communicated in *Colloquial Singapore English* (CSE, also known as *Singlish*) [6] rather than standard *English*. CSE has a vocabulary that draws heavily on lexical borrowing from local languages such as Malay and Chinese dialects (e.g., Hokkien). In our forum data set, the emotionally expressive words are usually words in such borrowed lexicon. For example, the sentence "Aiyah, damn cham one" expresses a very negative sentiment, wherein the word *cham*, borrowed from Hokkien, means "pitfully disastrous, usually uttered with a sad shake of one's head"[2], and the word *aiyah* is "an exclamation mark to express consternation, despair, dismay, exasperation, etc."[3]. In Figure 2, we show that the RNTN model trained on movie reviews wrongly predicts this sentence to have a positive sentiment.

This motivates us to investigate domain-adaptation approaches to deal more effectively with OOV words. Our first approach, abbreviated as $\mathrm{AdaptW2V}$, is based on the idea of projecting the semantic space learned using the word2vec [7] software onto the sentiment analysis semantic space. However, this approach did not yield any significant improvement over the baseline, and we postulate that the reason was because word similarity learned in the word2vec space are syntactic in nature and unrelated to sentiments. For example, in the word2vec model trained on our forum data, the closest word to *good* in the word2vec space is the word *bad*, which has the opposite sentiment. Hence, we propose a second approach called $\mathrm{AdaptCo}$ that learns to project each OOV word onto the semantic space by leveraging on other known words that co-occur with these OOV words in our forum data. This second approach achieves a final result of 78% and 69% generalized ROC for binary and ordinal troll classification respectively.

We organise the remainder of this paper as follows. We first discuss related work in Section II. The troll detection task is defined in Section III, where we also describe a small troll detection data set we built for training and evaluation. Our approach to troll detection leveraging on sentiment analysis is described in Section IV, and the approach is refined with domain adaptation in Section IV-B. Section V describes the experiments and the results, and Section VI concludes the paper.

[1]The Stanford Sentiment Treebank dataset can be downloaded at http://nlp.stanford.edu/sentiment.

[2]http://www.talkingcock.com/html/lexec.php

[3]http://www.mysmu.edu/faculty/jacklee/singlish_A.htm

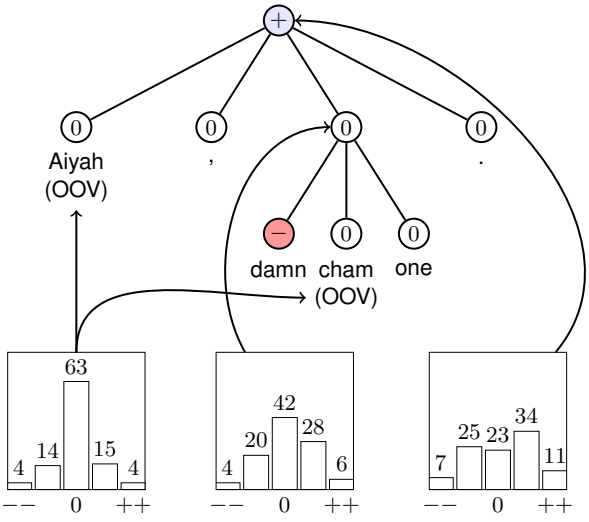

Fig. 2. The parse tree of "Aiyah , damn cham one ." annotated with sentiments by RNTN. *Aiyah* and *cham* are OOV words because they are not present in the movie review corpus. The three histograms at the bottom are the percentages of the very negative $(--)$, negative $(-)$, neutral (0), positive $(+)$ and very positive $(++)$ sentiments for four selected nodes, indicated by the arrows. This sentence expresses a negative sentiment in CSE, but it is wrongly classified to positive sentiment by RNTN.

II. Related Work

Internet trolling is a relatively recent phenomenon, and there are comparatively few works on its detection. One recent work [8] on troll detection uses supervised learning with 400 annotated reddit comments and a variety of features derived from meta-data and content such as the number of matching bad words. Another related work is the Kaggle competition [9] to detect insults in social media, where an annotated data set of 3,947 instances was provided for training. In contrast to such fine-grained annotation of posts, we do coarse-grained annotation of discussion threads, and we annotate for 40 users in 200 threads for training and evaluation purposes.

The sentiment analysis problem is a well studied problem in natural language processing, with most approaches applying machine learning techniques. Most domain adaptation works on sentiment analysis focus on predicting sentiments of documents [3, 5, 10, 11], such as movie reviews and product reviews. In this paper, we use the RNTN sentiment analysis model [1] that predicts sentiments of sentences, and we adapt the model to our forum domain by projecting the OOV words into the sentiment vector space learned by the RNTN. The advantage of the RNTN model over document-based models [3, 5, 10–12] is that it leverages on the Stanford sentiment data set which contains fine-grained sentiment annotations on words and phrases on parse trees, while other sentiment analysis machine learning approaches are typically trained on document-level annotated data. This allows the model to learn sentence constructions that humans use to express sentiments, such as by negation as shown in the examples in Figure 1.

The SemEval 2013 task [13] focuses on the sentiment analysis task on Twitter. Although the focus was Twitter, one data set used in the evaluation was an SMS data set [14],

collected by the National University of Singapore. In SemEval 2013, it has been noted that the degradation of performance from the Twitter data set to the SMS data set could be due to the "Colloquial Singaporean English (aka Singlish) found in the SMS test set" [15]. In our work, we attempt to address this by doing domain adaptation using a sentiment classifier trained on English to detect trolls in a forum where the language is often in CSE.

It is well known that machine learning systems built in one domain often performs badly in another domain. Blitzer et al. [5] investigated domain adaptation for sentiment classification, where they used structured correspondence learning (SCL) to adapt sentiment classifiers trained for one product to classify sentiments for another product. SCL works by first learning predictors of common words that occur in both domains called pivots. Using these predictors, SCL then models correlations between pivots and all other words to project features to a low dimensional space that is common across domains. However, the application of SCL still requires some training data in the target domain. In this paper, we do not have labeled data in the target domain of CSE forum data. Hence, we work under the setting of *zero-shot* domain adaptation [16]. The zero-shot domain adaptation work in [16] focused on multi-view dimension reduction, where a low dimensional feature space is learned from multiple source domains. However, the sentiment analysis model we used was trained on the Stanford Sentiment corpus, which is annotated at a phrase level on parse trees. To the best of our knowledge, there are no other sentiment analysis data sets that are annotated on parse trees that would permit us to apply a multi-view approach.

SCL can be seen as improving cross-domain machine learning performance by learning a domain-independent low dimensional projection of the feature space. Recent work on distributional semantics, notably word2vec [7], projects words into a low dimensional space with interesting properties. For example, it was observed that the closest vector to the vector *king* $-$ *man* $+$ *woman* in this space is the vector for the word *queen*. Hence, by simply applying such arithmetic operations on word2vec vectors, good performance can be achieved in an analogy recovery task where the objective is to solve analogy questions of the form "a is to a^* as b is to —" in which the nature of the relation is hidden. Levy and Goldberg [17] showed that the objective function behind the vector arithmetic used in [7] is effectively

$$\arg\max_{b^* \in V} \cos(b^*, b - a + a^*), \tag{1}$$

where cos is the cosine similarity. The cosine similarity, $\cos(a, b)$, between two vectors a and b, is defined as the dot-product between the two vectors, denoted by $< a, b >$, normalised by their l_2 norm denoted by $||.||$:

$$\cos(a, b) = \frac{< a, b >}{||a||\ ||b||}.$$

Levy and Goldberg [17] improved on the analogy recovery work using the objective function,

$$\arg\max_{b^* \in V} \frac{\cos(b^*, b)\ \cos(b^*, a^*)}{\cos(b^*, a) + 0.001}. \tag{2}$$

Our first approach for domain adaptation, called $\mathrm{AdaptW2V}$, is based on the above principle. However, this approach is not significantly different from the baseline of doing no domain adaptation. We postulate that the reason behind the failure is the fact that word similarities in the word2vec space is general and not specific to sentiments. In contrast, our second approach $\mathrm{AdaptCo}$, which leverages on co-occurrence of OOV words with known words, outperforms the baseline by 18% and 11% for binary and 6-class ordinal classification respectively.

III. Trolls

A. Task Definition

Our objective is to detect trolls in internet forums, and in this section, we define what are trolls and use our definition to annotate a small data set for troll detection on forum data. We list a few definitions of trolls in Table I. We found some inconsistencies between the various definitions. For example, Shachaf and Hara [19] defines trolls to be "intentional" in their actions, while Buckels et al. [20] defines them to have "no apparent instrumental purpose". In this work, we decide to focus on the following three attributes of trolls:

Repetitiveness	trolls send a large number of troll messages;
Destructiveness	troll messages express negative sentiments to sow discord; and
Deceptiveness	troll messages may be deceptive to achieve their objective of sowing discord.

B. Annotation Guideline

The above three attributes, while defining trolls, can be rather vague and hard to apply during annotation. In particular, judging whether a statement is deceptive requires knowing the underlying ground truth, which is not accessible to annotators generally. Hence, we gauge messages to be deceptive if they are deemed irrational or defamatory. In addition, we use *negative sentiments* as a concrete proxy for *destructive*.

In summary, for annotating our data, we use the following guideline:

> A troll message is a message that expresses negative sentiments, or irrational and defamatory opinions or messages. A troll is a person who posts a large number of such troll messages.

C. Data collection and annotation

Our data is collected from one subforum of a mainstream Singapore online forum. The data collection process begins with assembling a list of forum threads of interest. Subsequently, forum threads in this list are crawled for their posts and replies. The text content of the posts are sanitized to eliminate Javascripts, forum-specific markups and other artifacts irrelevant to this study. Altogether, there are 27,086 users, 109,934 threads and 1,534,131 posts.

To select samples for annotation, we use the open source *Hater News* (http://haternews.herokuapp.com/) to score all users who had posted in at least 5 threads and at least 5 times in each of these threads. After ranking the users by their *Hater News* scores, we selected the 15 top-ranked users, and sampled

TABLE I. SOME EXISTING DEFINITIONS OF TROLLS

Source	Description of troll or troll behaviour
Wikipedia [18]	"In Internet slang, a troll is a person who sows discord on the Internet by starting arguments or upsetting people, by posting inflammatory, extraneous, or off-topic messages in an online community (such as a newsgroup, forum, chat room, or blog) with the deliberate intent of provoking readers into an emotional response or of otherwise disrupting normal on-topic discussion."
Shachaf and Hara [19]	"Findings also suggest that trolls' behaviours are characterized as repetitive, intentional, and harmful actions that are undertaken in isolation and under hidden virtual identities, involving violations of Wikipedia policies, and consisting of destructive participation in the community."
Buckels, Trapnell and Paulhus [20]	"Online trolling is the practice of behaving in a deceptive, destructive, or disruptive manner in a social setting on the Internet with no apparent instrumental purpose."

Fig. 3. Frequency of users for different troll-scores. During to the small training data, there are no users with a troll score of 0.8.

another 25 others uniformly from the rest. We annotated these 40 users in our dataset using 5 threads for each user sampled according to the number of posts posted in each thread.

Based on our guideline, we annotated each thread of a particular user on whether those posts appeared in the thread is categorized as troll message. Then, from the 5 annotated threads of each user, we compile the troll score of that user by averaging his *troll-score* over the 5 threads. For example, if he has been annotated as troll in 4 out of 5 threads, he has a troll-score of 0.8. Figure 3 gives the histogram of troll-scores in our data.

Using the computed troll scores for all users, the troll detection task is to do ordinal troll ranking, that is, user with higher troll-score is ranked higher. The evaluation is done using ordinal classes, where higher ordinal classes have a higher troll-scores. We evaluate against three groupings of ordinal classes, with two, three and six ordinal classes. For the case of six ordinal classes, the data is used as grouped in Figure 3; for three ordinal classes, we group into scores of 0 to 0.2, 0.4 to 0.6, and 0.8 to 1; and for two ordinal classes, we group into scores of 0 to 0.4 and 0.6 to 1.

IV. TROLL DETECTION

A. Features and SVM$^{\text{rank}}$

For troll detection, we apply SVM^{rank} [2] to a sentiment analysis feature set derived from the sentiment analysis scores output by the recursive neural tensor network (RNTN) sentiment analysis software [1] implemented in Stanford NLP. The RNTN outputs sentiment scores for each input sentence by learning a representation of words and phrases in a d-dimensional semantic vector space (d was 25 in our experiments). In the rest of this paper, we will call this semantic vector space the *RNTN-space*.

Given a sentence, it is first represented as a constituent parse tree. Then, the RNTN obtains the projection of the parse tree onto its space by recursively applying tensor composition to the vector space representation of its constituents.

We first applied the RNTN without modification, using its outputs to derive the input features for SVM^{rank}. The RNTN package was trained on the Stanford Sentiment Treebank corpus, a movie review corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiments in language.

The RNTN outputs sentiment scores at a sentence level. In this paper, we proposed to classify whether a user is a troll. Hence, we need to summarise the sentence-level scores into user-level scores, and this is done in the following manner using the features listed in Table II. We first summarised the sentence scores into the post-level features. From these post-level features, we derived the thread-level features. Finally, we derived the user-level features.

In greater details, the post features of $MinP$, $MaxP$ and $AvgP$ represent the most negative, most positive and the average sentiment of a particular post. $RatEqP_i$ represents the probability of a post having an ordinal sentiment category i, where i ranges from 0 to 4, with 0 for very negative, 1 for negative, 2 for neutral, 3 for positive and 4 for very positive. $RatLeP_i$ represents the accumulative probability of a post having sentiment category from very negative to i. Similarity $RatGrP_i$ represents the accumulative probability of a post having sentiment category from i to very positive. Thread features are then generated by using the minimum, maximum, average and ratio functions on the post features. When the average function is used on $RatEqP_i$, $RatLeP_i$ and $RatGrP_i$ for a particular thread, it gives the probabilities of sentiment category i, accumulative probability sentiment category from very negative to i and accumulative probability sentiment category from i to very positive, respectively. In a similar manner, user-level features are derived from the thread-level features.

While it is reasonable to contemplate using the contents of the messages as features for troll detection, we did not do that in this paper for two reasons. Firstly, given such a small training set for troll detection, the use of content features may result in high performance simply because the trolls in the training and test data set are talking about the same topics. Secondly, we want to investigate the limits of using features derived only from sentiment analysis for troll detection, and such an approach would result in minimal overfitting of our

small training set. In fact, we found that SVM^{rank} performs reasonably well under this setup. However, the RNTN was trained on a movie review corpus, but our data is from a forum with a large amount of content in CSE. Hence, we sought to improve the troll detection results by improving the performance of the RNTN on this target domain of interest. One simple but extremely laborious way would be to rebuild a labeled sentiment analysis data set on this forum data. In this paper, we proposed a less labor-intensive approach based on domain adaptation.

B. Domain Adaptation

For each sentence s, the RNTN builds a d-dimensional feature vector $a_s \in \mathbb{R}^d$ where d is set to 25. The RNTN then assigns probabilities to the sentence for the five sentiment classes using the softmax function

$$\arg_c \max_{c\in[0,4]} \frac{\exp W_c a_s}{\sum_{i=0}^{4} \exp W_i a_s}, \tag{3}$$

where the matrices W_i were learned with the Stanford Sentiment Treebank corpus.

The feature vector a_s for each sentence s is obtained in the following way. The sentence s is first parsed using the Stanford parser [21]. We denote each leaf node as l_i where i represents the ith word for the sentence. Each word in the sentence is first projected into a d-dimensional vector in the semantic space, and the sentence vector a_s is computed by recursive tensor composition of the leaf nodes using the structure of the parse tree.

Each leaf node and intermediate node can be assigned a sentiment score using Equation (3), and the composition matrices functions are learned using the labeled trees in the Stanford Sentiment Treebank. For words that are in the vocabulary of the RNTN, they are projected onto the sentiment space, $\mathbb{R}^d$, using a lookup table learned from the training data. Words that are not in the vocabulary, the OOV words, are mapped onto the a single-vector, and they would have limited effect on the output of the classifier.

When applied to our forum data set, a large number of words are OOV (around 90% of the entire corpus), due to the colloquial nature of the language. To address the missing supports of OOV words, we propose two domain adaptation methods to project OOV words onto the semantic feature space $\mathbb{R}^d$ of the RNTN.

1) Sentiment Word2Vec Adaptation: Word2Vec [7] is an unsupervised method that learns a (task-independent) semantic representation of words in a vector space, using unannotated data. In this first approach, we trained word2vec [7] on a large unannotated corpus of our target domain data. We will call this vector space the word2vec-space. Inspired by [17], we propose to use the representation of the OOV words in the word2vec-space to project them into the RNTN-space.

Motivated by the success of cosine-similarity in the word-analogy task (see Equation 2), we project an OOV word u onto the RNTN-space by computing a weighted average of all known word vectors in the RNTN-space, with the weights given by the cosine-similarity in the word2vec-space:

$$a_u = \sum_{w\in S} \pi_w \, a_w,$$

where

$$\pi_w = \frac{\cos(v_w, v_u)}{\sum_{w'\in S} \cos(v_{w'}, v_u)},$$

and a_w (resp. v_w) is the projection of word w onto the RNTN-space (resp. the word2vec-space); and S is the set of words common to the two spaces that have cosine similarity of at least θ with the OOV word. The purpose of the threshold θ is to reduce the influence of noise from words that are very dissimilar to the OOV word.

2) Sentiment Cooccurrence Adaptation: The RNTN computes a sentence vector by recursive tensor composition of the vectors of sub-trees in the parse-tree of the input sentence. Each sub-tree in the parse tree can be viewed as an n-gram of words of the input sentence, and the RNTN projects each n-gram into its d-dimensional semantic space by tensor composition. At the root node of the parse tree of sentence s, the RNTN vector a_s is a "sentimental" summary of the words in the entire sentence, where the computation of the summary takes into account the structure of the parse tree.

In our second domain adaptation approach, we propose to learn the support of each OOV word, a_u, using co-occurrence adaptation to take into consideration how the OOV word appeared within our target data set. Denoting S_u to be the set of sentences containing the OOV word u, we compute

$$\begin{aligned} a_u &= \mathbb{E}_{s\in S_u}[a_s] \\ &\approx \frac{1}{|S_u|} \sum_{s\in S_u} a_s, \end{aligned} \tag{4}$$

where a_s is the sentiment vector of the root node of the parse tree for sentence s. Intuitively, we are computing the projection of an OOV word by averaging the representation of the sentences containing that word.

In Figure 2, we have shown the sentiment analysis tool makes a wrong prediction for the sentiment output when dealing with out-of-vocabulary words "Aiyah" and "cham". However, using our co-occurrence adaptation Equation (4) to obtain the semantic projection for OOV words, the sentence, "Aiyah , damn cham one . ", is now correctly classified to be of negative sentiment, as depicted in Figure 4.

V. Experiments

A. Experimental Setups

Using the features defined in Table II for each user, we trained SVM^{rank} [2] with a linear kernel, using 20 users, and evaluated the model on the remaining 20 users. We randomly generated 100 sets of training and testing sets without replacement and used the same sets for each method in each run. In each run, we used 5-fold cross validation to select the SVM^{rank} parameter C, the trade-off between the loss function and the margin, from among $\{2^{-10}, 2^{-9}, \ldots, 2^{9}, 2^{10}\}$. The purpose of cross validation is to get an estimated low generalization error bound. In addition, to have a useful generalization error bound that is less than one, the number of features used for training should not be greater than the number of training samples.

TABLE II. FEATURES DERIVED FROM SENTIMENT ANALYSIS OUTPUT FOR TROLL DETECTION.

Scope	Name	Description
Post	$MinP$	Minimum sentiment score among all sentences of a post divide by 4
	$MaxP$	Maximum sentiment score among all sentences of a post divide by 4
	$AvgP$	Average sentiment score among all sentences of a post divide by 4
	$RatEqP_i$	Ratio of sentences in a post that has score i, where $0 \leq i \leq 4$
	$RatLeP_i$	Ratio of sentences in a post that has score less than i, where $0 \leq i \leq 4$
	$RatGrP_i$	Ratio of sentences in a post that has score greater than or equal to i, where $0 \leq i \leq 4$
Thread	$MinXT$	Minimum among posts for X where X is done for $MinP$, $MaxP$, $AvgP$, $RatEqP_i$, $RatLeP_i$, $RatGrP_i$
	$MaxXT$	Maximum among posts for X where X is done for $MinP$, $MaxP$, $AvgP$, $RatEqP_i$, $RatLeP_i$, $RatGrP_i$
	$AvgXT$	Average among posts for X where X is done for $MinP$, $MaxP$, $AvgP$, $RatEqP_i$, $RatLeP_i$, $RatGrP_i$
	$RatEqXT_i$	Ratio of posts in a thread that has score i where X is done for $MinP$, $MaxP$, for $0 \leq i \leq 4$
	$RatLeXT_i$	Ratio of posts in a thread that has score less than i where X is done for $MinP$, $MaxP$, $AvgP$, for $0 \leq i \leq 4$
	$RatGrXT_i$	Ratio of posts in a thread that has score greater than i where X is done for $MinP$, $MaxP$, $AvgP$, for $0 \leq i \leq 4$
User	$MinXU$	Minimum among threads for X where X is done for $MinXT$, $MaxXT$, $AvgXT$, $RatEqXT_i$, $RatLeXT_i$, $RatGrXT_i$
	$MaxXU$	Maximum among threads for X where X is done for $MinXT$, $MaxXT$, $AvgXT$, $RatEqXT_i$, $RatLeXT_i$, $RatGrXT_i$
	$AvgXU$	Average among threads for X where X is done for $MinXT$, $MaxXT$, $AvgXT$, $RatEqXT_i$, $RatLeXT_i$, $RatGrXT_i$

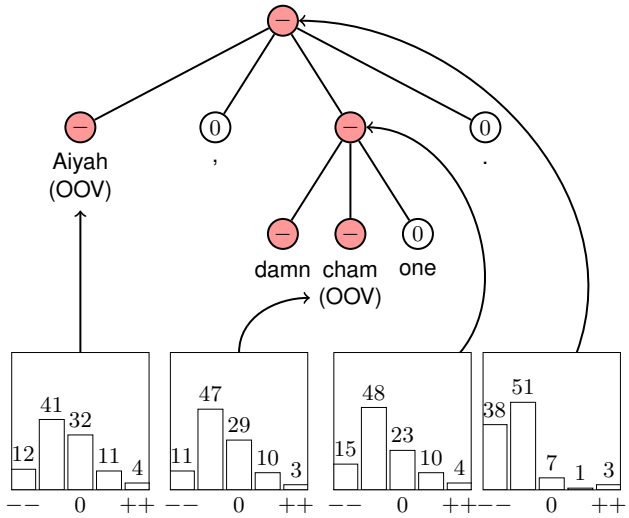

Fig. 4. The same example as Figure 2, but with the OOV words projected into the RNTN-space using AdaptCo. With this adaptation, the projections for *Aiyah* and *cham* would be classified as negative. This results in the entire sentence being classified to a negative sentiment.

Therefore, we used joint mutual information (shown to be robust for small datasets [22]) to pre-select 10 features for each run.

We evaluated each system as a ranking system: in our annotation, trolls are annotated with a *troll-score* defined in Section III-A. We ranked the users according to SVM^{rank}with the highest predicted scores first.

SVM^{rank} minimizes the number of pairs of instances where the order of their ranks are inverted or swapped. Theoretically, for a binary class problem, minimizing the number of swaps is equivalent to maximizing the area under the Receiver operating characteristic [23]. For ranking, SVM^{rank} optimizes the minimum number of swaps for multiple ordinal classes, which generalizes to maximizing the area under the receiver operating characteristic (ROC) of multiple classes [24]. The generalized ROC for multiple classes reported in the experiment results can be computed as follows:

$$\text{generalized ROC} = \left(1 - \frac{\#\text{PairedSwap}}{\text{maxPairedSwap}}\right) \times 100, \quad (5)$$

where #PairedSwap is the number of swapped pairs between the ground truth and the predictions, and maxPairedSwap is the maximum possible (worst case) number of swapped pairs that can occur in ranking the test set. In the case where the number of classes is two, generalized ROC is an estimation of ROC for two classes [23].

We compare the following four approaches in our experiments:

Random	The expected number of swaps estimated by Monte Carlo method over 1 million times.
NoAdapt	The vanilla RNTN sentiment analysis in Stanford NLP [1], trained on movie reviews. We used the pre-trained model that can be downloaded as part of Stanford NLP.
AdaptW2V	The word2vec adaptation described in Section IV-B1. Word2vec was trained on more than 100K forum threads to project into a 100-dimensional vector space using the continuous bag-of-words model for a window of 5 words. The threshold for screening noise is $\theta = 0.01$.
AdaptCo	The co-occurrence domain adaptation on the Stanford sentiment analysis as described in Section IV-B2. We used 45,000 threads out of the 100K threads for this purpose.

TABLE III. GENERALIZED ROC OF THE FOUR METHODS FOR DIFFERENT NUMBERS OF ORDINAL CLASSES. METHOD AdaptCo OUTPERFORMS ALL OTHER METHODS IN ALL CASES.

	Generalized ROC /%			
#classes	Random	NoAdapt	AdaptW2V	AdaptCo
2	50	60	69	78
3	50	61	62	67
6	50	58	57	69

We compare these methods using groupings into two, three and six ordinal classes (see Section III-C).

B. Experimental Results

The generalized ROC results are summarised in Table III. The results of Random for all cases are 50%; this shows that generalized ROC is a good indicator of the performance. Although, SVM^{rank} is trained on only 20 samples (that is, users), all the three methods, namely NoAdapt, AdaptW2V and AdaptCo, performed much better than Random, by at least 7%. This shows that the features in Table II are useful for detecting trolls. For all number of ordinal classes, AdaptCo always reported the highest generalized ROC among all the methods, improving over NoAdapt results by at least 6% and over AdaptW2V by at least 5%.

Table IV gives the 95% significance-test results using the Wilcoxon signed-rank test on each pair of method. It shows that AdaptCo outperformed all the others significantly. Taken together with the results presented in Table III, we conclude that AdaptCo is the preferred method for detecting trolls in a new domain with limited annotated data.

In Table III, AdaptW2V reports higher generalized ROC than NoAdapt for 2 classes, with 9% improvement, but it fails to improve results for more ordinal classes. We speculate the reason to be the different natures of the word2vec-space compared to the RNTN-space: while the word2vec-space is learned only from the sequence of words in sentences, the RNTN-space is learned specifically for the prediction of sentiments in words and phrases. For example, in the word2vec-space trained on our forum data, the word closest to *good* is *bad*. While we understand the semantic closeness between the two words, they express opposite sentiment for the sentiment analysis task.

VI. CONCLUSION

In this paper, we have shown that sentiment analysis is useful in detecting trolls: we achieved 60% and 58% generalized receiver operating characteristic (ROC) for binary and ordinal troll classification on our forum data respectively using features derived from sentiment analysis as inputs to SVM^{rank} for learning to detect trolls.

Further, to improve the Stanford recursive-neural-tensor-network sentiment analysis that was trained on annotated movie reviews, we applied domain adaptation methods to adapt the model to the targeted forum of interest. We experimented with two domain adaptation methods to project out-of-vocabulary words into the sentiment-analysis semantic-space:

TABLE IV. WILCOXON SIGNED-RANK TEST RESULTS AT 95% CONFIDENCE LEVEL. A $+$ VALUE MEANS THAT THE METHOD LABELLED BY THE ROW IS SIGNIFICANTLY BETTER THAN THE METHOD LABELLED BY THE COLUMN. A $\approx$ VALUE MEANS THAT IS NO SIGNIFICANT DIFFERENCE.

		Significance comparison		
#classes	Method	Random	NoAdapt	AdaptW2V
2	NoAdapt	$+$		
	AdaptW2V	$+$	$+$	
	AdaptCo	$+$	$+$	$+$
3	NoAdapt	$+$		
	AdaptW2V	$+$	$\approx$	
	AdaptCo	$+$	$+$	$+$
6	NoAdapt	$+$		
	AdaptW2V	$+$	$\approx$	
	AdaptCo	$+$	$+$	$+$

(i) AdaptW2V that uses the word2vec semantic-space learned on the target domain data to project unknown words into the sentiment space, and (ii) AdaptCo that projects unknown words into the sentiment-space using the sentiment vectors of sentences containing the unknown words in the target domain data.

Our experiments showed that AdaptCo signficantly outperformed all the other methods, achieving 78% and 69% generalized ROC for the binary and ordinal troll prediction tasks respectively. In constast, AdaptW2V did have have any significant improvement over no adaptation. We have also identified a possible reason for the failure of AdaptW2V.

There are a number of different directions for possible future work. One is to augment sentiment features with content features from topic modelling. Another is to directly use the small target data set with the large movie-review data set during the learning of RNTN as a transfer-learning approach to domain adaptation.

ACKNOWLEDGMENT

We thank Toon Joo Wee, Suzanna Xin Yun Sia, Li Qin Johny Quek, and Lin Zhi Bernice Khoo for their contributions to the definition of the troll annotation task for us. We also thank Toon Joo Wee for involvement in building the corpus. We are grateful to DSO National Laboratories, Singapore, for the permission to publish this work.

REFERENCES

[1] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA: Association for Computational Linguistics, October 2013, pp. 1631–1642.

[2] T. Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY,

USA: ACM, 2002, pp. 133–142. [Online]. Available: http://doi.acm.org/10.1145/775047.775067

[3] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: Sentiment classification using machine learning techniques," in *Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp. 79–86. [Online]. Available: http://dx.doi.org/10.3115/1118693.1118704

[4] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2005, pp. 115–124. [Online]. Available: http://dx.doi.org/10.3115/1219840.1219855

[5] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification," in *Association for Computational Linguistics*, Prague, Czech Republic, 2007.

[6] J. R. E. Leimgruber, "Singapore English," *Language and Linguistics Compass*, vol. 5, no. 1, 2011.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, 2013. [Online]. Available: http://arxiv.org/abs/1301.3781

[8] S. Dollberg, "The metadata troll detector," Swiss Federal Institute of Technology, Zurich, Distributed Computing Group, Computer Engineering and Networks Laboratory, Tech. Rep. Semester Thesis, 2015.

[9] Kaggle Inc., "Detecting insults in social commentary," 2012. [Online]. Available: http://www.kaggle.com/c/detecting-insults-in-social-commentary

[10] C.-W. Seah, I. W. Tsang, and Y.-S. Ong, "Transductive ordinal regression," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 7, pp. 1074–1086, 2012. [Online]. Available: http://dx.doi.org/10.1109/TNNLS.2012.2198240

[11] ——, "Transfer ordinal label learning." *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 11, pp. 1863–1876, 2013. [Online]. Available: http://dx.doi.org/10.1109/TNNLS.2013.2268541

[12] C.-W. Seah, I. W. Tsang, Y.-S. Ong, and G. K. K. Lee, "Predictive distribution matching SVM for multi-domain learning," in *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part I*, 2010, pp. 231–247. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-15880-3_21

[13] P. Nakov, S. Rosenthal, Z. Kozareva, V. Stoyanov, A. Ritter, and T. Wilson, "SemEval-2013 task 2: Sentiment analysis in Twitter," in *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. Atlanta, Georgia, USA: Association for Computational Linguistics, June 2013, pp. 312–320. [Online]. Available: http://www.aclweb.org/anthology/S13-2052

[14] T. Chen and M. Kan, "Creating a live, public short message service corpus: The NUS SMS corpus," *CoRR*, 2011. [Online]. Available: http://arxiv.org/abs/1112.2468

[15] L. Becker, G. Erhart, D. Skiba, and V. Matula, "AVAYA: Sentiment analysis on Twitter with self-training and polarity lexicon expansion," in *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. Association for Computational Linguistics, 2013, pp. 333–340. [Online]. Available: http://aclweb.org/anthology/S13-2055

[16] J. Blitzer, D. P. Foster, and S. M. Kakade, "Zero-shot domain adaptation: A multi-view approach," Toyota Technological Institute, Tech. Rep. TTI-TR-2009-1, 2009. [Online]. Available: http://www.ttic.edu/technical_reports/ttic-tr-2009-1.pdf

[17] O. Levy and Y. Goldberg, "Linguistic regularities in sparse and explicit word representations," in *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*. Ann Arbor, Michigan: Association for Computational Linguistics, June 2014, pp. 171–180. [Online]. Available: http://www.aclweb.org/anthology/W/W14/W14-1618

[18] Wikipedia, "Troll (internet) — Wikipedia, the free encyclopedia," 2015, [Online; accessed 13-March-2015]. [Online]. Available: http://en.wikipedia.org/wiki/Troll_%28Internet%29

[19] P. Shachaf and N. Hara, "Beyond vandalism: Wikipedia trolls," *Journal of Information Science*, vol. 36, no. 3, pp. 357–370, 2010.

[20] E. E. Buckels, P. D. Trapnell, and D. L. Paulhus, "Trolls just want to have fun," *Personality and Individual Differences*, vol. 67, pp. 97–102, Sep. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.paid.2014.01.016

[21] D. Klein and C. D. Manning, "Accurate unlexicalized parsing," in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, 2003, pp. 423–430.

[22] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *Journal of Machine Learning Research*, vol. 13, pp. 27–66, Jan. 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2188385.2188387

[23] T. Joachims, "A support vector method for multivariate performance measures," in *Proceedings of the 22Nd International Conference on Machine Learning*, ser. ICML '05. New York, NY, USA: ACM, 2005, pp. 377–384. [Online]. Available: http://doi.acm.org/10.1145/1102351.1102399

[24] ——, "Training linear SVMs in linear time," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 217–226. [Online]. Available: http://doi.acm.org/10.1145/1150402.1150429