Semantic Patterns for Sentiment Analysis of Twitter

Hassan Saif,[1] Yulan He,[2] Miriam Fernandez,[1] and Harith Alani[1]

[1] Knowledge Media Institute, The Open University, United Kingdom
{h.saif, m.fernandez, h.alani}@open.ac.uk
[2] School of Engineering and Applied Science, Aston University, United Kingdom
y.he@cantab.net

Abstract. Most existing approaches to Twitter sentiment analysis assume that sentiment is explicitly expressed through affective words. Nevertheless, sentiment is often implicitly expressed via latent semantic relations, patterns and dependencies among words in tweets. In this paper, we propose a novel approach that automatically captures patterns of words of similar contextual semantics and sentiment in tweets. Unlike previous work on sentiment pattern extraction, our proposed approach does not rely on external and fixed sets of syntactical templates/patterns, nor requires deep analyses of the syntactic structure of sentences in tweets. We evaluate our approach with tweet- and entity-level sentiment analysis tasks by using the extracted semantic patterns as classification features in both tasks. We use 9 Twitter datasets in our evaluation and compare the performance of our patterns against 6 state-of-the-art baselines. Results show that our patterns consistently outperform all other baselines on all datasets by 2.19% at the tweet-level and 7.5% at the entity-level in average F-measure.

Keywords: Sentiment Analysis, Semantic Patterns, Twitter

1 Introduction

Sentiment analysis on Twitter has established itself in the past few years as a solid research area, providing organisations and businesses with efficient tools and solutions for monitoring their reputation and tracking the public opinion on their brands and products.

Statistical methods to Twitter sentiment analysis rely often on machine learning classifiers trained from syntactical and linguistic features such as word and letter n-grams, part-of-speech tags, prior sentiment of words, microblogging features, etc [2, 5, 13]. However, merely relying on the aforementioned features may not lead to satisfactory sentiment detection results since sentiment is often context-dependent. Also, people tend to convey sentiment in more subtle linguistic structures or patterns [16]. Such patterns are usually derived from the syntactic [16] or semantic relations [6] between words in text. For example, the adjective word "`mean`" when preceded by a verb, constitutes a pattern of negative sentiment as in: "`she said mean things`". Also, the word "`destroy`" formulates a positive pattern when occurs with the concept "`invading germs`".

Both syntactic and semantic approaches to extracting sentiment patterns have proven successful when applied to documents of formal language and well-structured sentences [16, 8]. However, applying either approach to Twitter data faces several challenges.

Firstly, tweets data are often composed of sentences of poor grammatical and syntactical structures due to the the extensive use of abbreviations and irregular expressions in tweets [20]. Secondly, both approaches function with external knowledge sources. Most syntactic approaches rely on fixed and pre-defined sets of syntactic templates for pattern extraction. On the other hand, semantic approaches rely on external ontologies and common sense knowledge bases. Such resources, although useful, tend to have fixed domains and coverages, which is especially problematic when processing general Twitter streams, with their rapid semiotic evolution and language deformations [19].

In this paper, we propose a novel approach for automatically extracting semantic sentiment patterns of words on Twitter. We refer to these patterns from now on as *SS-Patterns*. Unlike most other approaches, our proposed approach does not rely on the syntactic structure of tweets, nor requires pre-defined syntactic templates. Instead, it extracts patterns from the contextual semantic and sentiment similarities between words in a given tweet corpus [19]. Contextual semantics (aka statistical semantics) are based on the proposition that meaning can be extracted from words co-occurrences [28, 26].

We apply our approach to 9 different Twitter datasets, and validate the extracted patterns by using them as classification features in two sentiment analysis tasks: (i) tweet-level sentiment classification, which identifies the overall sentiment of individual tweets, and (ii) entity-level sentiment classification, which detects sentiment towards a particular entity (e.g., `Obama`, `Cancer`, `iPad`). To this end, we train several supervised classifiers from SS-Patterns and compare the sentiment classification performance against models trained from 6 state-of-the-art sets of features derived from both the syntactic and semantic representations of words.

Our results show that our SS-Patterns consistently outperform all our baseline feature sets, on all 9 datasets, in both tweet-level and entity-level sentiment classification tasks. At the tweet level, SS-Patterns improve the classification performance by 1.94% in accuracy and 2.19% in F-measure on average. Also, at the entity level, our patterns produce 6.31% and 7.5% higher accuracy and F-measure than all other features respectively.

We also conduct quantitative and qualitative analyses on a sample of the patterns extracted by our approach and show that the effectiveness of using SS-Patterns as additional features for classifier training is attribute to their ability in capturing words with similar contextual semantics and sentiment. We also show that our extraction approach is able to detect patterns of controversial sentiment (strong opposing sentiment) expressed by people towards certain entities.

The main contributions of this paper can be summarised as follows:

- Propose a novel approach that automatically extracts patterns from the contextual semantic and sentiment similarities of words in tweets.
- Use patterns as features in tweet- and entity-level sentiment classification tasks, and compare the classification performance against 6 state-of-the-art baselines on 9 Twitter datasets in order to avoid the bias that any single dataset or baseline may introduce.
- Perform a cross comparison between the syntactic and semantic baseline feature sets used in our work and show the effectiveness of the latter for tweet-level sentiment classification over the former.

– Conduct quantitative and qualitative analyses on a sample of our extracted semantic sentiment patterns and show the potential of our approach for finding patterns of entities of controversial sentiment in tweets.

The remainder of this paper is structured as follows. Related work is discussed in Section 2. The proposed approach to extracting semantic sentiment patterns is presented in Section 3. Experimental setup and results are presented in Sections 4 and 5 respectively. Our pattern analysis study is described in Section 6. Discussion and future work are covered in Section 7. Finally, we conclude our work in Section 8.

2 Related Work

Much work on Twitter sentiment analysis follows the statistical machine learning approach by training supervised classifiers (e.g., Naïve Bayes, Maximum Entropy and Support Vector Machines) from features extracted from tweets such as word and letter n-grams [9, 15, 2], lexicon features (i.e., prior sentiment of words in sentiment lexicons) [5], microblogging features [10], POS tags [1] and several combinations of them [13]. Classifiers trained from these types of features have produced relatively high performance on various Twitter datasets with accuracies raging between 80% and 86%. However, it has been argued that sentiment in text is not always associated with individual words, but instead, through relations and dependencies between words, which often formulate sentiment [16].

In previous work, these relations are usually complied as a set of syntactic patterns (i.e., Part-of-Speech patterns) [25, 16, 24], common sense concepts [6], semantic concepts [21, 8], or statistical topics [20, 11].

For example, Riloff et al. [16] proposed extracting sentiment patterns from the syntactic relations between words in sentences. To this end, they used a fixed set of pre-defined POS templates, e.g., `<subject> passive-verb` which maps to the opinionated sentence "`<customer> was satisfied`" and `<subject> active-verb` that maps to "`<she> complained`". The extracted patterns were then incorporated into high-precision classifiers (HP-Subj and HP-Obj) in order to increase their recall.

One limitation of the syntactic extraction methods is that they are usually limited to the number of the syntactic templates they use. Moreover, these methods are often semantically weak, that is, they do not consider the semantics of individual words in their patterns. This may constitute a problem when trying, for example, to identify context-sensitive sentiment (e.g., `<beer> is cold` and `<weather> is cold`).

Conceptual semantic sentiment methods, on the other hand, utilize both syntactic and semantic processing techniques in order to capture the latent conceptual semantic relations in text that implicitly convey sentiment. For example, Cambria and Hussain [6] proposed *Sentic Computing*, a sentiment analysis paradigm, in which common sense concepts (e.g., "`happy birthday`", "`simple life`") are extracted from texts and assigned to their sentiment orientations using semantic parsing and affective common sense knowledge sources. Gangemi et al. [8] further investigated the syntactic structure of sentences in order to detect more fine grained relations between the different semantic parts within it. For example, their approach is able to detect not only the sentiment in text, but also the opinionated topics, subtopics, the opinion holders and their sentiment.

The semantic methods, therefore, are more sensitive to the latent semantic relations between words in texts than syntactic methods. Nevertheless, in the above works, neither

syntactic nor semantic methods are tailored to Twitter due to the lack of language formality and well structured sentences in tweets. Moreover, Semantic methods are usually limited to the scope of their underlying knowledge bases, which is especially problematic when processing general Twitter streams, with their rapid semiotic evolution and language deformations.

Contextual or statistical semantic methods extract patterns of semantically similar words by looking at the words' co-occurrence patterns in a given corpus [28, 26]. LDA is a state-of-the-art method that have been widely used to this end [4].[3] For example, Lin et al. [11] propose JST, a topic generative model based on LDA. JST extracts, not only the patterns (topics) of words in text, but also their associated sentiment. The topics along with their associated sentiment have been evaluated in our previous work [20] and proven valuable for sentiment analysis on Twitter. However, these methods usually rely on the bag-of-words representation, and therefore are often unable to handle negations and other patterns that strongly influence sentiment.

In order to overcome the aforementioned limitations of the above methods, we design our sentiment pattern extraction approach in a way that captures patterns based on the contextual semantic and sentiment similarities between words in a Twitter corpus. Our approach does not rely on the syntactic structures in tweets, nor requires using pre-defined syntactic template sets or external semantic knowledge sources.

3 Semantic Sentiment Patterns of Words

Semantic sentiment patterns, by definition, are clusters of words which have similar contextual semantics and sentiment in text. Based on this definition, the problem of capturing these patterns in tweets data breaks down into three phases as illustrated in Figure 1. In the first phase, tweets in a given data collection are syntactically processed in order to reduce the amount of noise and language informality in them. In the second phase we apply the SentiCircle representation model [19] on the processed tweets to capture the contextual semantics and sentiment of words in the tweets. In the third step, the semantic sentiment patterns are formed by clustering words that share similar semantics and sentiment (i.e., similar SentiCircles).

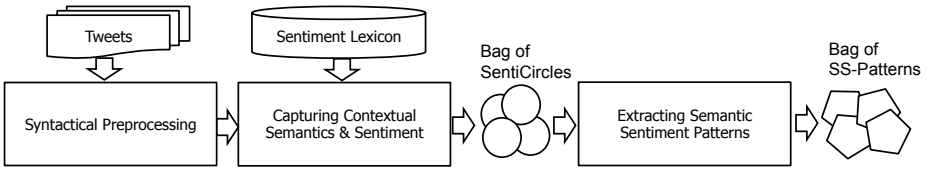

Fig. 1: The systematic workflow of capturing semantic sentiment patterns from Twitter data

In the subsequent sections we further describe each of the aforementioned phases in some more details:

3.1 Syntactical Preprocessing

Tweets are usually composed of incomplete, noisy and poorly structured sentences due to the frequent presence of abbreviations, irregular expressions, ill-formed words and

[3] Patterns extracted by LDA are usually called Topics

non-dictionary terms. Such noisy nature of tweets has been shown to indirectly affect the sentiment classification performance [20]. This phase therefore, aims at reducing the amount of noise in the tweets by applying a series of pre-processing steps as follows:

- All URL links in the corpus are replaced with the term "`URL`"
- Remove all non-ASCII and non-English characters
- Revert words that contain repeated letters to their original English form. For example, the word "`maaadddd`" will be converted to "`mad`" after processing.
- Process contraction and possessive forms. For example, change "`he's`" and "`friend's`" to "`he`" and "`friend`"

Note that we do not remove stopwords from the data since they tend to carry sentiment information as shown in [18].

3.2 Capturing Contextual Semantics and Sentiment of Words

SS-patterns are formed from the contextual semantic similarities among words. Therefore, a key step in our pipeline is to capture the words' contextual semantics in tweets. To this end, we use our previously proposed semantic representation model, SentiCircle [19].

Briefly speaking, the SentiCircle model extracts the contextual semantics of a word from its co-occurrences with other words in a given tweet corpus. These co-occurrences are then represented as a geometric circle which is subsequently used to compute the contextual sentiment of the word by applying simple trigonometric identities on it. In particular, for each unique term m in a tweet collection, we build a two-dimensional geometric circle, where the term m is situated in the centre of the circle, and each point around it represents a context term c_i (i.e., a term that occurs with m in the same context). The position of c_i, as illustrated in Figure 2, is defined jointly by its Cartesian coordinates x_i, y_i as:

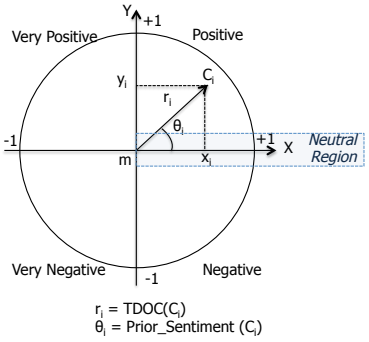

Fig. 2: SentiCircle of a term m

$$x_i = r_i \cos(\theta_i * \pi) \qquad\qquad y_i = r_i \sin(\theta_i * \pi)$$

Where θ_i is the polar angle of the context term c_i and its value equals to the prior sentiment of c_i in a sentiment lexicon before adaptation, r_i is the radius of c_i and its value represents the degree of correlation (tdoc) between c_i and m, and can be computed as:

$$r_i = tdoc(m, c_i) = f(c_i, m) \times \log \frac{N}{N_{c_i}}$$

where $f(c_i, m)$ is the number of times c_i occurs with m in tweets, N is the total number of terms, and N_{c_i} is the total number of terms that occur with c_i. Note that all terms' radii in the SentiCircle are normalised. Also, all angles' values are in radians.

The rational behind using this circular representation shape is to benefit from the trigonometric properties it offers for encoding the contextual semantics of a term as *sentiment orientation* and *sentiment strength*. Y-axis defines the sentiment of the term, i.e., a positive y value denotes a positive sentiment and vice versa. The X-axis defines

the sentiment strength of the term. The smaller the x value, the stronger the sentiment.[4] This, in turn, divides the circle into four sentiment quadrants. Terms in the two upper quadrants have a positive sentiment ($\sin\theta > 0$), with upper left quadrant representing stronger positive sentiment since it has larger angle values than those in the top right quadrant. Similarly, terms in the two lower quadrants have negative sentiment values ($\sin\theta < 0$). Moreover, a small region called the "*Neutral Region*" can be defined. This region, as shown in Figure 2, is located very close to X-axis in the "*Positive*" and the "*Negative*" quadrants only, where terms lie in this region have very weak sentiment (i.e, $|\theta| \approx 0$).

The Sentiment Median of SentiCircle In summary, the SentiCircle of any term m is composed by the set of (x, y) Cartesian coordinates of all the context terms of m. An effective way to compute the overall sentiment of m is by calculating the geometric median of all the points in its SentiCircle. Formally, for a given set of n points $(p_1, p_2, ..., p_n)$ in a SentiCirlce Ω, the 2D geometric median g is defined as: $g = \arg\min_{g\in\mathbb{R}^2} \sum_{i=1}^{n} \|p_i - g\|_2$. The boundaries of the neutral region can be computed by measuring the density distribution of terms in the SentiCircle along the Y-axis. In this paper we use similar boundary values to the ones in [19] as we use the same evaluation datasets. We call the geometric median g the **SentiMedian** as its position in the SentiCircle determines the total contextual-sentiment orientation and strength of m.

3.3 Extracting Patterns from SentiCircles

At this stage all the unique words in the tweet collection have their contextual semantics and sentiment extracted and represented by means of their SentiCircles. It is very likely to find words in text which share similar contextual semantics and sentiment. In other words, finding words with similar SentiCircles. Therefore, this phase seeks to find such potential semantic similarities in tweets by building clusters of similar SentiCircles. The output of this phase is a set of clusters of words, which we refer to as the *semantic sentiment patterns of words (SS-Patterns)*.

SentiCircles Clustering We can capture patterns that emerge from the similarity of word's sentiment and contextual semantics by clustering the SentiCircles of those words. In particular, we perform a clustering task fed by dimensions that are provided by SentiCircles; *density*, *dispersion*, and *geometry*. Density and dispersion usually characterise terms and entities that receive controversial sentiment in tweets as will be further explained and validated in Section 6. Geometry, on the other hand, preserves the contextual sentiment orientation and strength of terms. Once we extract the vectors that represent these three dimensions from all the terms' SentiCircles, we feed them into a common clustering method; k-means.

In the following, we describe the three dimensions we extract from each term's SentiCircle Ω along with the components they consist of:

- *Geometry*: includes the X- and Y-component of the SentiMedian $g(x_g, y_g) \in \Omega$
- *Density*: includes the total density of points in the SentiCircle Ω and its computed as: $density(\Omega) = N/M$, where N is the total number of points in the SentiCircle and M is the total number of points in the SentiCircles of all terms.

[4] This is because $\cos\theta < 0$ for large angles.

We also compute five density components, representing the density of each sentiment quadrant in the SentiCircle (i.e., positive, very positive, negative and very negative quadrants) along with the density of its neutral region. Each of these components is computed as $density(Q) = P/N$ where P is the total number of points in the sentiment quadrant Q.

- *Dispersion*: the total dispersion of a SentiCircle refers to how scattered or condensed the points (context terms) in the circle. To calculate the value of this component, we use the *median absolute deviation measure (MAD)*, which computes the dispersion of Ω as the median of the absolute deviations from the SentiCircle's median point (i.e., the SentiMedian g_m) as:

$$mad(\Omega) = (\sum_{i=1}^{n} |p_i - g_m|)/N$$

Similarly, using the above equation, we calculate the dispersion of each sentiment quadrant and the neutral region in the SentiCircle. We also calculate the dispersion of the active region in SentiCircle (i.e., The SentiCircle after excluding points in the neutral region)

The last step in our pipeline is to apply k-means on all SentiCircles' dimensions' vectors. This results in a set of clusters $\mathcal{K} = (k_1, k_2, ..., k_c)$ where each cluster consists of words that have similar contextual semantics and sentiment. We call $\mathcal{K}$ as the pattern set and and $k_i \in \mathcal{K}$ the *semantic sentiment pattern*.

In the subsequent section we describe how to determine the number of patterns (clusters) in the data and how to validate the extracted patterns by using them as features in two sentiment classification tasks.

4 Experimental Setup

Our proposed approach, as shown in the previous section, extracts patterns of words of similar contextual semantics and sentiment. We evaluate the extracted SS-patterns by using them as classification features to train supervised classifiers for two sentiment analysis tasks, tweet- and entity-level sentiment classification. To this end, we use 9 publicly and widely used datasets in Twitter sentiment analysis literature [17]. Nine of them will be used for tweet-level evaluation and one for entity-level evaluation. As for evaluation baselines, we use 6 types of classification features and compare the performance of classifiers trained from our SS-patterns against those trained from these baseline features.

4.1 Tweet-Level Evaluation Setup

The first validation test we conduct on our SS-patterns is to measure their effectiveness as features for binary sentiment analysis of tweets, i.e., classifying the individual tweets as positive or negative. To this end, we use SS-patterns extracted from a given Twitter dataset to train two supervised classifiers popularly used for tweet-level sentiment analysis, Maximum Entropy (MaxEnt) and Naïve Bayes (NB) from Mallet.[5] We use 9 different Twitter datasets in our validation in order to avoid any bias that a single dataset can introduce. Numbers of positive and negative tweets within these datasets are summarised in Table 1, and detailed in the references added in the table.

[5] `http://mallet.cs.umass.edu/`

Dataset	Tweets	#Negative	#Positive	#Unigrams
Stanford Twitter Test Set (STS-Test) [9]	359	177	182	1562
Sanders Dataset (Sanders) [17]	1224	654	570	3201
Obama McCain Debate (OMD) [7]	1906	1196	710	3964
Health Care Reform (HCR) [22]	1922	1381	541	5140
Stanford Gold Standard (STS-Gold) [17]	2034	632	1402	4694
Sentiment Strength Twitter Dataset (SSTD) [23]	2289	1037	1252	6849
The Dialogue Earth Weather Dataset (WAB) [3]	5495	2580	2915	7485
The Dialogue Earth Gas Prices Dataset (GASP) [3]	6285	5235	1050	8128
Semeval Dataset (Semeval) [14]	7535	2186	5349	15851

Table 1: Twitter datasets used for tweet-level sentiment analysis evaluation. Instructions on how to obtain these datasets are provided in [17].

4.2 Entity-Level Evaluation Setup

In the second validation test, we evaluate the usefulness of SS-Patterns as features for entity-level sentiment analysis, i.e., detecting sentiment towards a particular entity. To this end, we perform a 3-way sentiment classification (negative, positive, neutral) on a dataset of 58 named entities extracted from the STS-Gold dataset and manually labelled with their sentiment class. Numbers of negative, positive and neutral entities in this dataset are listed in Table 2 along with five examples of entities under each sentiment class. Details of the extraction and the annotation of these entities can be found in [17].

	Negative Entities	Positive Entities	Neutral Entities
Total Number	13	29	16
Examples	Cancer	Lakers	Obama
	Lebron James	Katy Perry	Sydney
	Flu	Omaha	iPhone
	Wii	Taylor Swift	Youtube
	Dominique Wilkins	Jasmine Tea	Vegas

Table 2: Numbers of negative, positive and neutral entities in the STS-Gold Entity dataset along with examples of 5 entities under each sentiment class.

The entity sentiment classifier we use in our evaluation is based on maximum likelihood estimation (MLE). Specifically, we use tweets in the STS-Gold dataset to estimate the conditional probability $P(c|e)$ of an entity e assigned with a sentiment class $c \in \{Positive, Negative\}$ as: as $P(c|e) = N(e,c)/N(e)$ where $N(e,c)$ is the frequency of an entity e in tweets assigned with a sentiment class c and $N(e)$ is the frequency of the entity e in the whole corpus.

We incorporate our SS-Pattern features and other baseline features (Section 4.3) into the sentiment class estimation of e by using the following back-off strategy:

$$\hat{c} = \begin{cases} P(c|e) & \text{if } N(e,c) \neq 0 \\ P(c|f) & \text{if } N(e,c) = 0 \end{cases} \quad (1)$$

where f is the incorporated feature (e.g., the SS-Pattern of e) and $P(c|f)$ is the conditional probability of the feature f assigned with a sentiment class c and it can be also estimated using MLE. The rationale behind the above back-off strategy is that some entities might not occur in tweets of certain sentiment class, leading therefore, to zero probabilities. In such cases we resort to the sentiment of the latent features associated with these entities in the dataset.

The final sentiment of e can be derived from the ratio $R_e = P(c = Positive|e)/P(c = Negative|e)$. In particular, the sentiment is *neutral* if R_e is less than a threshold γ, otherwise the sentiment is *negative* if $R_e < 1$ or *positive* if $R_e > 1$.

We determine the value of γ by plotting the ratio R_e for all the 58 entities and check where the plot converges. In our case, the ratio plot converged with $\gamma = 0.3$.

4.3 Evaluation Baselines

The baseline model in our evaluation is a sentiment classifier trained from word unigram features. Table 1 shows the number of unique unigram features extracted from our datasets.

In addition to unigrams, we propose comparing our SS-Pattern features against the below described five state-of-the-art types of features in sentiment analysis. Amongst them, two sets of features are derived from the syntactical characteristics of words in tweets (POS features, and Twitter features), one is based on the prior sentiment orientation of words (Lexicon features) and two are obtained from the semantic representation of words in tweets (Semantic Concept features and LDA-Topic features):

1. Twitter Features: refer to tokens and characters that are popularly used in tweet messages such as hashtags (e.g., "`#smartphone`"), user mentions (e.g, "`@obama`"), the tweet reply token ("`RT`") and emoticons (e.g., "`:)  :D <3  o_O`").

2. Part-of-Speech Features: refer to the part-of-speech tags of words in tweets (e.g., verbs, adjectives, adverbs, etc). We extract these features using the TweetNLP POS tagger.[6]

3. Lexicon Features: these features are formed from the opinionated words in tweets along with their prior sentiment labels (e.g., "`good_positive`", "`bad_negative`", "`nice_positive`", etc.). We assign words with their prior sentiments using both Thelwall [23] and MPQA [27] sentiment lexicons.

4. Semantic Concept Features: This type of features refers to the semantic concepts (e.g., "`person`", "`company`", "`city`") that represent entities (e.g., "`Obama`", "`Motorola`", "`Vegas`") appearing in tweets. To extract the entities and their associated concepts in our datasets we use AlchemyAPI,[7] which we have previously evaluated its semantic extraction performance on Twitter data [21]. The number of extracted concepts in each dataset is listed in Table 3.

Dataset	STS-Test	Sanders	OMD	HCR	STS-Gold	SSTD	WAB	GASP	Semeval
No. of Concepts	299	1407	2191	1626	1490	699	1497	3614	6875

Table 3: Numbers of the semantic concepts extracted from all datasets

5. LDA-Topic Features: These features denote the latent topics extracted from tweets using the probabilistic generative model, LDA [4]. LDA assumes that a document is a mixture of topics and each topic is a mixture of probabilities of words that are more likely to co-occur together under the topic. For example the topic "`iPhone`" is more likely to generate words like "`display`" and "`battery`". Therefore, LDA-Topics represent groups of words that are semantically related. To extract these latent topics from our datasets we use an implementation of LDA provided by Mallet. LDA requires defining the number of topics to extract before applying it on the data. To this end, we ran LDA with different choices of numbers of topics (e.g., 1 topic, 10 topics, 20 topics, 30 topics, etc). Among all choices, 10 topics was the opitmal number that gave the highest

[6] `http://www.ark.cs.cmu.edu/TweetNLP/`

[7] `http://www.alchemyapi.com`

sentiment classification performance when the topics were incorporated as additional features into the feature space.

Note that all the above sets of features are combined with the original unigram features when training the baseline sentiment classifiers for both entity- and tweet-levels.

4.4 Number of SS-Patterns in Data

As described earlier, extracting SS-patterns is a clustering problem that requires determining beforehand the number of clusters (patterns) to extract. To this end, we run k-means for multiple times with k varying between 1 and 100. We then plot the within-cluster sum of squares for all the outputs generated by k-means. The optimum number of clusters is found where an "*elbow*" appears in the plot [12]. For example, Figure 4 shows that the optimum number of clusters for the GASP dataset is 17, which in other words, represents the number of SS-Patterns features that our sentiment classifiers should be trained from. Table 4 shows the number of SS-Patterns extracted by our model for each dataset.

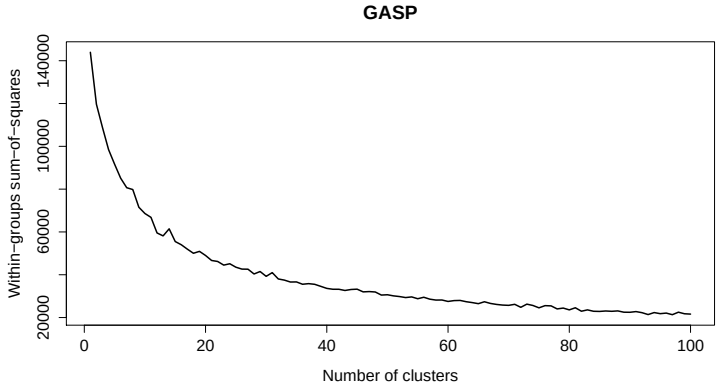

Fig. 3: Within-cluster sum of squares for different numbers of clusters (SS-Patterns) in the GASP dataset.

Dataset	STS-Test	Sanders	OMD	HCR	STS-Gold	SSTD	WAB	GASP	Semeval
No. of SS-Patterns	18	20	23	22	26	24	17	17	19

Table 4: Numbers of SS-Patterns extracted from all datasets

5 Evaluation Results

In this section, we report the results from using our proposed SS-Patterns as features for tweet- and entity-level sentiment classification tasks and compare against the baselines described in Section 4.3. All experiments in both evaluation tasks are done using 10-fold cross validation.

5.1 Sentiment Patterns for Tweet-Level Sentiment Classification

The first task in our evaluation aims to asses the usefulness of SS-Patterns as features for binary sentiment classification of tweets (positive vs. negative).[8] We use NB and

[8] Unlike entity-level, we do not perform 3-way classification (positive, negative, netrual) in this task since not all the 9 datasets contain tweets of neutral sentiment.

MaxEnt classifiers trained from word unigrams as the starting baseline models (aka, unigram models). We then compare the performance of classifiers trained from other types of features against these unigram models.

Table 5 shows the results in accuracy and average F1 measure of both unigram models across all datasets. The highest accuracy is achieved on the GASP dataset using MaxEnt with 90.49%, while the highest average F-measure of 84.08% is obtained on the WAB dataset. On the other hand, the lowest performance in accuracy is obtained using NB on the SSTD dataset with 72.36%. Also, NB produces the lowest F1 of 66.69% on the HCR dataset. On average, MaxEnt outperforms NB by 1.04% and 1.35% in accuracy and F1 respectively. Hence, we use MaxEnt only to continue our evaluation in this task.

	Dataset	STS-Test	Sanders	OMD	HCR	STS-Gold	SSTD	WAB	GASP	SemEval	**Average**
MaxEnt	Acc	77.82	83.62	82.90	77.02	86.02	72.84	84.12	90.49	82.11	**81.88**
	F1	77.94	83.58	81.34	69.10	83.10	72.27	84.08	81.81	77.03	**78.91**
NB	Acc	81.06	82.66	81.57	74.27	84.22	72.36	82.79	88.16	80.44	**80.84**
	F1	81.07	82.52	79.93	66.69	80.46	72.20	82.74	78.15	74.35	**77.57**

Table 5: Accuracy and the average harmonic mean (F1 measure) obtained from identifying positive and negative sentiment using unigram features, where *Acc* is the classification accuracy.

Table 6 shows the results of MaxEnt classifiers trained from the 5 baseline sets of features (See Section 4.3) as well as MaxtEnt trained from our proposed SS-patterns, applied over all datasets. The table reports the average results in three sets of *minimum*, *maximum*, and *average* win/loss in accuracy and F-measure relating to the results of the unigram model in Table 5. For simplicity, we refer to MaxEnt classifiers trained from any syntactic feature set as *syntactic models* and we refer to those trained from any semantic feature set as *semantic models*.

It can be observed from these results in Table 6 that all syntactic and semantic models outperform on average the unigram model in both accuracy and F-measure. However, MaxEnt trained from our SS-Patterns significantly outperforms those models trained from any other set of features. In particular, our SS-Patterns produce on average 3.05% and 3.76 % higher accuracy and F1 than the unigram model. This is 2% higher performance than the average performance gain of all syntactic and semantic models. Moreover, we get a maximum improvement in accuracy and F-measure of 9.87% and 9.78% respectively over the unigram model when using our SS-Patterns for training. This is at least 3.54% and 3.61% higher than any other model. It is also worth noting that on the GASP dataset, where the minimum performance gain is obtained, MaxEnt trained from SS-Patterns gives a minimum improvement of 0.70%, while all other models suffer a performance loss of -0.45% averagely.

Finally, we notice that syntactic features, and more specifically the lexicon ones are highly competitive features to the semantic type of features. For example, lexicon features slightly outperform concept and LDA-Topic features. However, from the average performance in Table 6 of both types of features, we can see that semantic models are still bypassing syntactic models in both accuracy and F-measure by 0.71% and 0.75% on average respectively.

5.2 Results of Entity-Level Sentiment Classification

In this section, we report the evaluation results of using our SS-Patterns for entity-level sentiment classification on the STS-Gold Entity dataset using the entity sentiment

	Features	MaxEnt Classifier					
		Accuracy			F-Measure		
		Minimum	Maximum	Average	Minimum	Maximum	Average
Syntactic	Twitter Features	-0.23	3.91	1.24	-0.25	4.53	1.62
	POS	-0.89	2.92	0.79	-0.91	5.67	1.25
	Lexicon	-0.44	4.23	1.30	-0.38	5.81	1.83
	Average	-0.52	3.69	1.11	-0.52	5.33	1.57
Semantic	Concepts	-0.22	2.76	1.20	-0.40	4.80	1.51
	LDA-Topics	-0.47	3.37	1.20	-0.68	6.05	1.68
	SS-Patterns	**0.70**	**9.87**	**3.05**	**1.23**	**9.78**	**3.76**
	Average	0.00	5.33	1.82	0.05	6.88	2.32

Table 6: Win/Loss in Accuracy and F-measure of using different features for sentiment classification on all nine datasets.

classifier described in Section 4.2. Note that STS-Gold is the only dataset among the other 9 that provides named entities manually annotated with their sentiment labels (positive, negative, neutral). Therefore, our evaluation in this task is done using the STS-Gold dataset only.

Features	**Accuracy**	**Positive Sentiment**			**Negative Sentiment**			**Neutral Sentiment**			**Average**		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
Unigrams	48.28	92	79.31	85.19	6.67	7.69	7.14	22.22	25	23.53	40.3	37.33	38.62
LDA-Topics	58.62	92	79.31	85.19	31.82	53.85	40	36.36	25	29.63	53.39	52.72	51.6
Semantic Concepts	55.17	92	79.31	85.19	25	38.46	30.3	30.77	25	27.59	49.26	47.59	47.69
SS-Patterns	**60.34**	92	79.31	85.19	**34.78**	**61.54**	**44.44**	**40**	**25**	**30.77**	**55.59**	**55.28**	**53.47**

Table 7: Accuracy and averages of Precision, Recall, and F measures of entity-level sentiment classification using different features.

Table 7 reports the results in accuracy, precision (P), recall (R) and F1 measure of positive, negative and neutral sentiment classification performances from using unigrams, semantic concepts, LDA-Topics and SS-Patterns features. Generally, our SS-Patterns outperform all other features including word unigrams in all measures. In particular, merely using word unigrams for classification gives the lowest performance of 48.24% and 38.62% in accuracy and average F1. However, augmenting the feature space with SS-Patterns improves the performance significantly by 12.06% in accuracy and 14.85% in average F1. Our SS-Patterns also outperform LDA-Topics and semantic concepts features by at least 1.72% and 1.87% in accuracy and average F1.

As for per-class sentiment classification performance, we observe that all features produce high and similar performances on detecting positive entities. This is because classifiers trained from either feature set fail in detecting the sentiment of the same entities. Moreover, it seems that detecting negative and neutral entities are much more difficult tasks than detecting positive ones. For example, unigrams perform very poorly in detecting negative entities with a F1 less then 8%. Although the performance improves a lot by using SS-Patterns, it is still much lower than the positive classification performance. For neutral sentiment classification the performance is the lowest with unigrams (F1 = 23.53%) while it is the highest with SS-Patterns (F1 = 30.77%). Such varying performance might be due to the uneven sentiment class distribution in the entity dataset. As can be noted from Table 2, positive entities constitute 50% of the total number of entities while the neutral and negative entities form together the other 50%.

6 Within-Pattern Sentiment Consistency

Our approach, by definition, seeks to find SS-Patterns of terms of similar contextual semantics and sentiment. Therefore, SS-Patterns are best when they are consistent with the sentiment of their terms, that is, they consist mostly of terms of similar contextual sentiment orientations. In this section, we further study the sentiment consistency of our patterns on a set of 14 SS-Patterns extracted from the 58 annotated entities in the STS-Gold dataset. These number of patterns was determined based on the elbow method as explained in Section 4.4.

Table 8 shows four of the extracted patterns along with the top 5 entities within them and the entities' gold-standard sentiment. Patterns 3, 12 and 11 are strongly consistent since all entities within them have the same sentiment. On the other hand, Pattern 5 has low sentiment consistency as it contains entities of mixed sentiment orientations. We systematically calculate the sentiment consistency of a given SS-Pattern k_i as:

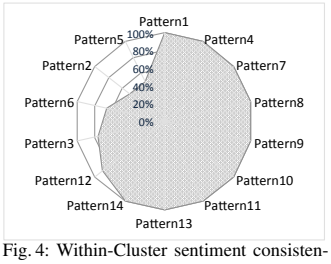

Fig. 4: Within-Cluster sentiment consistencies of in the STS-Gold Entity dataset

$$consistency(k_i) = \arg\max_{s \in \mathcal{S}} \frac{E_s}{E'} \qquad (2)$$

where $s \in \mathcal{S} = \{Positive, Negative, Neutral\}$ is the sentiment label, E_s is the number of entities of sentiment s and E' is the total number of entities within K_i.

Figure 4 depicts the sentiment consistency of the 14 SS-Patterns. 9 patterns out of 14 are perfectly consistent with the sentiment of their entities while two patterns have a consistency higher than 77%. Only patterns 2,5 and 6 have a consistency lower than 70%. Overall, the average consistency value across the 14 patterns reaches 88%.

Pattern.3 (Neutral)		**Pattern.12 (Positive)**		**Pattern.5 (Mixed)**		**Pattern.11 (Positive)**	
Entity	True Sentiment	Entity	True Sentiment	Entity	True Sentiment	Entity	True Sentiment
Brazil	Neutral	Kardashian	Positive	Cancer	Negative	Amy Adams	Positive
Facebook	Neutral	Katy Perry	Positive	Fever	Negative	Dallas	Positive
Oprah	Neutral	Beatles	Positive	Headache	Negative	Riyadh	Positive
Sydney	Neutral	Usher	Positive	McDonald	Neutral	Sam	Positive
Seattle	Neutral	Pandora	Positive	Xbox	Neutral	Miley Cyrus	Positive

Table 8: Example of three strongly consistent SS-Patterns (Patterns 3, 11, and 12) and one inconsistent SS-Pattern (Pattern 5), extracted from the STS-Gold Entity dataset

Sentiment Consistency vs. Sentiment Dispersion

From the above, we observed that patterns 2,5 and 6 have low sentiment consistency. Looking at characteristics of entities in these patterns, we notice that the average dispersion of their SentiCircles is 0.18 on average. This is twice higher than the dispersion of the entities within the other 11 strongly consistent patterns. Overall, we found a negative correlation of -0.42 between the sentiment consistency of SS-Patterns and the dispersion of their entities' SentiCircles. This indicates that SS-Patterns that contain entities of high dispersed SentiCircles are more likely to have low sentiment consistency. Based on the SentiCircle model (Section 3), these high dispersed entities either occur very infrequently or occur in different contexts of different sentiment in the tweet corpus.

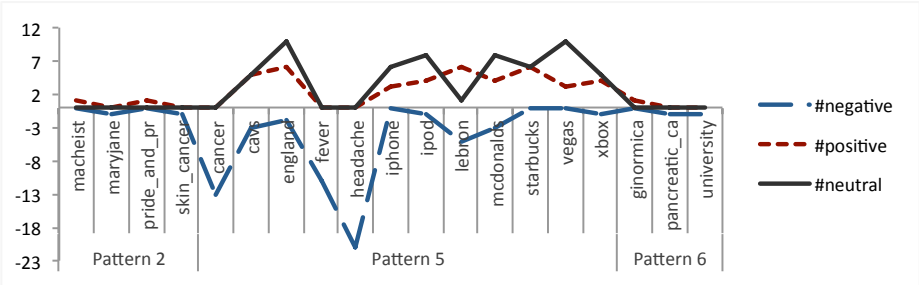

Fig. 5: Number of times that entities in Pattern 2, 5 and 6 receive negative, positive and neutral sentiment

To validate our above observation, we analyse the human sentiment votes on the 58 entities in STS-Gold dataset.[9] Figure 5 shows entities under patterns 2,5 and 6 along with number of times they receive negative, positive and neutral sentiment in tweets according to the three human coders. We observe that entities in patterns 2 and 6 occur very infrequently in tweets, yet with consistent sentiment. On the other hand, most entities in Pattern 5 occur more frequently in tweets. However, they receive strong and controversial sentiment (i.e., opposite sentiment). For example, the entity "`McDonald's`" occurs 3, 4 and 8 times with negative, positive and neutral sentiment respectively.

The above analysis shows the potential of our approach for generating patterns of entities that indicate sentiment disagreement or controversy in tweets.

7 Discussion and Future Work

We showed the value of our proposed approach in extracting semantic sentiment patterns of words and exploiting them for sentiment classification of tweets and entities. Our patterns, by definition, are based on words' similarities in a given context in tweets, which make them relevant to that specific context. This means that they might need updating more frequently than context-independent patterns, i.e., patterns derived based on pre-defined syntactic templates [16] or common-sense knowledge bases [6]. Hence, potential gain in performance may be obtained by combining context-independent patterns with our patterns, which constitutes a future task to this work.

For tweet-level sentiment classification, SS-Patterns were evaluated on 9 Twitter datasets with different results. For example, our SS-Patterns produced the highest performance improvement on the STS-Test dataset (+9.78% over the baseline) while the lowest improvement was obtained on the GASP dataset (+1.23%). Different factors might be behind such variance. For example our datasets differ in their sizes, sparsity degrees and sentiment classification distributions. We plan to further study the impact of these factors on (i) the quality of the extracted patterns and (ii) the sentiment classification performance.

For entity sentiment classification, evaluation was performed on one dataset and by using a single classifier. We noticed that detecting positive entities was much easier than detecting neutral or negative entities. This might be due to (i) the choice of the classifier we use or (ii) the large number of positive entities in this dataset. Therefore, as future

[9] Human votes on each entity are available to download with the STS-Gold dataset under `http://tweenator.com`.

work, we intend to continue experimenting with our patterns on multiple and balanced entity datasets and using several and more advanced entity sentiment classifiers.

We showed that our approach was able to discover patterns of terms and entities that could indicate sentiment disagreement, instability, or controversy in tweets. Those patterns have also shown low consistency with the sentiment of entities within them. Thus, one may expect terms under these patterns to have low contribution to the sentiment classification performance, and therefore, remove them from the feature space for sentiment classification. We are currently investigating this issue and its impact on the classification performance.

8 Conclusions

We proposed a novel approach for extracting patterns of words of similar contextual semantics and sentiment on Twitter. Our approach does not rely on the syntactical structure of tweets, nor uses external syntactic templates for pattern extraction.

We applied our approach on 9 Twitter datasets and validated the extracted patterns by incorporating them as classification features for sentiment classification of both tweets and entities. For tweet level sentiment classification, we used two supervised classifiers, NB and MaxEnt while for entity level we proposed a sentiment classifier based on Maximum Likelihood Estimation.

In both sentiment classification tasks and on all datasets, classifiers trained from our SS-Patterns showed a consistent and superior performance over classifiers trained from other 4 syntactical and 2 semantic sets of features.

We conducted an analysis of our SS-Patterns and showed that our patterns are strongly consistent with the sentiment of the terms within them. Also, the analysis showed that our approach was able to derive patterns of entities of controversial sentiment in tweets.

Acknowledgment

This work was supported by the EU-FP7 project SENSE4US (grant no. 611242).
We thank Grégoire Burel and Carlos Pedrinaci for the helpful discussions.

References

1. Agarwal, A., Xie, B., Vovsha, I., Rambow, O., Passonneau, R.: Sentiment analysis of twitter data. In: Proc. ACL 2011 Workshop on Languages in Social Media. Portland, Oregon (2011)
2. Aisopos, F., Papadakis, G., Varvarigou, T.: Sentiment analysis of social media content using n-gram graphs. In: Proc. the 3rd ACM international workshop on Social media (2011)
3. Asiaee T, A., Tepper, M., Banerjee, A., Sapiro, G.: If you are happy and you know it... tweet. In: Proc. the 21st ACM conference on Information and knowledge management (2012)
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. the Journal of machine Learning research 3, 993–1022 (2003)
5. Bravo-Marquez, F., Mendoza, M., Poblete, B.: Combining strengths, emotions and polarities for boosting twitter sentiment analysis. In: Proc. the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining (2013)
6. Cambria, E., Hussain, A.: Sentic computing: Techniques, tools, and applications, vol. 2. Springer (2012)
7. Diakopoulos, N., Shamma, D.: Characterizing debate performance via aggregated twitter sentiment. In: Proc. 28th Int. Conf. on Human factors in computing systems. ACM (2010)
8. Gangemi, A., Presutti, V., Reforgiato Recupero, D.: Frame-based detection of opinion holders and topics: A model and a tool. Computational Intelligence Magazine, IEEE (2014)

9. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford (2009)
10. Kouloumpis, E., Wilson, T., Moore, J.: Twitter sentiment analysis: The good the bad and the omg! In: Proceedings of the ICWSM. Barcelona, Spain (2011)
11. Lin, C., He, Y., Everson, R., Ruger, S.: Weakly supervised joint sentiment-topic detection from text. Knowledge and Data Engineering, IEEE Transactions on 24(6), 1134–1145 (2012)
12. Milligan, G.W., Cooper, M.C.: An examination of procedures for determining the number of clusters in a data set. Psychometrika 50(2), 159–179 (1985)
13. Mohammad, S.M., Kiritchenko, S., Zhu, X.: Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. arXiv preprint arXiv:1308.6242 (2013)
14. Nakov, P., Rosenthal, S., Kozareva, Z., Stoyanov, V., Ritter, A., Wilson, T.: Semeval-2013 task 2: Sentiment analysis in twitter. In: Proc. the 7th ACL International Workshop on Semantic Evaluation. (2013)
15. Pak, A., Paroubek, P.: Twitter as a corpus for sentiment analysis and opinion mining. In: Proceedings of LREC 2010. Valletta, Malta (2010)
16. Riloff, E., Wiebe, J.: Learning extraction patterns for subjective expressions. In: Proc. the 2003 conference on Empirical methods in natural language processing (2003)
17. Saif, H., Fernandez, M., He, Y., Alani, H.: Evaluation datasets for twitter sentiment analysis a survey and a new dataset, the sts-gold. In: Proceedings, 1st ESSEM Workshop. Turin, Italy (2013)
18. Saif, H., Fernandez, M., He, Y., Alani, H.: On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter. In: Proc. 9th Language Resources and Evaluation Conference (LREC). Reykjavik, Iceland (2014)
19. Saif, H., Fernandez, M., He, Y., Alani, H.: Senticircles for contextual and conceptual semantic sentiment analysis of twitter. In: Proc. 11th Extended Semantic Web Conf. (ESWC). Crete, Greece (2014)
20. Saif, H., He, Y., Alani, H.: Alleviating data sparsity for twitter sentiment analysis. In: Proc. Workshop on Making Sense of Microposts (#MSM2012) in WWW 2012. Lyon, France (2012)
21. Saif, H., He, Y., Alani, H.: Semantic sentiment analysis of twitter. In: Proc. 11th Int. Semantic Web Conf. (ISWC). Boston, MA (2012)
22. Speriosu, M., Sudan, N., Upadhyay, S., Baldridge, J.: Twitter polarity classification with label propagation over lexical links and the follower graph. In: Proceedings of the EMNLP First workshop on Unsupervised Learning in NLP. Edinburgh, Scotland (2011)
23. Thelwall, M., Buckley, K., Paltoglou, G.: Sentiment strength detection for the social web. J. American Society for Information Science and Technology 63(1), 163–173 (2012)
24. Thet, T.T., Na, J.C., Khoo, C.S., Shakthikumar, S.: Sentiment analysis of movie reviews on discussion boards using a linguistic approach. In: Proc. the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion (2009)
25. Turney, P.: Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02). Philadelphia, Pennsylvania (2002)
26. Turney, P.D., Pantel, P., et al.: From frequency to meaning: Vector space models of semantics. Journal of artificial intelligence research 37(1), 141–188 (2010)
27. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. In: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing. Vancouver, British Columbia, Canada (2005)
28. Wittgenstein, L.: Philosophical Investigations. Blackwell, London, UK (1953, 2001)