Sentic Parser: A Graph-Based Approach to Concept Extraction for Sentiment Analysis

Erik Cambria, Rui Mao, Sooji Han, Qian Liu
School of Computer Science and Engineering, NTU, Singapore
{cambria, rui.mao, sooji.han, liu.qian}@ntu.edu.sg

***Abstract*—Concept-level sentiment analysis improves on standard word-level opinion mining by leveraging the power of multiword expressions, linguistic objects formed by two or more words that behave like 'semantic atoms' by displaying formal or functional idiosyncratic properties with respect to free word combinations. The extraction of meaningful multiword expressions from text, however, is not an easy task, as it goes beyond simple n-gram modeling. In the context of sentiment analysis, such meaningful concepts are represented by those multiword expressions with high connotative, rather than denotative, information, i.e., combination of words that convey a certain degree of subjectivity (positive or negative polarity) rather than objectivity (neutral polarity). In this work, we propose a morphology-aware concept parser for the efficient extraction and generalization of affective multiword expressions from English text. The same methodology can potentially be applied to other knowledge bases, as well as different languages and multiple modalities.**

***Index Terms*—Semantic parsing; Concept extraction; Multiword expressions; Morphology; Sentiment analysis; Natural language processing**

I. Introduction

Multiword expressions include an extremely varied set of items (from idioms to collocations, from formulae to sayings) which have been the privileged subject matter of fields such as phraseology, lexicology, lexicography, and computational linguistics. Far from being a marginal phenomenon, multiword expressions are ubiquitous and pervasive: some estimate that they are as numerous as words in some languages, which makes them as central an issue as words for the understanding of human language. However, their relation with words, and morphology, is by far less explored, not to say neglected, especially in terms of demarcation, competition, and cross-linguistic variation [1].

Multiword expressions are a key problem for the development of large-scale, linguistically sound natural language processing (NLP) technology. The various kinds of multiword expressions should be analyzed in distinct ways, including listing 'words with spaces', hierarchically organized lexicons, restricted combinatoric rules, lexical selection, 'idiomatic constructions' and simple statistical affinity. An adequate comprehensive analysis of multiword expressions must employ both symbolic and subsymbolic techniques [2], [3]. Although deep learning has enabled context modeling in natural language text, we are still far from a semantic deconstruction of text that could enable the tackling of significant AI issues such as symbol grounding, commonsense reasoning and natural language understanding.

In this work, we propose a graph-based technique for effectively and quickly identifying sentiment-bearing concepts in open English text and generalize these into primitives for advanced affective reasoning tasks. In particular, the technique draws upon SenticNet [4], a commonsense knowledge base for concept-level sentiment analysis built by means of neurosymbolic AI, and, hence, the parser is termed Sentic Parser. The same methodology, however, can potentially be applied to other knowledge bases, as well as different languages and multiple modalities. The paper is organized as follows: Section II introduces related work; Section III describes in details the algorithms for concept extraction; Section IV explains how concepts are generalized into primitives; Section V evaluates the proposed approach; finally, Section VI offers concluding remarks and discusses avenues for future work.

II. Related Work

The deconstruction of natural language text into multiple-word concepts is a key step for many NLP tasks [5]. A concept like `cloud_computing`, for example, is a semantic atom that should never be broken down into single words, or else the word `cloud` may be misinterpreted as weather-related. Same goes for colloquial expressions like `go_bananas`, which has nothing to do with fruits. Semantic parsing can be performed using a combination of syntax and semantics, via syntax alone (making use of phrase structure grammars), or statistically, using classifiers based on training algorithms. Dependency parsing offers high semantic sensitivity, the ability to extract knowledge from grammatically-incorrect text, and can use world knowledge to choose the most likely parses, but requires access to construction corpora.

The Open Mind Common Sense (OMCS) project [6] used syntactical parsing to compare natural language sentences against regular expression patterns for collecting specific pieces of commonsense knowledge. OMCS employed a purely syntactical approach encompassing stopwords, punctuation removal, word stemming to identify commonsense concepts. Later works employed weighting methods to extract contexts for concept extraction. Yu et al. [7] proposed a method for extracting concepts specialized, standardized structured text based on an entropy weight method computing the contribution of each subset of input text to the evaluation of concepts' weights. Zhao et al. [8] proposed a method based on tf-idf weighting for extracting concepts from documents retrieved from the Web for augmenting domain-specific ontology graphs.

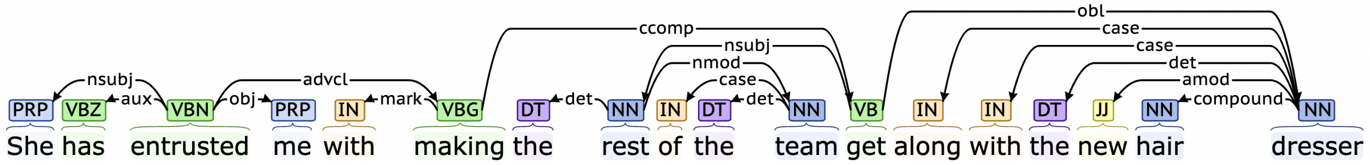

Fig. 1. A parse tree for a sample English sentence.

Mehanna et al. [9] proposed a rule-based concept extraction method using ConceptNet and incorporate it into semantic conceptualization for sentiment analysis. Semantic parsing based on part-of-speech (POS) tagging involves annotating syntactic structures with language-specific parts of speech. Related works include tag sequence probability [10] and lexical probabilities [11], but statistical parsing has been the most widely adopted technique for collecting information from text [12]–[14], together with active learning, which aims to select effective features [15], [16].

Recently, concept extraction efforts have focused on different research domains. For example, Ge et al. [17] studied source and target concept extraction for explainable metaphor identification. They proposed a dynamic reward mechanism to achieve the concept extraction without using direct annotated data. Fan et al. [18] used attentive deep neural networks to fuse both character and word embeddings and, hence, achieve clinical and biomedical concept extraction. Tohti et al. [19] extracted concept words for building a bilingual ontology, based on various statistical features, semantic and syntactic features and word segmentation techniques. Rana et al. [20] proposed a text summarization-motivated concept extraction method. The method uses K-means to discover the key concepts of a document. Shvets and Wanner [21] employed a pointer-generator network, LSTM and distant supervision to extract concepts from Wikipedia and leveraged a copy mechanism to address the issue of out-of-vocabulary words. Finally, Fang et al. [22] proposed a guided Attention concept extraction network that is supervised by additional features, e.g., title, topic, and clue words.

III. Concept Extraction

Sentic Parser is a hybrid semantic parser that leverages an ensemble of constituency and dependency parsing and a mix of stemming and lemmatization. The aim of the parser is to deconstruct text into sentences, sentences into clauses, and clauses into concepts, i.e., words or multiword expressions. The algorithm is inspired by our previous work on graph-based concept parsing [23]. The key novelties introduced by Sentic Parser are that it leverages morphology for syntactic normalization and it uses primitives for semantic normalization.

A. From Text to Verb and Noun Chunks

Sentic Parser firstly applies sentence boundary disambiguation to deconstruct text into sentences. After that, sentences are broken down into clauses by considering each verb and its associated noun phrases one by one.

For a sentence like “She has entrusted me with making the rest of the team get along with the new hair dresser”, the algorithm first individuates the verb words and multiword expressions `entrusted`, `making`, and `get_along_with`. A general assumption during clause separation is that, if a piece of text contains a preposition or subordinating conjunction, the words preceding these function words are interpreted not as events but as objects. The next step of the algorithm is to separate clauses into verb and noun chunks, as suggested by the parse tree in Fig. 1. This is done by using SenticNet concepts as a reference, that is, by leveraging the n-grams that make up SenticNet’s multiword expressions to correctly extract phrasal verbs and other ‘semantic atoms’ such as `cloud_computing`, `go_bananas`, or `pain_killer`. While this would be a monumental task for NLP in general, it is rather feasible for sentiment analysis as affect words are just a small subset of the total number of words that make up a language.

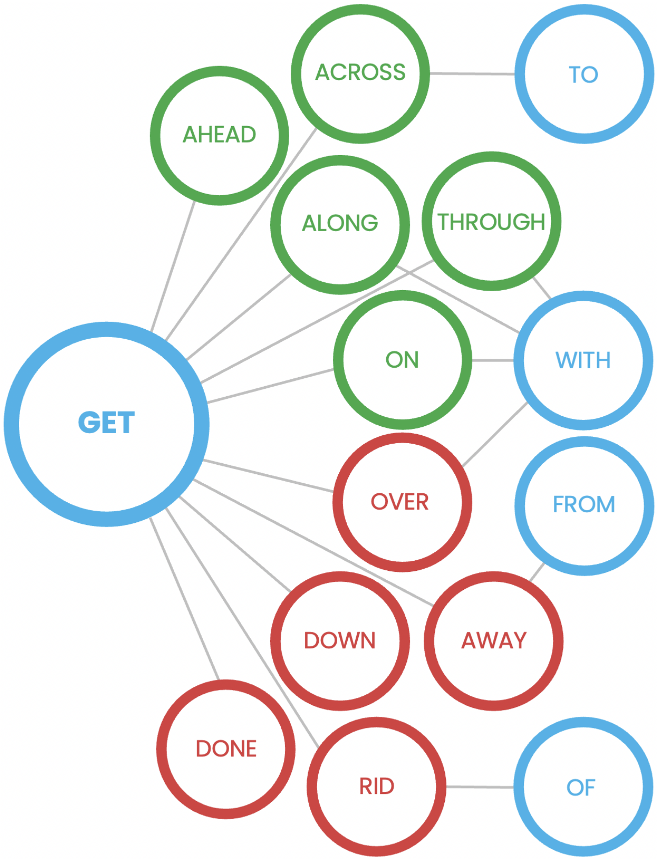

Fig. 2. A sample parse graph for multiword expressions.

```
Data: NounPhrase
Result: Valid object concepts
Split the NounPhrase into bigrams ;
Initialize concepts to Null ;
for each NounPhrase do
    while For every bigram in the NounPhrase do
        POS Tag the Bigram ;
        if adj noun then
            add to Concepts: noun, adj+noun
        else if noun noun then
            add to Concepts: noun+noun
        else if stopword noun then
            add to Concepts: noun
        else if adj stopword then
            continue
        else if stopword adj then
            continue
        else
            Add to Concepts : entire bigram
        end
        repeat until no more bigrams left ;
    end
end
```

Algorithm 1: POS-based bigram algorithm

B. Obtaining the Full List of Concepts

Next step is to normalize clauses using a lemmatization or stemming algorithm. Sentic Parser uses a hybrid approach that stems words by making sure that the resulting output is in SenticNet. The difference between a standard stemmer and Sentic Parser could be compared to the difference between k-means and k-medoids: the former focuses on distances, the latter focuses on instances. Thus, a standard stemmer would blindly apply rules such as removing canonical suffixes like *-y*, *-ed*, *-ish* to normalize words like `angry`, `refined`, and `accomplish` to `angr`, `refin`, and `accompl`, respectively. Sentic Parser, instead, does not finalize the stemming process unless the resulting word is an English word present in SenticNet, e.g., `risky` to `risk`, `blessed` to `bless`, or `foolish` to `fool`. Additionally, Sentic Parser is not triggered if the input word is already present in SenticNet to make sure that words like `slimy`, `scared`, and `flourish` are not wrongly normalized as `slim`, `scar`, and `flour`, respectively. This mechanism also allows Sentic Parser to distinguish between words that, despite sharing the same stem, have different meaning and polarity, e.g., `stunning` vs `stunned`, `blandish` vs `bland`, `bullish` vs `bully`, or `bearable` vs `bearish`.

Next, each potential *noun* chunk associated with individual verb chunks is paired with the stemmed verb in order to detect multiword expressions of the form 'verb plus object'. Objects alone, however, can also represent a concept. To detect such expressions, a POS-based n-gram algorithm checks noun phrases for stopwords and adjectives. In particular, noun phrases are first split into n-grams and then processed through POS patterns, as shown in Algorithm 1 for bigrams.

In the case of bigrams[1], POS pairs are processed as follows:

1) ADJECTIVE NOUN : An adj+noun combination and a noun as a stand-alone concept are added to the objects list.
2) ADJECTIVE STOPWORD : The entire bigram is discarded.
3) NOUN ADJECTIVE : As trailing adjectives do not tend to carry sufficient information, the adjective is discarded and only the noun is added as a valid concept.
4) NOUN NOUN : When two nouns adjacently occur in a sentence, they are considered to be part of a single concept as a multiword expression, e.g., `egg_sandwich`, `ice_cream`, and `chocolate_biscuit`.
5) NOUN STOPWORD : The stopword is discarded and only the noun is considered valid.
6) STOPWORD ADJECTIVE: The entire bigram is discarded.
7) STOPWORD NOUN : In bigrams matching this pattern, the stopword is discarded and the noun alone qualifies as a valid concept.

Next, in order to capture event concepts, matches between the object concepts and the normalized verb chunks are searched. This is done by exploiting a parse graph that maps all the multiword expressions contained in SenticNet, e.g., `get_together`, `get_on_with`, `get_over_with`, `get_to_bottom_of`, etc. (Fig. 2). Such an unweighted directed graph helps to quickly detect multiword concepts, without performing an exhaustive search throughout all the possible word combinations that can form a concept.

Single-word concepts, e.g., `hit`, that already appear in the clause as a multiword concept, e.g., `hit_economy`, `hit_wall`, `hit_roof`, `hit_button`, or `hit_parade`, are pleonastic and, hence, are discarded. In this way, Algorithm 2 is able to extract event concepts such as `slow_down_inflation`, `speed_up_erosion`, and `end_dispute`, but also idiomatic expressions such as `hit_road` (as in 'hit the road'), `cut_mustard` (as in 'cut the mustard'), `weather_storm` (as in 'weather the storm'), or `kick_bucket` (as in 'kick the bucket').

```
Data: Natural language sentence
Result: List of concepts
Find the number of verbs in the sentence ;
for every clause do
    extract VerbPhrases and NounPhrases ;
    lemmatize VERB ;
    for every NounPhrase with the associated verb do
        find forms of objects that exist in SenticNet ;
        link all objects to lemmatized verb to get events ;
    end
    repeat until no more clauses are left ;
end
```

Algorithm 2: Event concept extraction algorithm

[1] A similar procedure is used for trigrams and higher-order n-grams.

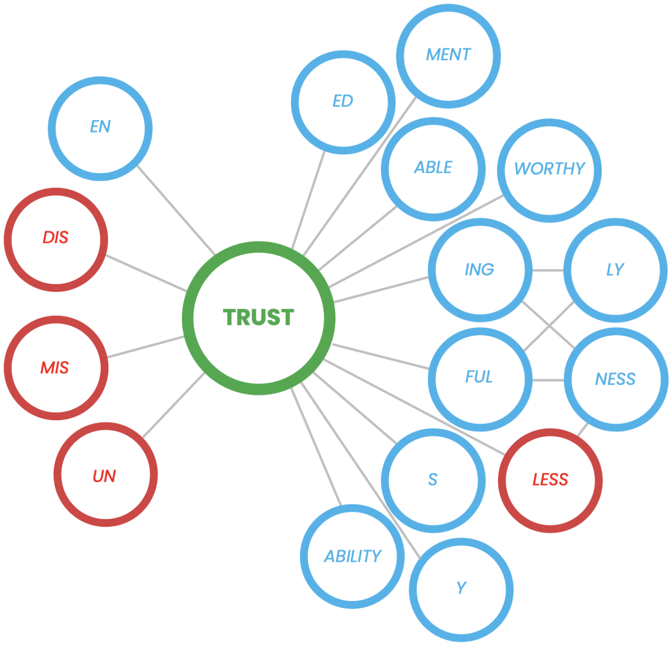

Fig. 3. A sample parse graph for word inflections.

The same graph-based approach is applied to remove inflections, such as *-ing*, *-ful*, and *-able*, and neutral prefixes, such as *en-*, *re-*, and *co-* (Fig. 3), so that words like ‘entrust’, ‘entrusts’, ‘entrusted’, ‘entrusting’, ‘entrustment’, ‘entrustments’, ‘trustable’, ‘trustability’, ‘trusts’, ‘trusted’, ‘trustful’, ‘trustfully’, ‘trustfulness’, ‘trustily’, ‘trustiness’, ‘trusting’, ‘trustingly’, ‘trustingness’, ‘trustworthy’, ‘trustworthily’, ‘trustworthiness’, etc. are all normalized to trust. The same mechanism applies to other non-canonical suffixes such as *-like*, *-hood*, *-dom*, and *-ship* so that words like ‘saintlike’, ‘sainthood’, ‘saintdom’, and ‘saintship’ are all normalized as saint.

The algorithm can also handle negative prefixes such as *mis-*, *dis-*, and *un-* so that words like ‘distrust’ and ‘mistrust’ can be normalized as NOT trust. Such negative prefix handling happens concomitantly with inflection removal so that also words like ‘distrusts’, ‘distrusted’, ‘distrustable’, ‘distrustful’, ‘distrustfully’, ‘distrustfulness’, ‘distrusting’, ‘distrustingly’, ‘mistrusts’, ‘mistrusted’, ‘mistrustable’, ‘mistrustful’, ‘mistrustfully’, ‘mistrustfulness’, ‘mistrusting’, ‘mistrustingly’, ‘trustless’, ‘untrustworthy’, ‘untrusty’, ‘untrusting’, and more, are all normalized as NOT trust.

Thanks to such a mechanism, which leverages both inflectional and derivational morphology, Sentic Parser is also able to decode wrong English expressions such as ‘stucked’, ‘accessable’, or ‘inglamorous’, which can be rather common in social media text. The same rule of checking whether a concept is present in SenticNet still applies here so that concepts like defraud, distress, and disclose are not wrongly normalized as NOT fraud, NOT stress, and NOT close, respectively. Finally, Sentic Parser also performs microtext normalization [24] so that words like ‘gooooooood’ and ‘horrrible’ are normalized as ‘good’ and ‘horrible’, respectively.

IV. Generalization into Primitives

A recent big shift in NLP research has been the upgrade from the bag-of-words (BOW) model to the continuous-bag-of-words (CBOW) model, which allowed NLP systems to take into account context in the same way one can tell what is the role of a pixel in an image based on its neighbors [25]. This same shift, however, is what had slowly turned NLP systems into black-box systems [26]. Since they are better than CBOW at preserving meaning, multiword expressions are a possible solution to reverse this trend. Nevertheless, multiword expressions are hard to discover and can cause the size of a lexicon to increase exponentially [27].

Instead of assigning polarity to millions of multiword expressions, concept-level sentiment analysis allows polarity to be inferred on the fly by combining verb primitives (e.g., SUPPORT and its semantic opposite OBSTRUCT) and noun primitives (e.g., FRIEND and its semantic opposite ENEMY), so that expressions like help_buddy, assist_pal, or stand_up_for_homeboy are all generalized as SUPPORT(FRIEND) and, thus, categorized as positive. Besides reducing lexicon size and processing time, this approach also ensures higher accuracy as compared to many statistical approaches that simply classify text based on word occurrence frequencies.

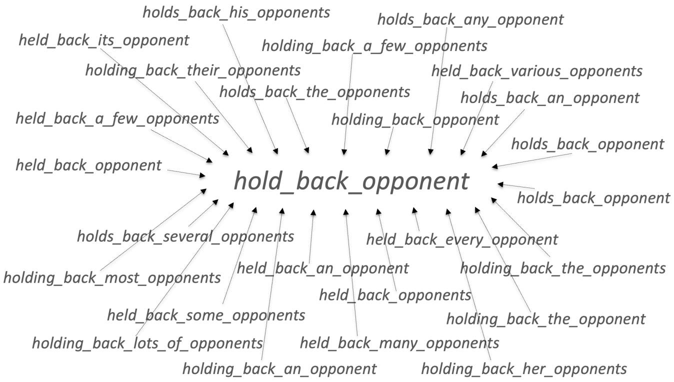

Fig. 4. An example of syntactic normalization.

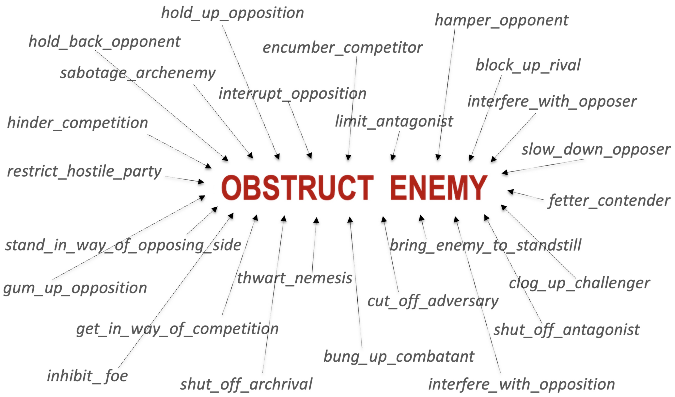

Fig. 5. An example of semantic normalization.

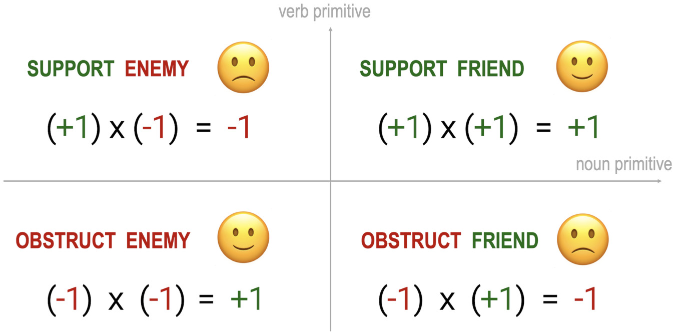

Fig. 6. An example of sentic algebra.

For example, a BOW model would classify expressions like `bring_enemy_to_standstill`, `slow_down_rival` or `stall_adversary` as negative because of the statistically negative words that compose them. In sentic computing, instead, such expressions are all generalized as the primitives `OBSTRUCT(ENEMY)` and then reasoning is performed on them. In particular, Sentic Parser firstly applies syntactic normalization (Fig. 4), e.g., it removes stopwords, it normalizes verb inflections to their infinitive forms, it replaces plurals with singulars, etc. Secondly, Sentic Parser applies semantic normalization (Fig. 5), i.e., it generalizes both verbs and nouns to their corresponding primitives so that sentic algebra (Fig. 6) can be applied on them.

In this way, concept-level sentiment analysis reduces the symbol grounding problem and, hence, gets one step closer to natural language understanding. We assign a label to each subset by selecting the most typical of the terms. In the positive subset {`add`, `soar`, `increase`, `escalate`, `mount_up`, ...}, for example, the term with the highest occurrence frequency is `increase`. Hence, the subset is named after it, i.e., `INCREASE`, and later defined manually using logic, i.e., `INCREASE(x):= x + 1`. Likewise, the corresponding negative subset is termed `DECREASE` and defined as `DECREASE(x):= x - 1`. Primitives like `INCREASE` and `DECREASE` are Level-0 primitives (or superprimitives) because they are 'grounded' using logic. Primitives defined in terms of these, e.g., `GROW:= INCREASE(SIZE)`, are Level-1 primitives. Primitives defined in terms of Level-1 primitives, e.g., `LENGTHEN:= GROW(LENGTH)`, are Level-2 primitives and so on (Fig. 7).

A. Sentic Paths

In the era of deep learning, semantics is likely represented as embeddings and the semantic similarity is measured in vector space [28]–[32]. However, symbolic- or graph-based similarity detection takes the advantage of explainability. Several measures were proposed for WordNet, such as Tversky's measure [33], Resnik's Measure [34], Wu & Palmer's Measure [35], and the shortest path [36]. To the best of our knowledge, there is no similar work for affective knowledge bases such as SenticNet.

To this end, we introduce sentic paths, a cognitive-inspired algorithm that takes into account the topology of affective data in a multidimensional vector space of commonsense knowledge. Sentic paths are a kernel method conceived to find smooth paths between objects in space through a number of waypoints (N_c). The main feature of the method is that the obtained path aims to move through high probability regions of the space, searching for a geodetic whose underlying topology is ruled by the samples' probability. This method aspires to mimic the cognitive intuition for which thinking is the process of moving from one concept to another through regions of the space where there is a high probability of finding other concepts [37].

In particular, we employ the plain feature space (linear kernel, primal problem). Rather than a distance, sentic paths calculate a discrete path between a primitive concept p_0 and its semantic opposite p_{N_c+1} throughout the vector space manifolds. While the shortest path (through the pure Euclidean distance) between two antithetic primitives risks to include many irrelevant concepts, a path that follows the topological structure of the vector space from a positive primitive (e.g., p_0=`ACCEPT`) to its semantic antithesis (e.g., p_{N_c+1}=`REJECT`) is more likely to contain concepts that are both semantically and affectively relevant. Because positive and negative concepts are found in diametrically opposite areas of the vector space [38], sentic paths always traverse it from one end to the other (Fig. 8).

This ensures the discovery of concepts that are both semantically and affectively related to both the positive primitive p_0 (e.g., `welcome`, `agree`, and `take_in`) and the negative one p_{N_c+1} (e.g., `refuse`, `turn_down`, and `deny`). To adapt the algorithm to the context of sentiment analysis, we employ a metric based on the Hourglass model [39], a biologically-inspired and psychologically-motivated emotion categorization model based on four independent but concomitant affective dimensions, which can potentially describe the full range of emotional experiences that are rooted in any of us. The core steps of the algorithm can be summarized as it follows:

1) *Sentic path initialization:* given the starting and the ending primitives p_0 and p_{N_c+1}, the Dijkstra algorithm is run over a penalized graph obtained by computing the penalized distance matrix among all the concepts c_i in $\mathbf{C}$ as follows:

$$d_p^2(c_i,c_j) = \begin{cases} d^2(c_i,c_j), & c_i \in \mathrm{nn}_k(c_j) \\ td^2(c_i,c_j), & \text{otherwise} \end{cases}$$

where $\mathrm{nn}_k(c_j)$ is the nearest neighbors set and t is a penalization factor. This approach allows to capture the manifold and avoid shortcuts.

2) *Waypoint concept positioning:* the Dijkstra algorithm is run on the penalized distance matrix and some intermediate concepts are returned. This path is then reparameterized to obtained equally distanced points.

3) *Cost function optimization*: the path is smoothed through a cost function optimized via the EM algorithm. The waypoint concept configuration $\mathbf{P}_{init}$ from the previous step is used as waypoint concept initialization and as the input matrix $\mathbf{C}$ ($\mathbf{P}_{init} = \mathbf{C}$).

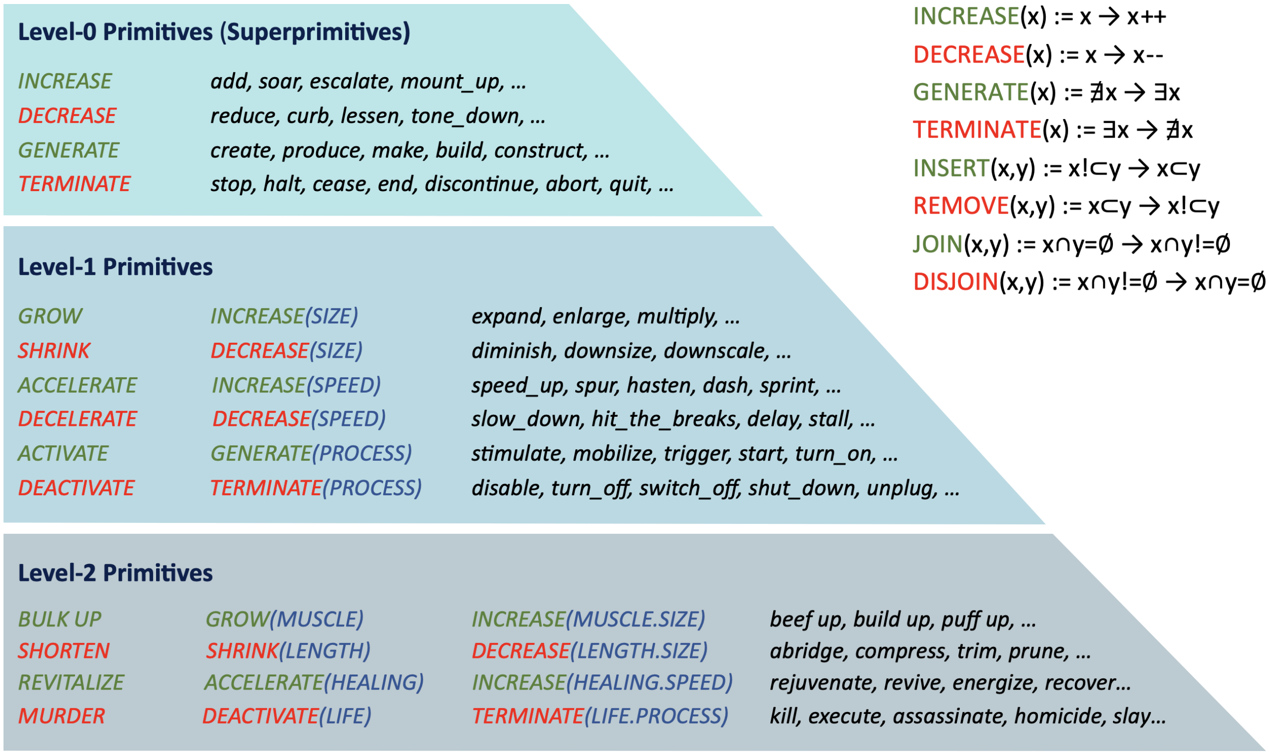
Level-0 Primitives (Superprimitives)
INCREASE add, soar, escalate, mount_up, …
DECREASE reduce, curb, lessen, tone_down, …
GENERATE create, produce, make, build, construct, …
TERMINATE stop, halt, cease, end, discontinue, abort, quit, …
Level-1 Primitives
GROW INCREASE(SIZE) expand, enlarge, multiply, …
SHRINK DECREASE(SIZE) diminish, downsize, downscale, …
ACCELERATE INCREASE(SPEED) speed_up, spur, hasten, dash, sprint, …
DECELERATE DECREASE(SPEED) slow_down, hit_the_breaks, delay, stall, …
ACTIVATE GENERATE(PROCESS) stimulate, mobilize, trigger, start, turn_on, …
DEACTIVATE TERMINATE(PROCESS) disable, turn_off, switch_off, shut_down, unplug, …
Level-2 Primitives
BULK UP GROW(MUSCLE) INCREASE(MUSCLE.SIZE) beef up, build up, puff up, …
SHORTEN SHRINK(LENGTH) DECREASE(LENGTH.SIZE) abridge, compress, trim, prune, …
REVITALIZE ACCELERATE(HEALING) INCREASE(HEALING.SPEED) rejuvenate, revive, energize, recover…
MURDER DEACTIVATE(LIFE) TERMINATE(LIFE.PROCESS) kill, execute, assassinate, homicide, slay…
INCREASE(x) := x → x++
DECREASE(x) := x → x--
GENERATE(x) := ∄x → ∃x
TERMINATE(x) := ∃x → ∄x
INSERT(x,y) := x!⊂y → x⊂y
REMOVE(x,y) := x⊂y → x!⊂y
JOIN(x,y) := x∩y=∅ → x∩y!=∅
DISJOIN(x,y) := x∩y!=∅ → x∩y=∅

Fig. 7. Primitives hierarchy.

The cost function, hence, is:

$$\min_{P,u} \sum_{i=1}^{N_c} \sum_{j=1}^{N_c} \|c_i - p_j\|^2 \delta(u_i, j) + s \sum_{i=0}^{N_c} \|p_{i+1} - p_i\|^2 \quad (1)$$

where $\delta(u_i, j)$ is a Kronecker delta to rule the waypoint membership and s is a regularization coefficient. Hence, the method is an out-of-sample smooth extension of Dijkstra shortest path, where the underlying graph is ruled by a penalized Euclidean metric and whose smoothness is ruled by s.

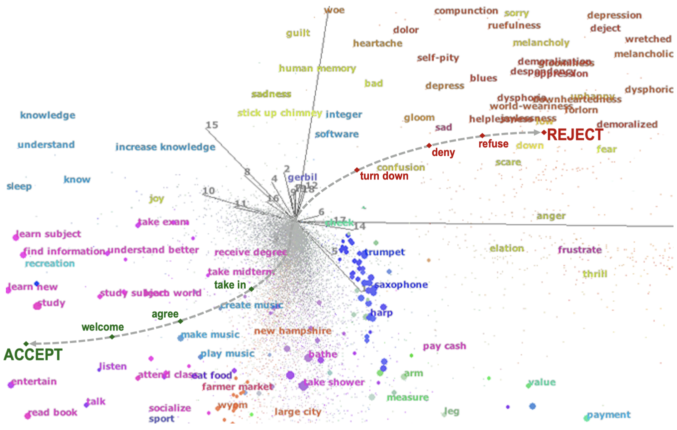

Fig. 8. Sentic path between ACCEPT and REJECT.

V. Evaluation

Ten benchmark datasets for sentiment analysis were considered for evaluating Sentic Parser, available as an application programming interface within the Sentic API Suite[2]. In particular, the proposed set of algorithms was compared against SpaCy Parser, a tool built upon SpaCy dependency parser[3] and SpaCy lemmatizer[4] for deconstructing natural language text into words and multiword expressions. Both parsers were coupled with ten popular sentiment lexica for the task of binary polarity classification. The lexica are WordNet-Affect [40], Micro WNOp [41], SentiStrength [42], Opinion Finder [43], General Inquirer [44], SentiWords [45], NOVAD [46], SO-CAL [47], HSSWE [48] and SenticNet [4]. We tested the combination of parsers and lexica on 10 well-known sentiment analysis datasets, namely: CR [49], MR [50], Amazon [51], IMDb [52], Sanders [53], SST [54], STS [55], SE13 [56], SE15 [57], and SE16 [58]. Results are listed in Table I.

Sentic Parser showed an average accuracy improvement of 8.55% over polarity detection performed without parsing. This is due to the fact that many lexica only contain affect words in their standard form, e.g., infinitive for verbs and singular for nouns, while Sentic Parser is able to replace all out-of-vocabulary inflections with their corresponding in-vocabulary root words.

[2] https://sentic.net/api

[3] https://spacy.io/api/dependencyparser

[4] https://spacy.io/api/lemmatizer

Lexicon	Parser	CR	MR	Amazon	IMDb	Sanders	SST	STS	SE13	SE15	SE16	Average
WordNet-Affect	No Parser	03.91%	04.23%	08.46%	17.84%	07.63%	03.92%	15.52%	09.26%	07.46%	05.13%	08.34%
	SpaCy Parser	04.25%	04.81%	12.35%	22.30%	11.92%	04.53%	19.00%	10.54%	11.82%	06.22%	10.78%
	Sentic Parser	04.61%	05.05%	18.87%	28.99%	17.81%	04.81%	24.23%	15.92%	16.35%	10.54%	14.72%
Micro WNOp	No Parser	10.86%	10.42%	35.61%	40.86%	14.03%	10.55%	17.92%	15.30%	16.93%	10.04%	18.26%
	SpaCy Parser	15.24%	13.60%	39.07%	43.23%	19.87%	12.98%	22.00%	19.74%	22.67%	13.92%	22.24%
	Sentic Parser	20.39%	18.73%	44.48%	49.17%	22.95%	17.64%	28.13%	24.89%	26.58%	18.41%	27.14%
SentiStrength	No Parser	35.40%	31.83%	50.13%	51.24%	38.77%	33.25%	48.37%	34.09%	35.21%	27.68%	38.60%
	SpaCy Parser	39.22%	35.46%	53.42%	54.36%	41.60%	36.21%	52.65%	36.18%	38.44%	30.00%	41.76%
	Sentic Parser	45.69%	41.72%	59.09%	60.18%	47.87%	41.57%	58.49%	42.32%	45.60%	35.46%	47.80%
Opinion Finder	No Parser	51.98%	50.11%	50.07%	47.82%	42.36%	51.00%	51.23%	42.67%	43.89%	36.19%	46.74%
	SpaCy Parser	56.34%	53.06%	53.11%	52.66%	45.17%	55.32%	54.80%	44.95%	47.66%	39.85%	50.30%
	Sentic Parser	62.05%	59.98%	59.48%	58.75%	51.22%	61.86%	60.72%	50.28%	53.57%	44.21%	56.22%
General Inquirer	No Parser	45.98%	44.05%	48.81%	50.44%	39.27%	45.86%	44.13%	39.30%	41.97%	32.45%	43.23%
	SpaCy Parser	50.72%	47.18%	53.26%	53.28%	42.66%	50.02%	48.26%	41.70%	45.05%	34.91%	46.71%
	Sentic Parser	56.56%	53.76%	59.63%	59.43%	46.81%	54.39%	54.59%	47.82%	51.12%	40.88%	52.50%
SentiWords	No Parser	53.82%	48.01%	47.93%	47.14%	42.80%	51.00%	50.92%	47.83%	47.30%	44.79%	48.16%
	SpaCy Parser	56.09%	52.37%	52.54%	52.03%	47.21%	54.63%	54.60%	52.71%	52.84%	48.03%	52.31%
	Sentic Parser	62.71%	58.65%	58.11%	57.29%	53.59%	60.57%	60.44%	58.82%	57.46%	54.38%	58.21%
NOVAD	No Parser	56.09%	47.00%	48.90%	49.52%	45.92%	50.76%	53.26%	53.39%	49.12%	49.99%	50.40%
	SpaCy Parser	58.12%	50.88%	51.53%	50.64%	46.20%	53.13%	55.98%	55.78%	51.61%	52.76%	52.67%
	Sentic Parser	64.88%	56.91%	57.06%	56.81%	51.06%	58.88%	61.55%	61.10%	57.87%	58.16%	58.43%
SO-CAL	No Parser	57.33%	58.17%	66.73%	69.72%	46.30%	58.16%	55.78%	34.31%	30.63%	35.75%	51.29%
	SpaCy Parser	59.92%	59.66%	69.11%	72.91%	49.22%	61.24%	57.44%	35.86%	31.55%	36.56%	53.35%
	Sentic Parser	65.58%	64.58%	75.86%	78.67%	52.78%	67.33%	63.51%	41.15%	37.63%	41.02%	58.82%
HSSWE	No Parser	63.08%	54.87%	59.44%	57.93%	64.66%	56.77%	68.99%	60.32%	57.09%	59.04%	60.22%
	SpaCy Parser	65.54%	55.32%	61.97%	59.55%	67.81%	59.94%	72.23%	62.41%	60.61%	60.38%	62.58%
	Sentic Parser	71.33%	60.61%	67.08%	65.27%	73.94%	63.15%	78.27%	68.67%	64.83%	66.62%	67.98%
SenticNet	No Parser	73.52%	68.29%	73.01%	72.54%	72.98%	70.03%	80.92%	74.42%	73.90%	75.22%	73.49%
	SpaCy Parser	77.11%	71.15%	74.98%	76.88%	74.32%	73.95%	84.04%	77.18%	75.13%	79.10%	76.39%
	Sentic Parser	83.60%	77.04%	81.53%	82.91%	80.54%	78.71%	90.08%	83.69%	81.67%	84.39%	82.42%

TABLE I

COMPARISON OF SEMANTIC PARSERS IN COMBINATION WITH TEN POPULAR LEXICA ON TEN BENCHMARK SENTIMENT DATASETS.

Sentic Parser also displayed an average performance improvement of 5.52% over SpaCy Parser, as this does not tackle problems such as negation handling, microtext normalization, and compound word processing. Moreover, Sentic Parser can handle nested affixes, e.g., normalize expressions like ‘threateningly’ to ‘threat’ and ‘superdupermegagood’ to ‘good’, double negations, e.g., replace words like ‘undiscouraged’ with ‘courage’, and prefixes like *co-*, *pre-*, *fore-*, and *super-* which may alter the meaning but not the polarity of root words so that concepts like ‘coordinated’, ‘coordinating’, ‘coordinatedly’, ‘coordinatingly’, ‘coordination’, ‘coordinations’, ‘preordinate’, ‘preordinated’, ‘preordinating’, ‘foreordinated’, ‘foreordinating’, ‘superordinated’, ‘superordinating’, etc. are all normalized as `ordinate`.

Finally, Sentic Parser can be applied to different languages and multiple modalities: we employed the same methodology to BabelSenticNet [59] and PhonSenticNet [60] (instead of standard SenticNet) and obtained comparable results for alphabetic languages. More experiments are to be carried out for ideographic languages as future work.

VI. CONCLUSION

In this paper, we proposed Sentic Parser, a knowledge-specific concept parser based on SenticNet, which leverages both inflectional and derivational morphology for the efficient extraction and generalization of affective multiword expressions from English text. In particular, Sentic Parser is a hybrid semantic parser that uses an ensemble of constituency and dependency parsing and a mix of stemming and lemmatization. The key novelties introduced by this parser with respect to our previous work are that it leverages morphology for syntactic normalization and it uses primitives for semantic normalization. We showed that Sentic Parser is superior to standard concept parsers because it focuses on the extraction of root words from text and, hence, it normalizes complex natural language constructs to a sort of primitive-based protolanguage. This is done by leveraging a graph-based approach which enables nested affix handling, microtext normalization, and compound word processing. Preliminary experiments on other alphabetic languages showed that the proposed methodology could be language-agnostic.

ACKNOWLEDGMENTS

This research is supported by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funding Scheme (Project #A18A2b0046).

REFERENCES

[1] F. Masini, “Multi-word expressions and morphology,” *Oxford Research Encyclopedia of Linguistics*, 2019.

[2] I. Sag, T. Baldwin, F. Bond, A. Copestake, and D. Flickinger, “Multiword expressions: A pain in the neck for nlp,” in *CICLing*, 02 2002.

[3] R. Mao, X. Li, M. Ge, and E. Cambria, “MetaPro: A computational metaphor processing model for text pre-processing,” *Information Fusion*, vol. 86-87, pp. 30–43, 2022.

[4] E. Cambria, Q. Liu, S. Decherchi, F. Xing, , and K. Kwok, “SenticNet 7: A commonsense-based neurosymbolic AI framework for explainable sentiment analysis,” in *LREC*, 2022, pp. 3829–3839.

[5] E. Cambria, B. Schuller, B. Liu, H. Wang, and C. Havasi, “Statistical approaches to concept-level sentiment analysis,” *IEEE Intelligent Systems*, vol. 28, no. 3, pp. 6–9, 2013.

[6] R. Speer, J. Chin, and C. Havasi, “ConceptNet 5.5: An open multilingual graph of general knowledge,” in *AAAI*, 2017, pp. 4444–4451.

[7] J. Yu, R. Chen, L. Xu, and D. Wang, "Concept extraction for structured text using entropy weight method," in *2019 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2019, pp. 1–6.

[8] G. Zhao and X. Zhang, "Domain-specific ontology concept extraction and hierarchy extension," in *Proceedings of the 2nd International Conference on Natural Language Processing and Information Retrieval*, 2018, pp. 60–64.

[9] Y. S. Mehanna and M. B. Mahmuddin, "A semantic conceptualization using tagged bag-of-concepts for sentiment analysis," *IEEE Access*, vol. 9, pp. 118 736–118 756, 2021.

[10] G. Carroll and E. Charniak, "Two experiments on learning probabilistic dependency grammars from corpora," Department of Computer Science, Univ., AAAI Technical Report WS-92-01, 1992.

[11] K. Toutanova, D. Klein, C. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *NAACL*, Stroudsburg, 2003, pp. 173–180.

[12] E. Charniak, "Statistical parsing with a context-free grammar and word statistics," in *AAAI*, Providence, 1997, pp. 598–603.

[13] D. Chen and C. D. Manning, "A fast and accurate dependency parser using neural networks," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 740–750.

[14] T. Bladier, J. Waszczuk, and L. Kallmeyer, "Statistical parsing of tree wrapping grammars," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 6759–6766.

[15] R. Hwa, "Sample selection for statistical grammar induction," in *EMNLP*, Hong Kong, 2000, pp. 45–52.

[16] M. Tang, X. Luo, S. Roukos *et al.*, "Active learning for statistical natural language parsing," in *ACL*, Philadelphia, 2002, pp. 120–127.

[17] M. Ge, R. Mao, and E. Cambria, "Explainable metaphor identification inspired by conceptual metaphor theory," in *AAAI*, 2022, pp. 10 681–10 689.

[18] S. Fan, H. Yu, X. Cai, Y. Geng, G. Li, W. Xu, X. Wang, and Y. Yang, "Multi-attention deep neural network fusing character and word embedding for clinical and biomedical concept extraction," *Information Sciences*, vol. 608, pp. 778–793, 2022.

[19] T. Tohti, L. Chang, A. Hamdulla, and H. Yilahun, "Concept word extraction for bilingual ontology construction in unstructured text environment," in *PRML*. IEEE, 2021, pp. 273–278.

[20] M. M. R. Rana, R. Afrin, M. A. Rahman, A. Haque, and M. A. Rahman, "Concept extraction from ambiguous text document using k-means," *International Research Journal of Engineering and Technology*, vol. 06, pp. 5317–5330, 2019.

[21] A. Shvets and L. Wanner, "Concept extraction using pointer-generator networks and distant supervision for data augmentation," in *EKAW*. Springer, 2020, pp. 120–135.

[22] S. Fang, Z. Huang, M. He, S. Tong, X. Huang, Y. Liu, J. Huang, and Q. Liu, "Guided attention network for concept extraction," in *IJCAI*, 2021, pp. 1449–1455.

[23] D. Rajagopal, E. Cambria, D. Olsher, and K. Kwok, "A graph-based approach to commonsense concept extraction and semantic similarity detection," in *WWW*, 2013, pp. 565–570.

[24] R. Satapathy, E. Cambria, A. Nanetti, and A. Hussain, "A review of shorthand systems: From brachygraphy to microtext and beyond," *Cognitive Computation*, vol. 12, no. 4, pp. 778–792, 2020.

[25] E. Cambria and B. White, "Jumping NLP curves: A review of natural language processing research," *IEEE Computational Intelligence Magazine*, vol. 9, no. 2, pp. 48–57, 2014.

[26] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.

[27] F. Xing, F. Pallucchini, and E. Cambria, "Cognitive-inspired domain adaptation of sentiment lexicons," *Information Processing and Management*, vol. 56, no. 3, pp. 554–564, 2019.

[28] E. Cambria, T. Mazzocco, A. Hussain, and C. Eckl, "Sentic medoids: Organizing affective common sense knowledge in a multi-dimensional vector space," in *Advances in Neural Networks*, ser. Lecture Notes in Computer Science, D. Liu, H. Zhang, M. Polycarpou, C. Alippi, and H. He, Eds., vol. 6677. Springer-Verlag, 2011, pp. 601–610.

[29] R. Mao, C. Lin, and F. Guerin, "Word embedding and WordNet based metaphor identification and interpretation," in *ACL*, 2018, pp. 1222–1231.

[30] Q. Liu, H.-Y. Huang, Y. Gao, X. Wei, Y. Tian, and L. Liu, "Task-oriented word embedding for text classification," in *COLING*, 2018, pp. 2023–2032.

[31] E. Cambria, Y. Song, H. Wang, and N. Howard, "Semantic multi-dimensional scaling for open-domain sentiment analysis," *IEEE Intelligent Systems*, vol. 29, no. 2, pp. 44–51, 2014.

[32] Q. Liu, H. Huang, J. Xuan, G. Zhang, Y. Gao, and J. Lu, "A fuzzy word similarity measure for selecting top-k similar words in query expansion," *IEEE Transactions on Fuzzy Systems*, vol. 29, pp. 2132–2144, 2020.

[33] A. Tversky, "Features of similarity." *Psychological Review*, vol. 84, no. 4, p. 327, 1977.

[34] P. Resnik, "Using information content to evaluate semantic similarity in a taxonomy," in *IJCAI*, 1995, pp. 448–453.

[35] Z. Wu and M. Palmer, "Verb semantics and lexical selection," in *ACL*, 1994, pp. 133–138.

[36] G. Varelas, E. Voutsakis, P. Raftopoulou, E. G. Petrakis, and E. E. Milios, "Semantic similarity methods in WordNet and their application to information retrieval on the web," in *WIDM*, 2005, pp. 10–16.

[37] E. Ragusa, P. Gastaldo, R. Zunino, M. J. Ferrarotti, W. Rocchia, and S. Decherchi, "Cognitive insights into sentic spaces using principal paths," *Cognitive Computation*, vol. 11, no. 5, pp. 656–675, 2019.

[38] E. Cambria, J. Fu, F. Bisio, and S. Poria, "AffectiveSpace 2: Enabling affective intuition for concept-level sentiment analysis," in *AAAI*, 2015, pp. 508–514.

[39] Y. Susanto, A. Livingstone, B. C. Ng, and E. Cambria, "The hourglass model revisited," *IEEE Intelligent Systems*, vol. 35, pp. 96–102, 2020.

[40] C. Strapparava and A. Valitutti, "WordNet-Affect: An affective extension of WordNet," in *LREC*, 2004, pp. 1083–1086.

[41] S. Cerini, V. Compagnoni, A. Demontis, M. Formentelli, and C. Gandini, "Micro-WNOp: A gold standard for the evaluation of automatically compiled lexical resources for opinion mining," *Language Resources and Linguistic Theory*, pp. 200–210, 2007.

[42] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas, "Sentiment strength detection in short informal text," *Journal of the American Society for Information Science and Technology*, vol. 61, pp. 2544–2558, 2010.

[43] T. Wilson, P. Hoffmann, S. Somasundaran, J. Kessler, J. Wiebe, Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan, "Opinionfinder: A system for subjectivity analysis," in *HLT/EMNLP*, 2005, pp. 34–35.

[44] P. Stone, D. Dunphy, M. Smith, and D. Ogilvie, *The General Inquirer: A Computer Approach to Content Analysis*. MIT Press, 1966.

[45] L. Gatti, M. Guerini, and M. Turchi, "SentiWords: Deriving a high precision and high coverage lexicon for sentiment analysis," *IEEE Transactions on Affective Computing*, vol. 7, no. 4, pp. 409–421, 2016.

[46] A. B. Warriner et al., "Norms of valence, arousal, and dominance for 13,915 english lemmas," *Behavior Research Methods*, vol. 45, pp. 1191–1207, 2013.

[47] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Computational linguistics*, vol. 37, no. 2, pp. 267–307, 2011.

[48] L. Wang and R. Xia, "Sentiment lexicon construction with representation learning based on hierarchical sentiment supervision," in *EMNLP*, 2017, pp. 502–510.

[49] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *SIGKDD*, 2004, pp. 168–177.

[50] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *ACL*, Ann Arbor, 2005, pp. 115–124.

[51] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification," in *ACL*, vol. 7, 2007, pp. 440–447.

[52] A. Maas, R. Daly, P. Pham, D. Huang, A. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *ACL*, 2011, pp. 142–150.

[53] S. Analytics, "Sanders dataset," 2012. [Online]. Available: http://sananalytics.com/lab

[54] 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 *EMNLP*, 2013, pp. 1631–1642.

[55] H. Saif, M. Fernandez, Y. He, and H. Alani, "Evaluation datasets for Twitter sentiment analysis: a survey and a new dataset, the STS-Gold," in *AIxIA*, 2013.

[56] P. Nakov, S. Rosenthal, Z. Kozareva, V. Stoyanov, A. Ritter, and T. Wilson, "SemEval-2013 task 2: Sentiment analysis in Twitter," in *SemEval*, 2013, pp. 312–320.

[57] S. Rosenthal, P. Nakov, S. Kiritchenko, S. Mohammad, A. Ritter, and V. Stoyanov, "Semeval-2015 task 10: Sentiment analysis in Twitter," in *SemEval*, 2015, pp. 451–463.

[58] P. Nakov, A. Ritter, S. Rosentha, F. Sebastiani, and V. Stoyanov, "Semeval-2016 task 4: Sentiment analysis in Twitter," in *SemEval*, 2016.

[59] D. Vilares, H. Peng, R. Satapathy, and E. Cambria, "Babelsenticnet: A commonsense reasoning framework for multilingual sentiment analysis," in *IEEE SSCI*, 2018, pp. 1292–1298.

[60] R. Satapathy, A. Singh, and E. Cambria, "PhonSenticNet: A cognitive approach to microtext normalization for concept-level sentiment analysis," in *CSoNet*, 2019, pp. 177–188.