The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

SenticNet 5: Discovering Conceptual Primitives for Sentiment Analysis by Means of Context Embeddings

Erik Cambria,[†] Soujanya Poria,[†] Devamanyu Hazarika,[‡] Kenneth Kwok[⋆]
[†]School of Computer Science and Engineering, Nanyang Technological University
[‡]School of Computing, National University of Singapore
[⋆]Institute of High Performance Computing, A*STAR

Abstract

With the recent development of deep learning, research in AI has gained new vigor and prominence. While machine learning has succeeded in revitalizing many research fields, such as computer vision, speech recognition, and medical diagnosis, we are yet to witness impressive progress in natural language understanding. One of the reasons behind this unmatched expectation is that, while a bottom-up approach is feasible for pattern recognition, reasoning and understanding often require a top-down approach. In this work, we couple sub-symbolic and symbolic AI to automatically discover conceptual primitives from text and link them to commonsense concepts and named entities in a new three-level knowledge representation for sentiment analysis. In particular, we employ recurrent neural networks to infer primitives by lexical substitution and use them for grounding common and commonsense knowledge by means of multi-dimensional scaling.

Introduction

Recently, AI has been the acronym on everyone's lips. Although nobody knows when the so-called AI revolution will actually take place, the AI gold rush has become increasingly intense in the past few years. Most of what is considered AI today is actually sub-symbolic AI, i.e., machine learning: an extremely powerful tool for exploring large amounts of data and, for instance, making predictions, suggestions, and categorizations based on them. Machine learning, however, suffers from three big issues, namely:

1. Dependency: it requires (a lot of) training data and is domain-dependent;
2. Consistency: different training or tweaking leads to different results;
3. Transparency: the reasoning process is unintelligible (black-box algorithms).

In the context of natural language processing (NLP), these issues are particularly crucial because, unlike in other fields, they prevent AI from achieving human-like performance. To this end, AI researchers need to bridge the gap between statistical NLP and many other disciplines that are necessary for understanding human language, such as linguistics, commonsense reasoning, and affective computing.

They will have to develop an approach to NLP that is both top-down and bottom-up: top-down for the fact that they should leverage symbolic models (e.g., semantic networks) to encode meaning; bottom-up because they should use sub-symbolic methods (e.g., neural networks) to infer syntactic patterns from data. Coupling symbolic and sub-symbolic AI is key for stepping forward in the path from NLP to natural language understanding. Relying solely on machine learning, in fact, is simply useful to make a 'good guess' based on past experience, because sub-symbolic methods only encode correlation and their decision-making process is merely probabilistic. Natural language understanding, however, requires much more than that. To use Noam Chomsky's words, "you do not get discoveries in the sciences by taking huge amounts of data, throwing them into a computer and doing statistical analysis of them: that's not the way you understand things, you have to have theoretical insights".

In this work, we propose an ensemble of symbolic and sub-symbolic AI techniques to perform sentiment analysis, a NLP problem that has raised growing interest within both the scientific community, for the many exciting open challenges, as well as the business world, due to the remarkable benefits to be had from marketing and financial prediction (Cambria et al. 2017). In particular, we employ a long short-term memory (LSTM) network (Hochreiter and Schmidhuber 1997) to discover verb-noun primitives by lexical substitution, and, hence, build a new three-level knowledge representation for sentiment analysis, termed SenticNet 5. SenticNet 5 encodes the denotative and connotative information commonly associated with real-world objects, actions, events, and people. It steps away from blindly using keywords and word co-occurrence counts, and instead relies on the implicit meaning associated with commonsense concepts. Superior to purely syntactic techniques, SenticNet 5 can detect subtly expressed sentiments by enabling the analysis of multiword expressions that do not explicitly convey emotion, but are instead related to concepts that do so.

The rest of the paper is organized as follows: firstly, we present related works in the field of sentiment analysis and explain the importance of conceptual primitives for this task; secondly, we describe in detail how to discover primitives and how to link them to concepts and entities; finally, we propose experimental results on several benchmark datasets and provide concluding remarks.

Related Work

Sentiment analysis systems can be broadly categorized into knowledge-based and statistics-based systems.While the use of knowledge bases was initially more popular for the identification of sentiment polarity in text, recently sentiment analysis researchers have been increasingly using statistics-based approaches, with a special focus on supervised statistical methods. Pang et al. (Pang, Lee, and Vaithyanathan 2002) pioneered this trend by comparing the performance of different machine learning algorithms on a movie review dataset and obtained 82% accuracy for polarity detection.

A recent approach by Socher et al. (Socher et al. 2013b) obtained 85% accuracy on the same dataset using a recursive neural tensor network (NTN). Yu and Hatzivassiloglou (Yu and Hatzivassiloglou 2003) used semantic orientation of words to identify polarity at sentence level. Melville et al. (Melville, Gryc, and Lawrence 2009) developed a framework that exploits word-class association information for domain-dependent sentiment analysis.

More recent studies exploit microblogging text or Twitter-specific features such as emoticons, hashtags, URLs, @symbols, capitalizations, and elongations to enhance sentiment analysis of tweets. Tang et al. (Tang et al. 2014a) used a convolutional neural network (CNN) to obtain word embeddings for words frequently used in tweets and Dos Santos et al. (dos Santos and Gatti 2014) employed a deep CNN for sentiment detection in short texts. Recent approaches also focus on developing word embeddings based on sentiment corpora (Tang et al. 2014b). Such word vectors include more affective clues than regular word vectors and produce better results for tasks such as emotion recognition (Poria et al. 2017), sarcasm detection (Poria et al. 2016) and aspect extraction (Poria, Cambria, and Gelbukh 2016).

By relying on large semantic knowledge bases, such approaches step away from the blind use of keywords and word co-occurrence counts, relying instead on the implicit features associated with natural language concepts. Unlike purely syntactic techniques, concept-based approaches are also able to detect sentiments expressed in a subtle manner; e.g., through the analysis of concepts that do not explicitly convey any emotion, but which are implicitly linked to other concepts that do so.

The bag-of-concepts model can represent semantics associated with natural language much better than bag-of-words. In the latter, in fact, concepts like `pretty_ugly` or `sad_smile` would be split into two separate words, disrupting both semantics and polarity of the input sentence.

The Importance of Conceptual Primitives

The main limitation of concept-level sentiment analysis and the bag-of-concepts model is that they cannot achieve a comprehensive coverage of meaningful concepts, i.e., a full list of multiword expressions that actually make sense (e.g., verb-noun pairs). Semantic parsing and n-gram models have taken a bottom-up approach to solve this issue by automatically extracting concepts from raw data. The resulting multiword expressions, however, are prone to errors due to both richness and ambiguity of natural language.

A more effective way to overcome this hurdle is to take a top-down approach by generalizing semantically-related concepts, such as `munch_toast` and `slurp_noodles`, into a conceptual primitive, such as `EAT_FOOD`. In this way, most concept inflections can be captured by the knowledge base: verb concepts like `ingest`, `slurp`, `munch` are all represented by their conceptual primitive `EAT` while noun concepts like `pasta`, `noodles`, `steak` are replaced with their ontological parent `FOOD`. The idea behind this generalization is that there is a finite set of mental primitives for affect-bearing concepts and a finite set of principles of mental combination governing their interaction.

It is inherent to human nature to try to categorize things, events and people, finding patterns and forms they have in common. One of the most intuitive ways to relate two entities is through their similarity. According to Gestalt theory (Smith 1988), similarity is one of six principles that guide human perception of the world. Similarity is a quality that makes one thing or person like another and 'similar' means having characteristics in common. There are many ways in which objects can be perceived as similar, based on things like color, shape, size and texture. If we move away from mere visual stimuli, we can apply the same principles to define similarity between concepts based on shared semantic features. Previous versions of SenticNet exploited this principle to cluster natural language concepts sharing similar affective properties (Cambria et al. 2015). Finding groups of similar concepts, however, does not ensure full coverage of all possible semantic inflections of multiword expressions.

In this work, we leverage sub-symbolic AI to automatically discover the conceptual primitives that can better generalize SenticNet's commonsense knowledge. This generalization is inspired by different theories on conceptual primitives (Schank 1972; Jackendoff 1976; Wierzbicka 1996), but also theoretical studies on knowledge representation (Minsky 1975; Rumelhart and Ortony 1977). All such theories claim that a decompositional method is necessary to explore conceptualization.

In the same manner a physical scientist understands matter by breaking it down into progressively smaller parts, a scientific study of conceptualization proceeds by decomposing meaning into smaller parts. Clearly, this decomposition cannot go on forever: at some point we must find semantic atoms that cannot be further decomposed. This is the level of conceptual structure; mental representation that encodes basic understanding and commonsense by means of primitive conceptual elements out of which meanings are built.

In SenticNet, this 'decomposition' translates into the generalization of multiword expressions that convey a specific set of emotions and, hence, carry a particular polarity. The motivation behind this process of generalization is that there are countless ways to express the same concept in natural language and having a comprehensive list of all the possible concept inflections is almost impossible. While lexical inflections such as conjugation and declension can be solved with lemmatization, semantic inflections such as the use of synonyms or semantically-related concepts need to be tackled by conceptual dependency and analogical reasoning.

One of the main reasons why conceptual dependency theory, and many other symbolic methods, were abandoned in favor of sub-symbolic techniques was the amount of time and effort required to come up with a comprehensive-enough set of rules. Sub-symbolic techniques do not require much time nor effort to perform classification but they are data-dependent and function in a black-box manner (i.e., we do not really know how and why classification labels are produced). In this work, leverage the generalization power of recurrent neural networks to automatically discover conceptual primitives for sentiment analysis.

This is not only useful for extending the coverage of SenticNet (100,000 concepts) but also interesting because it represents one of the first efforts to merge symbolic and sub-symbolic AI in the context of sentiment analysis. In particular, we exploit an ensemble of bottom-up data-driven inference (word embeddings and recurrent neural networks) and top-down knowledge representation (conceptual primitives and semantic networks) for polarity detection from text.

Discovering Primitives

A sentence S can be represented as a sequence of words, i.e., $S = [w_1, w_2, ...w_n]$ where n is the number of words in the sentence. The sentence can be split into sections such that the prefix: $[w_1, ...w_{i-1}]$ form the left context sentence with l words and the suffix: $[w_{i+1}, ...w_n]$ form the right context sentence with r words. Here, $c = w_i$ is the target word.

In the first step, we represent these words in a low-dimensional distributed representation known as word embeddings. Specifically, we use the pre-trained 300-dimensional word2vec embeddings provided by (Mikolov et al. 2013) trained on the 3-billion-word Google News corpus. Our context sentences and target concept can now be represented as a sequence of word vectors, thus constituting matrices, $L \in \mathcal{R}^{d_w \times l}$, $R \in \mathcal{R}^{d_w \times r}$ and $C \in \mathcal{R}^{d_w \times 1}$ ($d_w = 300$) for left context, right context and target word, respectively.

biLSTM

To extract the contextual features from these subsentences, we use the biLSTM model on $\mathcal{L}$ and $\mathcal{C}$ independently. Given that we represent the word vector for the t^{th} word in a sentence as x_t, the LSTM transformation can be performed as:

$$X = \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} \quad (1)$$

$$f_t = \sigma(W_f.X + b_f) \quad (2)$$

$$i_t = \sigma(W_i.X + b_i) \quad (3)$$

$$o_t = \sigma(W_o.X + b_o) \quad (4)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_c.X + b_c) \quad (5)$$

$$h_t = o_t \odot tanh(c_t) \quad (6)$$

where d is the dimension of the hidden representations and $W_i, W_f, W_o, W_c \in \mathbb{R}^{d \times (d+d_w)}$, $b_i, b_f, b_o \in \mathbb{R}^d$ are parameters to be learnt during the training (Table 1). σ is the sigmoid function and $\odot$ is element-wise multiplication. The optimal values of the d and k were set to 300 and 100, respectively (based on experiment results on the validation dataset). We used 10 negative samples.

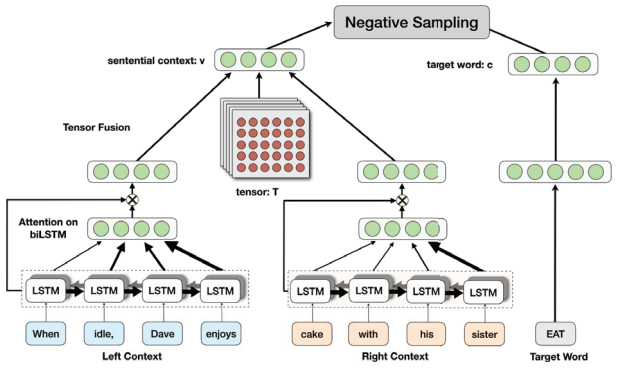

Figure 1: Overall framework for context and word embedding generation.

When a biLSTM is employed, these operations are applied in both directions of the sequence and the outputs for each timestep are merged to form the overall representation for that word. Thus, for each sentence matrix, after applying biLSTM, we get the recurrent representation feature matrix as $H_{LC} \in \mathcal{R}^{2d \times l}$, and $H_{RC} \in \mathcal{R}^{2d \times r}$.

Target Word Representation

The final feature vector **c** for target word c is generated by passing C through a multilayer neural network. The equations are as follows:

$$C^* = tanh(W_a.C + b_a) \quad (7)$$

$$\mathbf{c} = tanh(W_b.C^* + b_b) \quad (8)$$

where $W_a \in \mathcal{R}^{d \times d_w}, W_b \in \mathcal{R}^{k \times d}, b_a \in \mathcal{R}^d$ and $b_b \in \mathcal{R}^k$ are parameters (Table 1) and $\mathbf{c} \in \mathcal{R}^k$ is the final target word vector.

Sentential Context Representation

For our model to be able to attend to subphrases which are important in providing context, we incorporate an attention module on top of our biLSTM for our context sentences. The attention module consists of an augmentary neural network having a hidden layer followed by a softmax output (Fig. 1). It generates a vector which provides weights corresponding to the relevance of the underlying context across the sentence. Below, we describe the attention formulation applied on the left context sentence.

H_{LC} can be represented as a sequence of $[h_t]$ where $t \in [1, l]$. Let A denote the attention network for this sentence. The attention mechanism of A produces an attention weight vector α and a weighted hidden representation r as follows:

$$P = tanh(W_h.H_{LC}) \quad (9)$$

$$\alpha = softmax(w^T.P) \quad (10)$$

$$r = H_{LC}.\alpha^T \quad (11)$$

where, $P \in \mathbb{R}^{d \times l}, \alpha \in \mathbb{R}^l, r \in \mathbb{R}^{2d}$. And, $W_h \in \mathbb{R}^{d \times 2d}, w \in \mathbb{R}^d$ are projection parameters (Table 1).

Finally, the sentence representation is generated as:

$$r^* = tanh(W_p.r) \tag{12}$$

Here, $r^* \in \mathbb{R}^{2d}$ and $W_p \in \mathbb{R}^{d \times 2d}$ is the weight to be learnt while training. This generates the overall sentential context representation for the left context sentence: $E_{LC} = r^*$. Similarly, attention is also applied to the right context sentence to get E_{RC}.

To get a comprehensive feature representation of the context for a particular concept, we fuse the two sentential context representations, E_{LC} and E_{RC}, using a neural tensor network (Socher et al. 2013a). It involves a neural tensor $T \in \mathcal{R}^{2d \times 2d \times k}$ which performs a bilinear fusion across k dimensions. Along with a single layer neural model, the overall fusion can be shown as:

$$\mathbf{v} = tanh(E_{LC}^T.T^{[1:k]}.E_{RC} + W.\begin{bmatrix} E_{LC} \\ E_{RC} \end{bmatrix} + b) \tag{13}$$

Here, the tensor product $E_{LC}^T.T^{[1:k]}.E_{RC}$ is calculated to get a vector $\mathbf{v}^* \in \mathcal{R}^k$ such that each entry in the vector $\mathbf{v}^*$ is calculated as $\mathbf{v}_i^* = E_{LC}^T.T^{[i]}.E_{RC}$, where $T^{[i]}$ is the i^{th} slice of the tensor T. $W \in \mathcal{R}^{k \times 4d}$ and $b \in \mathcal{R}^k$ are the parameters (Table 1). The tensor fusion network thus finally provides the sentential context representation **v**.

Negative Sampling

To learn the appropriate representation of sentential context and target word, we use word2vec's negative sampling objective function. Here, a positive pair is described as a valid context and word pair and the negative pairs are created by sampling random words from a unigram distribution. Formally, the objective function can be defined as:

$$Obj = \sum_{\mathbf{c},\mathbf{v}} (log(\sigma(\mathbf{c}.\mathbf{v})) + \sum_{i=1}^{z} log(\sigma(-\mathbf{c}_i.\mathbf{v}))) \tag{14}$$

Here, the overall objective is calculated across all the valid word and context pairs. We choose z invalid word-context pairs where each $-\mathbf{c}_i$ refers to an invalid word with respect to a context.

Similarity Index

Once the joint embeddings of the target words and their respective sentential contexts are generated, substitution can be effectively performed. Our approach is based on the assumption that a relevant lexical substitute should be both semantically similar to the target word and have similar contextual background.

Thus, given a target word **c** and its sentential context **v**, we calculate the cosine distance of all the other words in the embedding hyperspace with both **c** and **v**. If **b** is a candidate word, the distance is then calculated as:

$$dist(\mathbf{b}, (\mathbf{c}, \mathbf{v}) = cos(\mathbf{b}, \mathbf{c}) + cos(\mathbf{b}, \mathbf{v}) \tag{15}$$

A stricter rule to ensure high similarity between the target and candidate word is to apply multiplication instead of addition:

$$dist(\mathbf{b}, (\mathbf{c}, \mathbf{v}) = cos(\mathbf{b}, \mathbf{c}).cos(\mathbf{b}, \mathbf{v}) \tag{16}$$

Parameters			
Weights			
W_i, W_f, W_o, W_c	$\in \mathbb{R}^{d \times (d+d_w)}$	W_p	$\in \mathbb{R}^{d \times 2d}$
W_b	$\in \mathcal{R}^{k \times d}$	Bias	
W_a	$\in \mathcal{R}^{d \times d_w}$	b_i, b_f, b_o	$\in \mathbb{R}^d$
T	$\in \mathcal{R}^{2d \times 2d \times k}$	b_a	$\in \mathcal{R}^d$
W_h	$\in \mathbb{R}^{d \times 2d}$	b	$\in \mathcal{R}^k$
W	$\in \mathcal{R}^{k \times 4d}$	b_b	$\in \mathcal{R}^k$
w	$\in \mathbb{R}^d$		
Hyperparameters			
d	dimension of LSTM hidden unit		
k	NTN tensor dimension		
z	negative sampling invalid pairs		

Table 1: Summary of notations used in Algorithm 1. Note: d_w is the word embedding size. All the hyperparameters were set using Random Search (Bergstra and Bengio 2012).

We rank the candidates as per their cosine distance and generate the list of possible lexical substitutes. First, we extract all the concepts of the form verb-noun and adjective-noun present in ConceptNet 5. An example sentence for each of these concepts is also extracted. Then, we take one word from the concept (either a verb/adjective or a noun) to be the target word and the remainder sentence acts as the context. The goal now is to find a substitute for the target word having same parts of speech in the given context.

To achieve this, we obtain the context and target word embeddings (**v** and **c**) from the joint hyperspace of the network. For all possible substitute words **b**, we then calculate the cosine similarity using equation (16) and rank them using this metric for possible substitutes. This substitution leads to new verb-noun or adjective-noun pairs which bear the same conceptual meaning in the given context.

Linking Primitives to Concepts and Entities

The deep learning framework described in the previous section allows for the automatic discovery of concept clusters that are semantically related and share a similar lexical function. The label of each of such cluster is a conceptual primitive. The deep framework will not name the clusters automatically so the definition of a conceptual primitive will have to be performed either manually or by automatically selecting the most typical of the terms.

In the verb cluster {`munch, eat, dine, devour, slurp, ingest, ...`}, for example, the term with the highest occurrence frequency in text (the one people most commonly use in conversation) is the second (`eat`). Hence, the cluster will be named as `EAT` and this label will serve as the conceptual primitive to identify the cluster. Thanks to the similarity index, each concept will be assigned to only one cluster (based on highest dot product). These newly discovered clusters will translate into special links to traverse the knowledge graph from Concept Level to Primitive Level.

SenticNet 5 is a three-level semantic network: the Primitive Level is where basic states and actions (and the interactions between them) are defined by means of primitives; the Concept Level is where commonsense concepts are inter-

Algorithm 1 Context and target word embedding generation

procedure TRAINEMBEDDINGS
  Given sentence $S = [w_1, w_2, ...w_n]$ s.t. w_i is target word.
  $\mathcal{L} \leftarrow E([w_1, w_2, ...w_{i-1}])$ ▷ E() : word2vec embedding
  $\mathcal{R} \leftarrow E([w_{i+1}, w_2, ...w_n])$
  $\mathcal{C} \leftarrow E(w_i)$
  c ←TargetWordEmbedding($\mathcal{C}$)
  v ←ContextEmbedding($\mathcal{L}, \mathcal{R}$)
  NegativeSampling(**c**, **v**)

procedure TARGETWORDEMBEDDING($\mathcal{C}$)
  $C^* = tanh(W_a.C + b_a)$
  $\mathbf{c} = tanh(W_b.C^* + b_b)$
  return c

procedure CONTEXTEMBEDDING($\mathcal{L}$, $\mathcal{R}$)
  $H_{LC} \leftarrow \phi$
  $h_{t-1} \leftarrow 0$
  for t : [1, $i-1$] **do**
    $h_t \leftarrow LSTM(h_{t-1}, L_t)$
    $H_{LC} \leftarrow H_{LC} \cup h_t$
    $h_{t-1} \leftarrow h_t$
  $H_{RC} \leftarrow \phi$
  $h_{t-1} \leftarrow 0$
  for t : [$i+1$, n] **do**
    $h_t \leftarrow LSTM(h_{t-1}, R_t)$
    $H_{RC} \leftarrow H_{RC} \cup h_t$
    $h_{t-1} \leftarrow h_t$
  E_{LC} ←Attention(H_{LC})
  E_{RC} ←Attention(H_{RC})
  v ←NTN(E_{LC}, E_{RC})
  return v

procedure LSTM(h_{t-1},x_t)
  $X = \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}$
  $f_t = \sigma(W_f.X + b_f)$
  $i_t = \sigma(W_i.X + b_i)$
  $o_t = \sigma(W_o.X + b_o)$
  $c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_c.X + b_c)$
  $h_t = o_t \odot tanh(c_t)$
  return h_t

procedure ATTENTION(H)
  $P = tanh(W_h.H)$
  $\alpha = softmax(w^T.P)$
  $r = H.\alpha^T$
  return r

procedure NTN(E_{LC}, E_{RC})
  $\mathbf{v} = tanh(E_{LC}^T.T^{[1:k]}.E_{RC} + W.\begin{bmatrix} E_{LC} \\ E_{RC} \end{bmatrix} + b)$
  return v

connected through semantic relationships; finally, the Entity Level is a layer of named entities that are linked to commonsense concepts by IsA relationships.

While developing a knowledge representation of this kind for general commonsense reasoning would be a formidable task, it is feasible in the context of sentiment analysis because we only aim to encode subjective commonsense concepts, i.e., concepts that convey either positive or negative polarity.

For SenticNet 5, in particular, we focus on polarity-bearing states, e.g., `INTACT`, and result verbs that modify such states, e.g., `BREAK` and `FIX` (Fig. 2). The clusters discovered by means of deep learning are exploited to link such primitives to their lexical substitutes (commonsense knowledge) in the Concept Level. In turn, commonsense concepts are linked to named entities (common knowledge[1]) in the Entity Level. While this does not solve the symbol grounding problem (as we only define `INTACT` by the emotions associated with it), it helps to consistently reduce it, as several adjectives and verbs are defined in function of only one item (the `INTACT` primitive).

The power of SenticNet 5 resides exactly in this. We do not need to infer polarity based on (direct or indirect) emotion links in the semantic network of commonsense knowledge anymore: affective reasoning is performed at primitive level. Once we define `INTACT` as positive, we automatically defined as positive all its lexical substitutes (direct links from the Concept Level but also indirect links from the Entity Level), plus the polarity of the result verbs related to it (`BREAK` as negative because it changes the state into its opposite, and `FIX` as positive as it restores the original state) and their lexical substitutes, plus all the lexical substitutes of its opposite (`!INTACT`, e.g., `broken`, `crumbled`, or `shredded`) as negative. Besides allowing for generalization of concepts, conceptual primitives are extremely powerful for inferring the polarity of multiword expressions dynamically, as per algebraic multiplication (where negative multiplied by positive, or vice versa, is negative). For example, when coupling the positive result verb `INCREASE` with a positive noun, the resulting polarity is positive (e.g., `INCREASE_PLEASURE`). If the noun is negative, instead, the resulting polarity is negative (e.g., `INCREASE_LOSS`).

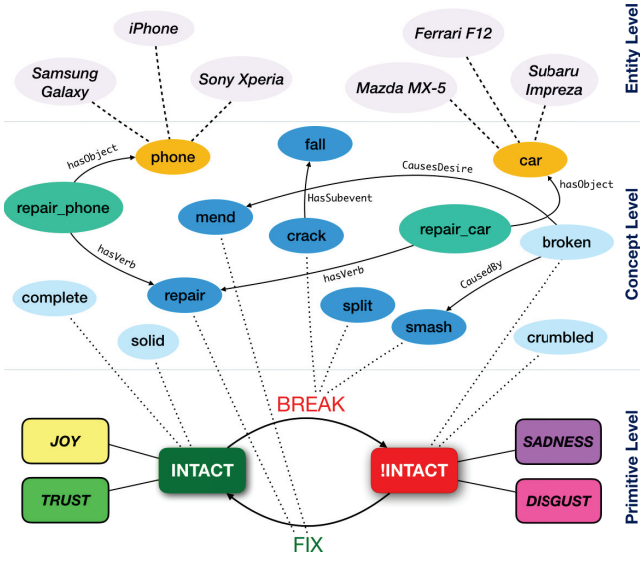

Figure 2: A sketch of SenticNet 5's graph showing part of the semantic network for the primitive `INTACT`.

[1] from IsaCore (Cambria et al. 2014)

Similarly, when coupling the negative result verb DECREASE with a positive noun, the resulting polarity is negative (e.g., DECREASE_GAIN). If the noun is negative, instead, the resulting polarity is positive (e.g., DECREASE_PAIN).

AffectiveSpace

In order to automatically infer the polarity of key states, e.g., INTACT, and to perform a consistency check on the clusters generated by means of deep learning, we use AffectiveSpace (Cambria et al. 2015), a vector space of affective commonsense knowledge built by means of random projections (Bingham and Mannila 2001). Random projections are a powerful dimensionality reduction technique based on the Johnson and Lindenstrauss's Lemma (Balduzzi 2013), which states that, with high probability, for all pairs of points $x, y \in X$ simultaneously:

$$\sqrt{\frac{m}{d}} \parallel x - y \parallel_2 (1-\varepsilon) \leq \parallel \Phi x - \Phi y \parallel_2 \leq \tag{17}$$

$$\leq \sqrt{\frac{m}{d}} \parallel x - y \parallel_2 (1+\varepsilon), \tag{18}$$

where X is a set of vectors in Euclidean space, d is the original dimension of this Euclidean space, m is the dimension of the space we wish to reduce the data points to, ε is a tolerance parameter measuring the maximum allowed distortion extent rate of the metric space, and Φ is a random matrix. Structured random projections for making matrix multiplication much faster was introduced in (Sarlos 2006). When the number of features is much larger than the number of training samples ($d \gg n$), subsampled randomized Hadamard transform (SRHT) is preferred. For $d = 2^p$ (where p is any positive integer), a SRHT can be defined as:

$$\Phi = \sqrt{\frac{d}{m}} \mathrm{RHD} \tag{19}$$

where • m is the number we want to subsample from d features randomly;

• R is a random $m \times d$ matrix. The rows of R are m uniform samples (without replacement) from the standard basis of $\mathbb{R}^d$;

• H$\in \mathbb{R}^{d \times d}$ is a normalized Walsh-Hadamard matrix, which is defined recursively:

$$H_d = \begin{bmatrix} H_{d/2} & H_{d/2} \\ H_{d/2} & -H_{d/2} \end{bmatrix} \text{ with } H_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix};$$

• D is a $d \times d$ diagonal matrix and the diagonal elements are i.i.d. Rademacher random variables.

Our subsequent analysis only relies on the distances and angles between pairs of vectors (i.e., the Euclidean geometry information) and it is sufficient to set the projected space to be logarithmic in the size of the data (Ailon and Chazelle 2010) and, hence, apply SRHT.

By exploiting the information sharing property of random projections, concepts with the same semantic and affective valence are likely to have similar features, i.e., concepts conveying the same meaning and emotions tend to fall near each other in AffectiveSpace.

Similarity does not depend on concepts' absolute position in the vector space, but rather on the angle these make with the origin. Thus, positive concepts such as happy, celebrate, and birthday are found in one area of the vector space, while negative concepts like depressed, cry, and loss are found in a diametrically opposite zone.

We use this information to both infer the polarity of conceptual primitives by majority voting, e.g., assign positive polarity to INTACT based on the fact that most concepts in its cluster are positive, and to ensure polarity consistency by discarding the concepts that have opposite polarity with respect to the majority, e.g., discard the negative lexical substitutes of INTACT (if any).

Experiments

In this section, we evaluate both the performance of the deep learning framework and the performance of SenticNet 5 as a knowledge base for sentiment analysis.

Evaluation of the Deep Learning Framework

Training Corpus In order to train the deep learning based primitive generation framework, we used the 2-billion-word ukWaC (Baroni et al. 2009) as our learning corpus. We removed the sentences which have length greater than 80, which resulted in a 7% reduction of the corpus (and significantly sped up the training). We lower-cased text and removed tokens with occurrence less than 90. Finally, we were left with 173K words in the corpus.

Evaluation Corpus In order to evaluate the performance of the proposed approach, we employed it to solve the problem of lexical substitution. As lexical substitution datasets, we used the LST-07 dataset from the SemEval 2007 lexical substitution task (McCarthy and Navigli 2007) and the 15-thousand target word all-words LST-14 dataset from SemEval 2014 (Kremer et al. 2014). The first one comes with a 300-sentence dev set and a 1710-sentence test set split; the second one comes with a 35% and 65% split, which we used as the dev set and test set, respectively. The performance is measured using generalized average precision (GAP) in which we rank the lexical substitutes of a word based on the cosine similarity score calculated among a substitution and the context embedding. This ranking is compared to the gold standard lexical substitution ranking provided in the dataset. The result of this approach is shown in Table 3.

Baseline Methods

- Baseline 1 – Melamud et al. (Melamud et al. 2015) proposed a model to find lexical substitution of a target based on skipgram word embeddings and incorporating syntactic relations in the skipgram model.
- Baseline 2 – This baseline has been implemented by training the skipgram model on the learning corpus and then simply taking the average of the words present in the context as context representation. The cosine similarity among this context representation and the target word embeddings is calculated to find a matching for the lexical substitution.

Our approach	Baseline 1	Baseline 2
Eat		
Dine	Consume	Glutonny
Consume	Taste	Devour
Gulp	Swallow	Drink
Chew	Bite	Nibble
Ingest	Savor	Relish

Table 2: The top substitution of the target word *eat* in "When idle, Dave enjoys eating cake with his sister".

Results The performance of the proposed approach and comparison with the state of the art and baselines are shown in Table 3.

Models	LST-07	LST-14
Our approach	58.55%	56.25%
Baseline 1	55.1%	53.6%
Baseline 2	52.35%	50.05%

Table 3: Comparison of state of the art and baseline to our proposed approach on the benchmark datasets. Baseline 1 is the current state of the art (Melamud et al. 2015).

Evaluation of SenticNet 5

We tested SenticNet 5 (available both as a standalone XML repository[2] and as an API[3]) against two well-known sentiment resources: the Blitzer Dataset (Blitzer, Dredze, and Pereira 2007) which was later modified by (Poria et al. 2015) and the Movie Review Dataset developed (Pang and Lee 2005). The Blitzer Dataset consists of product reviews in seven different domains. For each domain there are 1,000 positive and 1,000 negative reviews. We obtained this dataset, containing 3,800 positive sentences and 3,410 negative, from the authors of (Poria et al. 2015). Instead, the Movie Review Dataset has been restructured from document to sentence level by Socher et al. (Socher et al. 2013c) and it contains 4,800 positive sentences and 4,813 negative ones.

Performing Polarity Detection with SenticNet

While SenticNet 5 can be used as any other sentiment lexicon, e.g., concept matching or bag-of-concepts model, the right way to use the knowledge base for the task of polarity detection is in conjunction with sentic patterns (Poria et al. 2015), sentiment-specific linguistic patterns that infer polarity by allowing affective information to flow from concept to concept based on the dependency relation between clauses. The sentiment sources of such affective information are extracted from SenticNet 5 by firstly generalizing multiword expressions and words by means of conceptual primitives and, secondly, by extracting their polarity.

[2]http://sentic.net/downloads

[3]http://sentic.net/api

SenticNet 5 vs. SenticNet 4

We compared the performance of SenticNet 5 with its predecessor SenticNet 4 (Cambria et al. 2016) for the task of sentence-level polarity detection, using sentic patterns. The main drawback of SenticNet 4 is while analyzing sentiment using sentic patterns many concepts are not found due to the lack of a similarity measurement method. SenticNet 5, instead, has the capability to infer sentiment polarities of new concepts. Assuming the new concept to be a verb-noun pair with words $\mathbf{w}_1$ and $\mathbf{w}_2$. Our algorithm is able to identify the primitive cluster of word $\mathbf{w}_1$ and calculate its nearest neighbors in this word-context joint hyperspace. The polarity can now be adapted from the closest word belonging to the same cluster. The overall polarity of the multiword concept is then inferred using multiplicative rules. It should be noted that the same rule is also applied in the case of single word concept as required by sentic patterns. When tasked to find the sentiment of a new sentence containing an unknown concept, SenticNet 5 can now infer the polarity using the process mentioned. Our algorithm thus aids in zero-shot concept-based sentiment analysis. This inference scheme is also applied as a bootstrapping procedure to diversify the knowledge base itself.

Framework	Accuracy
Sentic Patterns and SenticNet 4	91.3%
Sentic Patterns and SenticNet 5	94.6%

Table 4: Comparison on the Blitzer Dataset

The results of the classification on the Blitzer and Movie Review Dataset with SenticNet 4 and SenticNet 5 are shown in Table 4 and Table 5. We also compare the result obtained using the proposed approach with SentiWordNet.

Framework	Accuracy
Socher et al., 2012	80.0%
Socher et al., 2013	85.4%
Sentic Patterns and SentiWordNet	84.2%
Sentic Patterns and SenticNet 4	90.1%
Sentic Patterns and SenticNet 5	92.8%

Table 5: Comparison on the Movie Review Dataset

Conclusion

In this work, we used an ensemble of symbolic and subsymbolic AI to automatically discover conceptual primitives for sentiment analysis. This generalization process allowed us to largely extend the coverage of SenticNet and to build a new knowledge representation for better encoding semantics and sentiment. In the future, we plan to extend such a representation to a new level, in attempt to further reduce the symbol grounding problem in the context of sentiment analysis. Since such level should be both language-independent and symbol-independent, we plan to define primitives in mathematical terms, e.g., via arithmetic proportionality. For example, a way to define the intactness of an item i could be INTACT : i = !INTACT : $i/(n+1), n \in \mathbb{Z}^+$.

References

Ailon, N., and Chazelle, B. 2010. Faster dimension reduction. *Communications of the ACM* 53(2):97–104.

Balduzzi, D. 2013. Randomized co-training: from cortical neurons to machine learning and back again. *arXiv preprint arXiv:1310.6536*.

Baroni, M.; Bernardini, S.; Ferraresi, A.; and Zanchetta, E. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation* 43(3):209–226.

Bergstra, J., and Bengio, Y. 2012. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research* 13(1):281–305.

Bingham, E., and Mannila, H. 2001. Random projection in dimensionality reduction: applications to image and text data. In *ACM SIGKDD*, 245–250.

Blitzer, J.; Dredze, M.; and Pereira, F. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, volume 7, 440–447.

Cambria, E.; Song, Y.; Wang, H.; and Howard, N. 2014. Semantic multi-dimensional scaling for open-domain sentiment analysis. *IEEE Intelligent Systems* 29(2):44–51.

Cambria, E.; Fu, J.; Bisio, F.; and Poria, S. 2015. AffectiveSpace 2: Enabling affective intuition for concept-level sentiment analysis. In *AAAI*, 508–514.

Cambria, E.; Poria, S.; Bajpai, R.; and Schuller, B. 2016. SenticNet 4: A semantic resource for sentiment analysis based on conceptual primitives. In *COLING*, 2666–2677.

Cambria, E.; Poria, S.; Gelbukh, A.; and Thelwall, M. 2017. Sentiment analysis is a big suitcase. *IEEE Intelligent Systems* 32(6):74–80.

dos Santos, C. N., and Gatti, M. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, 69–78.

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

Jackendoff, R. 1976. Toward an explanatory semantic representation. *Linguistic Inquiry* 7(1):89–150.

Kremer, G.; Erk, K.; Padó, S.; and Thater, S. 2014. What substitutes tell us-analysis of an" all-words" lexical substitution corpus. In *EACL*, 540–549.

McCarthy, D., and Navigli, R. 2007. Semeval-2007 task 10: English lexical substitution task. In *International Workshop on Semantic Evaluations*, 48–53.

Melamud, O.; Levy, O.; Dagan, I.; and Ramat-Gan, I. 2015. A simple word embedding model for lexical substitution. In *VS@ HLT-NAACL*, 1–7.

Melville, P.; Gryc, W.; and Lawrence, R. D. 2009. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *ACM SIGKDD*, 1275–1284.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Minsky, M. 1975. A framework for representing knowledge. In Winston, P., ed., *The psychology of computer vision*. New York: McGraw-Hill.

Pang, B., and Lee, L. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*, 115–124.

Pang, B.; Lee, L.; and Vaithyanathan, S. 2002. Thumbs up?: Sentiment classification using machine learning techniques. In *EMNLP*, volume 10, 79–86.

Poria, S.; Cambria, E.; Gelbukh, A.; Bisio, F.; and Hussain, A. 2015. Sentiment data flow analysis by means of dynamic linguistic patterns. *IEEE Computational Intelligence Magazine* 10(4):26–36.

Poria, S.; Cambria, E.; Hazarika, D.; and Vij, P. 2016. A deeper look into sarcastic tweets using deep convolutional neural networks. In *COLING*, 1601–1612.

Poria, S.; Cambria, E.; Hazarika, D.; Mazumder, N.; Zadeh, A.; and Morency, L.-P. 2017. Context-dependent sentiment analysis in user-generated videos. In *ACL*, 873–883.

Poria, S.; Cambria, E.; and Gelbukh, A. 2016. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems* 108:42–49.

Rumelhart, D., and Ortony, A. 1977. The representation of knowledge in memory. In *Schooling and the acquisition of knowledge*. Hillsdale, NJ: Erlbaum.

Sarlos, T. 2006. Improved approximation algorithms for large matrices via random projections. In *FOCS*, 143–152.

Schank, R. 1972. Conceptual dependency: A theory of natural language understanding. *Cognitive Psychology* 3:552–631.

Smith, B., ed. 1988. *Foundations of Gestalt Theory*. Munich and Vienna: Philosophia Verlag.

Socher, R.; Chen, D.; Manning, C. D.; and Ng, A. 2013a. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*, 926–934.

Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 1642–1654.

Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013c. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, volume 1631, 1642–1654.

Tang, D.; Wei, F.; Qin, B.; Liu, T.; and Zhou, M. 2014a. Coooolll: A deep learning system for twitter sentiment classification. In *SemEval*, 208–212.

Tang, D.; Wei, F.; Yang, N.; Liu, T.; and Qin, B. 2014b. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL*, 1555–1565.

Wierzbicka, A. 1996. *Semantics: Primes and Universals*. Oxford University Press.

Yu, H., and Hatzivassiloglou, V. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *EMNLP*, 129–136.