Mining Pros and Cons of Actions from Social Media for Decision Support

Ebad Ahmadzadeh
Dept. of Computer Sciences
Florida Institute of Technology
Email: mahmadzadehe2012@my.fit.edu

Philip K. Chan
Dept. of Computer Sciences
Florida Institute of Technology
Email: pkc@cs.fit.edu

***Abstract*—The task of mining pros and cons from actions has applications in decision support. Given an action query and social media data, we mine related pros and cons of the action via extracting significant events as potential outcomes of the action. We propose using actions and characteristics to select relevant messages, and adjective vectors to establish similarity among adjectives. We introduce SS to select event headlines, and to rank them in the final pros-and-cons table. Our results on two data sets indicate our algorithm can generate more meaningful pros and cons than an existing algorithm.**

I. Introduction

Discovering pros and cons of actions has many potential applications in decision support, like purchase recommendation, and finding likely side effects of medications. Establishing a knowledge base on actions and outcomes could assist individuals in making decisions by illustrating potential outcomes given an action that they intend to perform. Those outcomes then can be categorized to form pros and cons of the action.

In this paper, we propose algorithms to create such a knowledge base on actions and outcomes from social media. Inspired by Kiciman and Richardson [1] we introduce techniques to improve their core components to attain more meaningful pros and cons. Given an action and social media data, our goal is to effectively mine pros and cons of performing the action. Our contributions include:

1) identifying relevant messages containing observations or opinions about the entity of the query by extracting actions and characteristics, as opposed to filtering irrelevant messages in a semi-manual fashion [1],
2) introducing Adjective Vectors to measure semantic similarity between adjectives to improve the clustering quality as in [1],
3) proposing Significance Score (SS) to quantify significance of messages in terms of representing meaningful outcomes, in addition to [1]'s relative likelihood score as a measure to rank distinguishing events, and
4) based on two data sets collected from social media, showing that our algorithm mines more meaningful pros and cons of the given action compared to [1].

We discuss the related work in Sec. II. Sec. III provides the problem statement and describes the different steps of our algorithm. We evaluate our algorithms in Sec. IV and conclude in Sec. V.

II. Related Work

Much research exist based on the assumption that co-occurrence may establish some true relationships between actions and outcomes. For instance, in the health domain, social media studies have found relationships among diseases, medicines, related symptoms and side-effects [2].

Richardson [3] uses search queries to identify relationships between drugs and their adverse side-effects (consequences). Similar studies address the problem of learning about the real world events from social media. They predict the future signals from social media given a known signal. These techniques are applied to different domains like economics [4]. Olteanu et al. [5] performed an open-domain study on words expressed by social media users after experiencing distinct situations, and found that causal relationships between those words and the situations are in average 55-100% more likely than semantic relationships.

Kiciman and Richardson [1] investigate the feasibility of mining the relationship between actions and their consequences based on social media. The inputs include a large corpus of personal status messages from social media and an action query. The output is a list of pros and cons of doing the action. A timeline of events is constructed for each user, where each event is a collection of relevant personal-experience posts. The user timelines are center-aligned at the point of performing the action. They are then divided into two groups of positive and negative user timelines representing those who did the action and others who did the reverse action respectively. Finally, most important events are found based on relative likelihood, and they are split into pros and cons via aggregate affect valence.

Similarity between words can also be measured by vector representation of words. Training of such vectors has been done via different techniques like the ones based on matrix factorization [6] and window-based methods [7]. Pennington et al. [8] propose GloVe, an unsupervised method that benefits from both families and outperforms them on word similarity, word analogy, and name entity recognition tasks.

One of the main shortcomings of [1] is in the event extraction step where sentences are broken into phrases and then

Algorithm 1 ProCon($Corpus$, $ActionQuery$)

```
1: find users who performed ActionQuery, and collect a timeline of messages for each user from Corpus (Sec. III-A)
2: select messages that express **actions** or **characteristics** (Contribution 1) related to the action query (Sec. III-B)
3: extract events from messages with techniques including **Adjective Vectors** (Contribution 2; Sec. III-C)
4: rank the events via **Significance Score** (**SS**) (Contribution 3; Sec. III-D)
```

clustered into events. Events consist of short phrases that could be less meaningful sometimes. For example, "damn kitten" or "cat is literally" are phrases from their output table that could not express an outcome without referring to the message they belong to. Therefore, selecting messages that represent the event-phrase seems to be important. However, how to pick the example messages in the pros-and-cons is not clear. Furthermore, they performed semantic correlational analysis to order the events with respect to relative likelihood of the event occurring after doing the action compared to both before doing the action and after doing the reverse action. Although the relative likelihood score captures distinguishing events, the results potentially contain events that are not important consequences. For example, "cat being named" is in the results, but it doesn't seem to be the most significant outcome of adopting a cat.

III. Problem Statement and Algorithm

Given an action, the goal is to discover likely outcomes that could be useful in decision making. The inputs include a large corpus of social media messages, and a query about performing an action. The output is the most likely outcomes of the query action in form of a pros-and-cons table. While we maintain the main skeleton of [1], we propose improvements to its core components. The overall algorithm is shown in Alg. 1, and we discuss the steps next.

A. Identifying Relevant Users

From a corpus of social media messages, we find a large number of users who expressed their experience related to the action query. For instance, we search for "adopted a cat|kitty|kitten" when the action query is "adopting a cat". After identifying users who wrote those messages, we collect the entire timeline of messages for each user.

B. Selecting Relevant Messages

The goal of this step is to select messages that describe a relevant situation that the user has experienced. First we use an n-gram approach to filter out non-experiential messages. Then, we select messages with actions and characteristics.

Table I: Rules for identification of actions or characteristics

Type	Rule
characteristic	query-entity e_q is subject of verb v & adjective a is adjectival complement of v
characteristic	adjective a is adjectival modifier of query-entity e_q
action	query-entity e_q is subject of verb v
action	query-entity e_q is object of verb v

1) Experiential Messages: Relationship between actions and consequences are more meaningful when they are extracted from personal experiences. But social media messages also include other types, like news and advertisements. We use a simplified unsupervised technique based on n-gram models [1] to filter out undesired messages containing certain keywords and phrases. First, we hand-label a small set of experiential and non-experiential messages (100 messages each). Then, we find a set of n-grams whose likelihood of occurrence in the experiential set is much lower than the other set (we use $n \in \{1, 2, 3\}$ in our experiments). For instance, "We have a kitten ready for adoption" is considered a non-experiential message as it contains "ready for adoption", a trigram with much lower likelihood of occurrence in the experiential set than the other set.

2) Messages with Actions or Characteristics: Actions and characteristics are usually used in natural languages to express effects and outcomes of an action. Hence, we find messages containing actions or characteristics that refer to the *query entity*. Here we define *query entity* as the main object of the query. For example, given query "adopting a cat", the query entity would be "cat". *Actions* and *characteristics* are represented by verbs and adjectives respectively. An *action* is any verb done by or to the query entity, and similarly, a *characteristic* is any adjective mentioned about the query entity. For instance, given action query "Adopting a cat", and sample message "I love coming home and going to bed because my cute cat cuddles with me.", "cuddle" is an action done by the query entity "cat". Also "cute" is a characteristic about the query entity.

To find messages with such grammatical structures, we extract dependency relationships and part of speech tags from each sentence using Stanford Dependency Parser [9]. Then, we find messages with actions or characteristics via a set of handmade grammar rules listed in Table I. The first characteristic rule selects any message where the query entity is subject of a verb having an adjectival phrase. For instance, in "My fat cat is asleep.", "asleep" is the adjectival complement to "is" where "cat" is the subject of "is". The second characteristic rule selects any message with an adjectival modifier for the query entity. For instance, in the previous example, "fat" is an adjectival modifier of the query entity "cat". The first and second action rules select any message where the query entity is a subject or object of a verb. Sec. IV-G shows some examples of messages that are eliminated by our method.

C. Extracting Significant Events

The main goal of this step is to summarize the actions (verbs) and characteristics (adjectives) into events such that each event represents a collection of verbs or adjectives about the same action or characteristic. That is, the verbs and adjectives are clustered (separately) to form events. For example, {"nice", "cute", "lovely"} could form a cluster of adjectives, and {"plays", "runs", "jumps"} could form a cluster of verbs. Event extraction using phrases, as done in previous work [1], provides more effective results than using bag of words as it handles canonicalization. However, it falls short in establishing semantic relationships among words. Since it essentially works based on matching tokens, the clusters are small, with high precision but low recall. As a result, an event can be broken into many small events that otherwise might have formed a significant event. We employ different word representations to establish stronger semantic relationships between verbs and between adjectives. We expect the representations help create clusters with higher recall. Our event extraction algorithm first creates clusters of verbs and adjectives. Then, it identifies a best candidate message and event to represent each cluster.

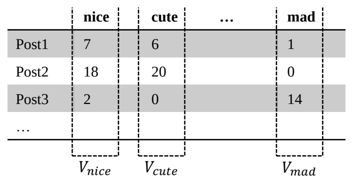
1) Clustering of Verbs/Adjectives:

Representation of Verbs: We use *Wordnet* [10] hierarchy of verbs. The hierarchy represents different relation types like *is-a, has-a* and it becomes more specific toward leaves. Thus, a data point in this case is a verb token.

Representation of Adjectives: Wordnet does not provide a hierarchy for adjectives, and the task of calculating similarity between adjectives remains difficult in the domain. We performed experiments with LESK [11] and Extended LESK [12] algorithms, they perform poorly for clustering. This is mainly because Wordnet provides only limited definitions and relations for adjectives. hence, we represent adjectives via a different approach.

We propose a different approach based on usage of language by human on the Web where the key assumption is that the query entity likely has a rather unique combination of uncontroversial characteristics. In specific, usually only a small number of characteristics is significantly noticeable about a particular instance of the query entity. Although there are many terms (adjectives) to express one characteristic, these terms are likely to be reused to express other instances with similar characteristic. So, distribution of similar terms about a characteristic of an entity should be similar.

For instance, each cat has a small number of noticeable characteristics. Characteristics such as "cute", "sweet", "nice" and many other terms might be used to describe a cat. Although these words are different, they share the same characteristic of the cat. So, they are likely to co-occur frequently in comments for the same cat. However, it is unlikely to see terms with opposite meaning (e.g. "ugly") to describe the same cat. Therefore, co-occurrence of similar terms (e.g. "cute" and "nice") are likely to be high on "cute" cats. We use this idea to represent adjectives.



	nice	cute	...	mad
Post1	7	6		1
Post2	18	20		0
Post3	2	0		14
...				
	V_{nice}	V_{cute}		V_{mad}

Figure 1: Adjective vectors: each number shows the occurrence frequency of an adjective in comments to a social media post.

We employ social media posts and comments about the query entity to establish semantic relationships between adjectives (in the experiments we use Reddit (www.reddit.com) data for this purpose). Each post is about an instance of the query entity (e.g. a cat), and the comments discuss the same entity from different perspectives, and mostly express similar characteristics with different words. We represent each post, along with its comments, with one vector. Each vector contains frequency distribution of the adjectives mentioned about the entity in the comments to the post. In other words, each adjective represents a dimension of a post. For example, each row in Fig. 1 is a vector representing a post. Then, each adjective is represented by an *adjective vector* containing the frequency of the adjective in all comments to the original posts. Each column in Fig 1 illustrates an adjective vector. For example, V_{nice} is the adjective vector for "nice", and V_{cute} is the adjective vector for "cute". Since "nice" and "cute" tend to co-occur more frequently than "nice" and "mad" or "cute" and "mad" for the same cat, $V_{nice} - V_{cute} << V_{nice} - V_{mad}$ is likely to hold. That is, the adjective vectors of similar adjectives are expected to be in close proximity.

In addition to our proposed adjective vectors, we also use GloVe [8], an unsupervised method for creating word embedding trained on different corpora to represent words.

Distance Function: After representing verbs and adjectives we need a distance function to calculate the distance between a pair of verbs or adjectives. For verbs, we use *hso* [13] which is based on the path length between two nodes of the verb hierarchy. For adjectives, we use cosine measure to calculate distance between adjective vectors or GloVe word vectors.

Clustering Algorithm: We use *Kmeans++* [14] to cluster verbs and adjectives separately. As a result we generate a number of action clusters from the action verbs and a number of characteristic clusters from the characteristic adjectives.

2) Identifying Representative Messages and Events:

The next step is to identify a single significant message that represents each event cluster. First, each cluster member, whether a verb or an adjective, is associated with one or more messages. Next, we use a scoring mechanism that we refer to as *Significance Score* (SS) to pick an exemplar

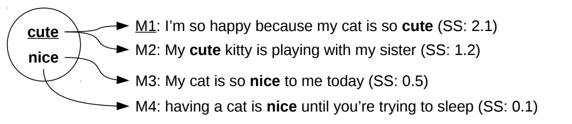

cute
nice
M1: I'm so happy because my cat is so cute (SS: 2.1)
M2: My cute kitty is playing with my sister (SS: 1.2)
M3: My cat is so nice to me today (SS: 0.5)
M4: having a cat is nice until you're trying to sleep (SS: 0.1)

Figure 2: Identifying Representative Messages: M_1 is first selected due to its highest Significance Score compared to the other messages. Then, "cute" is identified as the event associated with M_1, finally "cute" and M_1 are selected as the exemplar to represent the cluster.

message for the cluster. Then, that message along with the associated verb or adjective represent the event. Fig 2 illustrates an example cluster with two adjectives ("cute" and "nice"). Each adjective is associated with all messages that contain the adjective. For example, M_1 and $M2$ contain "cute", and M_3 and M_4 contain "nice". Next, we pick the message with the highest significance. The message (e.g. M_1) along with the associated adjective (e.g. "cute") are then selected as the exemplar to represent the cluster. In order to score messages we employ five factors that contribute to the significance of an event and are sentiment, reasoning, comparison, coverage, and length. We explain each factor in more details.

Sentiment Factor: We use VADER [15], a rule-based sentiment model to calculate sentiment factor as an aggregated score normalized between 0 to 1 to show negative to positive respectively.

Reasoning Factor: Messages with reasoning are highly desired because they provide reasons that probably support their feeling about the outcome. It is calculated as a binary variable that is 1 when any phrase indicating reasoning is observed in the message (e.g. because, therefore, as a result, is why), and it is 0 otherwise.

Comparison Factor: Comparison is often used in decision making process. Comparison factor is also calculated as a binary variable that is 1 when comparison tokens are observed and 0 otherwise. The tokens include both keywords and part-of-speech tags. We use POS tags that are used for comparison words (*JJR, RBR, JJS, RBS*) [16] to identify comparison in sentences. For example, in "cat makes a better pet", the POS tag for "better" is JJR. An advantage of using POS tags is that many words can be represented by one tag. However, POS tagging can be prone to error on incomplete or conversational sentences that usually contain typos. Therefore, we use a small set of keywords as well (*more, most, less, enough*).

Coverage Factor: A message is a stronger member of its cluster when it contains more than one cluster words. We define coverage factor as the percentage of cluster words observed in a message normalized between 0 and 1. For example, "my fat cat is asleep" has 0.67 coverage if the cluster contains three adjectives "fat", "asleep" and "tired".

Length Factor: Length of a message in terms of number of words is another potential indication for a message to

Algorithm 2 ExtractSignificantEvents($adjectives, verbs, messages, redditData, k_{vb}, k_{adj}$)

```
1: $distMatrix_{vb} \leftarrow calcDistMatrix(WordnetHierarchy_{vb}, verbs)$
2: $events_{vb} \leftarrow cluster(distMatrix_{vb}, k_{vb})$    ▷ KMeans++
3: for $event_i \in events_{vb}$ do
4:     for $verb_j \in event_i$ do
5:         $msgList_{vb} \leftarrow getMessages(verb_j, messages)$
6:         $m_{rep} \leftarrow msgMaxSS(msgList_{vb})$
7:         $events_{vb}[i][j] \leftarrow (verb_j, m_{rep})$
8: $adjVectors_{adj} \leftarrow createAdjVectors(adjectives, redditData)$
9: $distMatrix_{adj} \leftarrow calcDistMatrix(adjVectors_{adj}, adjectives)$
10: $events_{adj} \leftarrow cluster(distMatrix_{adj}, k_{adj})$    ▷ KMeans++
11: for $event_i \in events_{adj}$ do
12:     for $adj_j \in event_i$ do
13:         $msgList_{adj} \leftarrow getMessages(adj_j, messages)$
14:         $m_{rep} \leftarrow msgMaxSS(msgList_{adj})$
15:         $events_{adj}[i][j] \leftarrow (adj_j, m_{rep})$
16: return $events_{vb}, events_{adj}$
```

be informative. We exclude tokens like urls, hashtags, and mentions. This factor is normalized between 0 and 1 by comparing all messages within a cluster.

We combine the factors via a weighted sum approach:

$$SS = w_{snt}s_{snt} + w_{res}s_{res} + w_{cmp}s_{cmp} + w_{cov}s_{cov} + w_{len}s_{len} \tag{1}$$

where *SS* is the Significance Score, w_i is the weight used for ith factor, and s_i is the ith factor. snt, res, cmp, cov, len represent sentiment, reasoning, comparison, coverage, and length factors respectively. Finally, the message with the highest SS and its associated verb or adjective are selected as the event to represent the cluster to which they belong.

Alg. 2 summarizes the steps of extracting significant events. The inputs are the action verbs and characteristic adjectives along with messages to which they belong, the Reddit data discussed in Sec. III-C1, and cluster sizes for verbs events and adjective events. First, the distance matrix for verbs is created using Wordnet hierarchy of verbs (line 1), and then the verbs are clustered to form events (line 2). Next, for each verb in the clusters we get all messages from which the verb was extracted as an action verb. Then, we use SS to to select the best message to represent that verb (line 3-7). Subsequently, we create adjective events. First, we create an Adjective Vector for each adjective using Reddit data, as discussed in Sec. III-C1 (line 8). Then we follow similar steps as we did for the verbs, namely, calculating distance matrix (line 9), clustering (line 10), and selecting a representative for each adjective in the clusters (line 11-15). Finally, the verb events and adjective events are returned. Table. Vb in Sec. IV-H exhibits some example clusters with their associated events and representative messages created by Alg. 2.

D. Ranking and Categorizing Events

After extracting significant events and finding a representative message for each event, we rank and categorize them into a pros-and-cons table. We follow three steps to generate

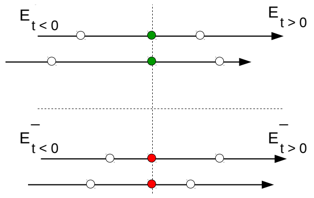


Figure 3: Quadrants of user timelines

the table: First, a collection of highly distinguishing events are selected via correlation analysis used in the previous work. Second, the messages from the first step are ranked by our SS. Third, the ranked messages are categorized into pros or cons when their sentiment scores are large enough. Next, we explain each of the three steps in more details.

1) Distinguishing Events via Correlation: We use correlation analysis on preceding and subsequent events after performing the action to infer potential outcomes. This step is equivalent to the correlational analysis with semantic scoring introduced in the previous work [1]. Distinguishing events are those that occur frequently after the action, but they are rarely seen before the action. Additionally, such events are expected to occur rarely after doing the reverse action. In order to perform such correlation analysis we need to temporally align the user timelines such that doing the action or the reverse action occur at $t = 0$ as illustrated in Fig 3, the task is to find events that are more likely to occur in one quadrant (E^+ for $t > 0$) than in its immediately neighboring quadrants (E^- for $t > 0$ and E^+ for $t < 0$). The relative likelihood of an event occurrence in q as compared to u is calculate this as $S_{q,u}(e) = \frac{\tilde{p}_{q,u}(e)}{\hat{p}_u(e)}$ where $\hat{p}_u(e) = \frac{N_q(e)}{|N_q|}$ and $N_q(e)$ is the number of occurrences of an event e in a given quadrant, and $|N_q|$ is the total number of events in a quadrant. Also, $\tilde{p}_{q,u}(e)$ is the Laplace-smoothed probability $\tilde{p}_{q,u}(e) = \frac{N_q(e)+\hat{p}_u(e)m}{|N_q|+m}$.

We apply pair-wise comparison of likelihoods of an event occurrence between the target quadrant $E^+_{t>0}$ and the neighboring quadrants $E^+_{t<0}$ and $E^-_{t>0}$. For an event to be distinguishing the minimum likelihood value between the two comparison should be much greater than one. We use RL (relative likelihood) to select top k% distinguishing events (k=30% in our experiments).

2) Ranking Distinguishing Events: Although distinguishing events ranked by RL are useful, they may not always represent significant events. For example, naming a cat is a distinguishing event, but it is not significant enough to be in the pros-and-cons table. Therefore, we apply our SS to rank events selected from the previous step (Sec. III-C).

3) Categorizing Events: The final step is to categorize the ranked events into two categories of pros and cons. In each iteration, we calculate sentiment score for the next top event and push it into the pros or cons list if the sentiment score condition is met. The pros list accepts events with sentiment score $+0.5$ or larger, and the cons list accepts events with sentiment score -0.2 and smaller. The loop stops once both pros and cons lists acquire *m* events (in our experiments we use $m = 5$).

Algorithm 3 RankAndCategorize($events, weights_{SS}, size$ $posSentThr, negSentThr$)

```
1:  $rlScores \leftarrow calcRL(events)$
2:  $topEvents \leftarrow getTopKEvents(events, rlScores)$
3:  $ssScores \leftarrow calcSS(topEvents, weights_{SS})$
4:  $rankedEvents \leftarrow rankWithSS(topEvents, ssScores)$
5:  $pros \leftarrow []; cons \leftarrow []$
6:  repeat
7:      $e \leftarrow getNextEvent(rankedEvents)$
8:      $s_e \leftarrow calcSentimentScore(e)$
9:      if $s_e > posSentThr$ & $len(pros) < size$ then
10:         $pros \leftarrow e$
11:     else if $s_e < negSentThr$ & $len(cons) < size$ then
12:         $cons \leftarrow e$
13: until $(len(pros) = len(cons) = size)$
14: return $pros, cons$
```

Alg. 3 summarizes the steps for ranking and categorizing events discussed in Sec III-D. The inputs are the events, including both verb and adjective events, weights to calculate SS, the pros-and-cons table size, and the sentiment thresholds for categorizing events into pros and cons. Each event contains a pair of the word (verb or adjective) and the representative message. First, top-k events with highest RL scores are selected (line 1-2). Then the top (distinguishing) events are ranked in decreasing order by SS applied to their messages (line 3-4). Next, sentiment score is calculated for each next event's message from the top of the ranked list (line 7-8). If an event's sentiment score falls in the sentiment threshold conditions, it is added to the corresponding list (i.e. pros or cons) (line 9-12). Finally, the pros and cons lists are returned (line 14).

IV. Experimental Evaluation

In this section, we evaluate our algorithm and compare the results with those obtained from the algorithm from Kiciman and Richardson [1] that we call KR15 hereafter.

A. Evaluation Criteria

We evaluate effectiveness of the pros and cons extracted by our technique compared to those of the KR15 algorithm [1]. Each event in the output table consists of an action (verb) or characteristic (adjective) with a representative message. In KR15, each event consists of a phrase and an example representative message. Our evaluation criterion is the extent to which events extracted by each algorithm indicate meaningful pros and cons. To establish the ground truth we asked three evaluators who are graduate students in computer science and engineering fields, and are not authors of this paper, to categorize each of the outputted messages into one of three classes {pro, con, neither} based on their personal opinion. Each message's label was then decided based on the majority of the three opinions. To avoid bias toward any algorithm, messages selected by the different algorithms were merged into one set before evaluation.

We measure precision for each generated pros-and-cons table as the percentage of messages identified by the evaluators as either pros or cons. We do not count a message in our precision score if it is in the wrong category. For instance, a pro in the cons list counts as a mistake of the algorithm. Since it would require the evaluators to identify pros and cons from lots of messages from the users, we do not calculate recall.

Moreover, we calculate Discounted Cumulative Gain (DCG) to quantify the ordering quality of the (events) messages in the pros-and-cons table T. We expect that more relevant messages appear higher in the table. DCG for a pros-and-cons table is calculated as an average between DCG of pros and cons lists: $DCG_T = \frac{1}{2}\sum_{i=1}^{Pros}\frac{rel_i}{log_2(1+i)} + \frac{1}{2}\sum_{i=1}^{Cons}\frac{rel_i}{log_2(1+i)}$, where rel_i is relevance of the message with rank i. Relevance is 1 when m is a pro in the pros list or a con in the cons list. It is 0 otherwise. DCG_T is the overall DCG score of table T.

B. Data

We use different social media sources in each part of our system. We discuss each data source next:

1) Twitter: The main source is Twitter where we collect timelines of users who experienced performing the action query. Two data sets were collected[1] for our experiments based on two action queries:

Cat Adoption: We study the consequences of adopting a cat and collect tweets based on search queries such as "adopted a pet", "got a pet", and "got a new pet", where pet is either "cat", "kitty" or "kitten". The query entity is either "cat", "kitty", or "kitten". We expect our algorithm to discover events and tweets that represent the potential outcomes of adopting a cat in form of pros and cons. We collected ~1.8 million tweets from 980 users who adopted a cat in March and May of 2016. For each user, we collected their timeline from three months before and after adoption.

Buying IPhone 6: The action query, in this case, is buying an IPhone. We collect tweets based on search queries like "bought an IPhone 6", "got a new IPhone 6", where the query entity is IPhone 6. We expect our algorithm to find events and tweets that represent potential outcomes of buying an IPhone 6 cellphone in form of pros and cons. We collected ~2.2 million tweets from 1420 users who purchased an IPhone 6 in January and February of 2017. We collected each user's timeline from three months before and after they purchased an IPhone 6.

2) Reddit: We used Reddit data in form of posts and comments about both query entities (cat and IPhone) to train the Adjective Vectors discussed in III-C1. We collected 400 posts about cats. Each post contains 6 comments on average. Also we collected 700 posts about IPhone, where each post contains 7 comments on average. We only captured comments in the first level. In other words, replies to comments were skipped. Because, our goal is to collect responses to the content in the post, and not those to other comments. Ultimately, we create Adjective Vectors to represent 931 unique adjectives about cat, and 1685 unique adjectives about IPhone.

3) GloVe-Common-Crawl and GloVe-Twitter [2]: In our experiments we also use vectors trained by GloVe [8] as an alternative to our Adjective Vectors. GloVe Common Crawl vectors are 300-dimension, trained based on 840 billion tokens and 2.2 million unique words. GloVe Twitter vectors are 200-dimension, trained on 2 billion tweets, 27 billion tokens and 1.2 million unique words.

C. Procedures

We set up our system with four different models for representation of adjectives and verbs: 1) Our Adjective Vectors trained with the Reddit data set for adjectives and Wordnet hierarchies for verbs, 2) GloVe Common Crawl vectors for adjectives and verbs, 3) GloVe Twitter vectors for adjectives and verbs, 4) GloVe Reddit vectors, where we use our Reddit data set to train 200-dimension vectors by the GloVe algorithm.

Furthermore, we employ a *hybrid* approach, based on voting among the four models. The voting process affects the event ranking component discussed in III-D2. After selecting the distinguishing events by *RL*, we sort the messages with respect to the number of votes from the four modes in descending order. Next, the messages are selected by SS. The new ordering assigns higher chance of selection to messages with more votes.

We also implemented the algorithm of Kiciman and Richardson [1] (referred to as KR15 in the results) to be able to compare the results. We used Microsoft Web Language Model API[3] to calculate joint probabilities to represent phrases. Then, agglomerative clustering with distance threshold (d=0.75) was applied to create clusters of phrases. Since the original algorithm (KR15) does not mention a specific way to select the representative message for each event, we assume that it picks a message arbitrarily. But we add another version of this algorithm where we use SS to do the selection. We refer to this version as *KR15+SS* in the results. Moreover, we use the following weights for the factors in SS: ($w_{snt} = 1.0$, $w_{res} = 0.67$, $w_{cmp} = 0.67$, $w_{cov} = 0.1$, $w_{len} = 0.1$)

D. Results on Precision

First we compare the effectiveness of our system with four models, our voting-based hybrid approach, GloVe word vectors with three different data sources, KR15, and KR15 with *SS* score (KR15+SS). Table II illustrates the results. For

[1]URL of our data will be ... [to remain anonymous]

[2]https://nlp.stanford.edu/projects/glove/

[3]https://azure.microsoft.com/en-us/services/cognitive-services/web-language-model/

Table II: Precision of algorithms on the Cat and IPhone6 data; IMP is relative improvement over KR15

Algorithm	Cat	IPhone6	Avg	IMP
KR15	50%	30%	40%	—
KR15+SS	60%	60%	60%	50%
AdjectiveVectors+Reddit	**90%**	**80%**	**85%**	**113%**
GloVe+CommonCrawl	80%	70%	75%	88%
GloVe+Twitter	**90%**	50%	70%	75%
GloVe+Reddit	80%	70%	75%	88%
Hybrid	**90%**	70%	80%	100%

Table III: DCG of the algorithms on the Cat and IPhone6 data set; IMP is relative improvement over KR15

Algorithm	Cat	IPhone6	Avg	IMP
KR15	1.52	0.91	1.22	—
KR15+SS	2.08	1.93	2.01	65%
AdjectiveVectors+Reddit	2.64	**2.42**	**2.53**	**107%**
GloVe+CommonCrawl	2.44	1.94	2.19	80%
GloVe+Twitter	**2.71**	1.52	2.12	74%
GloVe+Reddit	2.45	2.21	2.33	91%
Hybrid	**2.71**	2.25	2.48	103%

the Cat data, AdjectiveVectors+Reddit, GloVe+Twitter, and Hybrid outperform the others. GloVe+CommonCrawl and Glove+Reddit show slightly lower performance (80%) than the best algorithms. However, they outperform KR15 and KR15+SS. Hybrid does not produce results better than the individual algorithms. For the IPhone data, AdjectiveVectors+Reddit outperforms all other algorithms. The third column in Table II shows the average precision on the two data sets. Overall, AdjectiveVectors+Reddit outperforms other algorithms and shows 113% relative improvement over KR15.

1) WordVectors+Reddit vs GloVe: AdjectiveVectors+Reddit are more effective than GloVe in general. One reason is GloVe was trained with Twitter or CommonCrawl. The Reddit data contains the same context as the action query (cat adoption or buying IPhone6). In such context, the adjectives are used in similar semantic relationship with the query entity (cat or IPhone6). So, the adjective vectors potentially capture more effective semantic representation of adjectives. However, this is not necessarily the case for context-free data like Twitter and CommonCrawl. As a result, distance measurement between words in context-aware word embedding can be more precise, which leads to higher clustering quality. When we retrain GloVe vectors with our Reddit data the precision increases on average, but it is still lower than AdjectiveVectors+Reddit. One potential reason might be that dimension size of AdjectiveVectors+Reddit is larger than GloVe+Reddit. We tried increasing the vector size for GloVe to match that of AdjectiveVectors+Reddit, but we couldn't create such vectors due to a bug in the available implementation of GloVe. We intend to investigate this issue and experiment with longer GloVe vectors in future work.

2) AdjectiveVectors+Reddit vs Hybrid: The Hybrid algorithm underperforms our AdjectiveVectors+Reddit on average. A potential reason is that it works well only when most of the individual algorithms perform well. Specially, as seen in the IPhone6 data results, bad messages selected by weak individual algorithms can mislead the Hybrid algorithm by prioritizing those messages. Hybrid has three mistakes for the IPhone data set, but only one of them involves AdjectiveVectors+Reddit. The first mistake is created by votes from AdjectiveVectors+Reddit, GloVe+Reddit, and GloVe+Twitter. The second one is generated by votes from the three GloVe models. The third mistake is produced by votes from GloVe+CommonCrawl and GloVe+Reddit. From these mistakes, we observe that the same messages are selected by different variations of GloVe. One potential reason is that only the training data is different, but the training algorithm GloVe and the ranking method SS are the same.

3) AdjectiveVectors+Reddit vs KR15/KR15+SS: AdjectiveVectors+Reddit outperforms KR15 because of three reasons: First, it selects messages with actions and characteristics. This is not done in KR15 (discussed in Sec. IV-G). Second, it creates clusters of higher quality, mainly because our Adjective Vectors for adjectives and Wordnet for verbs establish stronger semantic relationship between words (discussed in Sec. IV-H). Moreover, we use SS to identify the significant representative messages. We also use SS to rank the significant events after selecting the distinguishing ones with RL. In KR15+SS, we use SS to identify representative messages for each event generated by KR15. Although it is helpful, the precision does not increase much. This is likely because events ranked by RL are not as significant as those ranked by SS.

4) KR15 vs KR15+SS: In this case the only difference between the two algorithms is that the former uses an arbitrary representative message, but the latter uses SS to do so. The precision increases 20% when we use SS. Although the extracted events and the ranking of those events are the same, SS is able to select better representative messages for the same events.

E. Results on DCG

Table III shows the ranking effectiveness of pros-and-cons based on DCG. In the Cat data, GloVe+Twitter and Hybrid outperform other models. AdjectiveVectors+Reddit stands second. Moreover, it is observed that Hybrid has picked the best ranking among the individual models. In the IPhone6 data set, AdjectiveVectors+Reddit outperforms other models and shows 107% relative improvement over KR15.

1) AdjectiveVectors+Reddit vs GloVe word vectors: Since our four models (AdjectiveVectors+Reddit, GloVe+CommonCrawl, GloVe+Twitter, GloVe+Reddit) all use SS for both selection of the representative messages and ranking of the events, the difference among their DCG scores is mostly due to the difference in precision.

2) AdjectiveVectors+Reddit vs Hybrid: AdjectiveVectors+Reddit outperforms Hybrid in terms of DCG, on average. However, this is not an effect of ranking, because Hybrid shows lower precision than AdjectiveVectors+Reddit on the IPhone6 data, as shown in Table II.

3) AdjectiveVectors+Reddit vs KR15/KR15+SS: AdjectiveVectors+Reddit outperforms both KR15 and KR15+SS because it uses SS to rank the significant events after selecting the distinguishing ones with RL. The mistakes by AdjectiveVectors+Reddit on the Cat and IPhone6 data occur as high as the second row of cons list. However, KR15 and KR15+SS have more mistakes and they occur as high as the first row.

4) KR15 vs KR15+SS: Although the extracted events and the ranking of those events are the same, our SS finds better representative messages for each event. As a result, many of the mistakes are corrected (20% increase in precision). Since RL is used for ranking of events in KR15+SS, SS can only improve DCG through increasing precision.

F. Examples of Pros-and-Cons tables

1) Cat Adoption: Tables. IVa and IVb illustrate top five pros and cons generated by KR15 and AdjectiveVectors+Reddit on the Cat data respectively. The events are ranked by RL score in KR15. However, they are ranked by SS in AdjectiveVectors+Reddit. SE represents sentiment score (1=good, -1=bad), and GT illustrates the ground truth based on the majority vote on the message by the evaluators.

Overall, the events extracted by AdjectiveVectors+Reddit represent outcomes of higher quality compared to those extracted by KR15. Although our events are single words (verbs or adjectives), they generally express more meaningful traits of cats compared to the ones extracted by KR15. For instance, if we only use the event column to report as a summary of pros and cons of adopting a cat, the ones reported by AdjectiveVectors+Reddit are more informative than those reported by KR15.

The only mistake from AdjectiveVectors+Reddit occurs in the second row of cons. The reason to select event *"lazy"* as a con goes back to identifying the representative message for a cluster via SS. *"lazy"* belongs to a cluster of three adjectives $\{$*lazy, obese, fat*$\}$. Looking at the messages within the cluster we find one alternative that could have been picked: *"our fat cat had to be put down. He was just in too much pain."* In this case, the associated event would be "fat". The SS values for the message that appears in our cons list and the alternative message are 3.90 and 3.69 respectively. We observe two main reasons for this undesired selection: First, the sentiment scores of the two messages are -0.57 and -0.51 respectively whereas the alternative message conveys much more negative meaning compared to the selected message. Therefore, the sentiment module [15] is not able to evaluate the alternative message effectively. This could be improved by retraining on different data sets, or using more accurate sentiment analysis algorithms. The second reason that the selected message gains higher SS value is that it contains reasoning token *"because"*. Although, the reasoning is correctly identified in the selected message, the alternative message also implies some reasoning that is not captured by SS. In fact, the reasoning token in the alternative message is invisible because the author used two sentences: The first one being the fact, and the second one being the reason. This is a drawback in SS that we aim to address in future work.

2) Buying IPhone6: Tables IVc and IVd illustrate the top five pros and cons generated by KR15 and AdjectiveVectors+Reddit on the IPhone6 data respectively. The table structure is the same as those for the Cat data.

Overall, the events extracted by AdjectiveVectors+Reddit represent outcomes of higher quality compared to those extracted by KR15. In some cases, the event word (verb or adjective) is not informative if used alone. However, the outcomes can be observed more clearly when the associated representative messages are viewed.

The two mistakes from AdjectiveVectors+Reddit occur in the second and third rows of the pros list. The reason to select the second message in the pros table is its high SS value (4.51). Its sentiment score (0.79) plays an important role in the SS value. But it seems that this large positive score is due to the words (e.g. *"like"*, *"pretty"*) that do not imply any positive sentiment in this message. This message contains a reasoning factor (*"because"*), but it is not significant enough to represent an event. The second mistake of AdjectiveVectors+Reddit is the third message in the pros table (event: *restore*) which is selected due to its high SS value. The abbreviation ("*lm*o*") that indicates humor increases the positive sentiment drastically. However, the sentiment score and SS would be 0.3 and 2.09 respectively without the abbreviation. An alternative message with the next highest SS value (3.89) is *"I've just updated my iphone 6 to allow for native wifi calling but the voice quality vs the app is drastically worse."*. The cluster to which this message belongs contains $\{$*restore, update, reboot, reset*$\}$, and the event in this case would be verb *"update"*. In addition to its large negative sentiment score (-0.57), this message contains a comparison token *"worse"* which is captured by SS because its POS tag is *JJR*. Semantically, it seems that this message could potentially represent a con of buying IPhone6.

3) Cat Adoption vs Buying IPhone6: The task of identifying pros and cons of buying IPhone6 is more difficult than that of cat adoption. Unlike cats, cellphones have many features like screen, battery, apps. Users might express their experience about any individual entity which makes it harder to identify actions and characteristics because they don't directly refer to cellphone. Moreover, sentiment analysis on subjects like cellphone becomes difficult. An off-the-shelf sentiment algorithm calculates the overall sentiment of the

Table IV: Example Pros & Cons table generated by algorithms: (a) KR15 on the Cat data, (b) AdjectiveVectors+Reddit on the Cat data, (c) KR15 on the IPhone6 data, (b) AdjectiveVectors+Reddit on the IPhone6 data. SE represents sentiment score (1=good, -1=bad), and GT (ground truth) illustrates the majority vote on the message by the evaluators (P=pro, C=con, N=neither).

(a) Algorithm: KR15 – Data: Cat

	Pros					Cons				
	Event	Representative Message	SE	RL	GT	Event	Representative Message	SE	RL	GT
1	adorable cat	[...] I've adopted an adorable cat within a span of a month. life is great.	0.81	6.2	P	cat play	My cat doesn't play nice with the dogs [...] so he's in the bedroom for most of the weekend	-0.3	6.1	N
2	my cat is	my cat is literally curious about anything i eat xd.	0.63	5.9	P	ignore me	Accidentally punched my cat in the nose. He's going to ignore me and make me feel guilty [...]	-0.63	5.5	C
3	kitten watched	My kitten watched her namesake get the win and come 1 game away from the.	0.59	5.2	N	kitten is sad	looking back through the window, it seemed my kitten was sad to see me go to work.	-0.45	5.1	N
4	wake up	The entire room just screamed "THE HOUND" and my cat didn't wake up so she's super cool.	0.66	4.8	N	stupid enough	my cat is stupid enough to sleep while eating.	-0.55	4.6	P
5	my kitten is	My kitten is definitely winning.	0.53	4.3	P	tearing up	My cat's tearing up my room trying to kill a fly.	-0.69	4.1	C

(b) Algorithm: AdjectiveVectors+Reddit – Data: Cat

	Pros					Cons				
	Event	Representative Message	SE	SS	GT	Event	Representative Message	SE	SS	GT
1	affectionate	Life is better with a cat. Tigger is affectionate and would make a great lap cat. [...]	0.91	4.94	P	smells	My cat is sleeping in my volleyball bag and I feel bad for him because it smells so bad.	-0.79	4.51	C
2	sweet	I'm so happy [...] that my sweet kitty came back home to me, I missed you so much sweet girl.	0.91	4.93	P	lazy	My cat is so lazy he just dragged himself across my bed because he didn't want to get up om*g.	-0.57	3.90	N
3	cuddles	I love coming home and going to bed because my cat cuddles with me. She is so lovely!	0.88	4.81	P	ignore	Accidentally punched my cat in the nose. He's going to ignore me and make me feel guilty [...]	-0.77	3.52	C
4	mews	Aww. My cat mews so cute. I love him so much.	0.85	4.67	P	claws	My kitten claws my couch and attacks my baby... not so sure I like him anymore.	-0.77	3.43	C
5	hungry	When my cat is hungry, [...] she just puts on her best Im starving face and stares at me.	0.53	3.80	P	mad	My cat is so mad at me being that I took her to the vet today.	-0.63	2.99	C

(c) Algorithm: KR15 – Data: IPhone6

	Pros					Cons				
	Event	Representative Message	SE	RL	GT	Event	Representative Message	SE	RL	GT
1	my new iphone	I'm in love with my new iPhone 6 Plus	0.64	7.7	P	iphone charger	listen! if you have an iphone 6 charger [...] i will literally cry because [...] my phone is dead.	-0.83	6.9	N
2	whip out	gotta whip out the iphone 4 since i got my iphone 6 taken away lmao help me.	0.77	6.8	N	unlock it	[...] i can give you my iphone 6 and i'll unlock it.	-0.88	5.9	N
3	got my new	i got my new phone today [...] still on this iphone 6 cuz i haven't ported my number lol.	0.52	6.2	N	getting my iphone	[...] my sister is 6 and she"s getting my iPhone 6 in two days and has no clue.	-0.78	5.3	C
4	using my iphone	if y'all text me my phone is restoring rn so i'm using my iphone 6 on wifi lmao hit me on here.	0.73	5.5	N	had iphone	my brother [...] got the iphone 6 had it for one day broke it and my mom now got him the 7.	-0.68	3.4	N
5	working perfectly	got my new iphone 6 working perfectly!	0.67	4.5	P	second phone	limited budget it's just a second phone for my kink life. i'm currently using an iphone6.	-0.23	3.1	C

(d) Algorithm: AdjectiveVectors+Reddit – Data: IPhone6

	Pros					Cons				
	Event	Representative Message	SE	SS	GT	Event	Representative Message	SE	SS	GT
1	greatest	[...] my iphone 6 plus was the greatest phone i've ever had. [...]	0.92	6.97	P	trying	hi i'm trying on my iphone6 [...] stuck with an error message at any point i've tried many times!	-0.61	4.03	C
2	turn on	like my old iphone 6 wouldn't turn on and i'm pretty sure it was bc if the last jailbreak i had..	0.79	4.51	N	loses	[...] since updating today my iphone6 loses power rapidly.	-0.61	3.99	C
3	restore	if y'all text me my phone is restoring rn so i'm using my iphone 6 on wifi lm*o hit me on here.	0.73	4.33	N	rebooting	my phone on iphone 6 jailbrake just keeps rebooting my phone randomly. so annoying!	-0.58	3.85	C
4	survived	dropped my naked iPhone 6 into a toilet and it survived so today has been pretty ok.	0.69	4.24	P	went	my iphone 6 went stupid and i can't get an appt at apple till saturday, [...].	-0.53	3.78	C
5	waterproof	my phone just fell in the tub and the music continued to play [...] the iphone6 is waterproof.	0.64	4.06	P	stupid	i love my apple iphone 6plus. but there is no [...] way i'll spend $1k [...] it's just a stupid phone.	-0.86	2.77	C

sentence. However, the user's sentiment about the entity (e.g. IPhone) or its features might be different than the overall sentiment of the message. This issue has happened in the third pro suggested by AdjectiveVectors+Reddit on the IPhone6 data. The message is quite neutral about IPhone. However, the abbreviation indicates humor that evidently is not related to IPhone. This issue in sentiment analysis potentially decreases ranking performance of our algorithm as the SS mechanism becomes less accurate. We intend to study this issue in future work.

G. Selecting Messages with Actions or Characteristics

Selecting relevant messages (discussed in Sec. III-B2) brings the focus of our successive components on actions and characteristics about the query entity. Tables IVb and IVd show some of the selected messages. We eliminate a large number of messages that do not contain observations or opinions about the query entity. Some examples are: *"here is a photo of my cat taken by my friend"*, *"Tips for the first 30 days of cat adoption"*, *"Now tweeting by brand new*

Table V: Examples of clusters and events created by (a) KR15 and (b) AdjectiveVectors+Reddit

(a) KR15

Cluster	Event	Representative Message
cat is adorable, adorable cat	adorable cat	[...] I've adopted an adorable cat within a span of a month. life is great.
got my cat, my cat hobbes, my cat is, my cat	my cat is	my cat is literally curious about anything i eat xd.
stupid enough	stupid enough	my cat is stupid enough to sleep while eating.

(b) AdjectiveVectors+Reddit

Cluster	Event	Representative Message
satanic, annoying, demon, insane, bad, mad, evil, mean	mad	My cat is so mad at me being that I took her to the vet today.
gorgeous, sweet, happy, lovely, pretty, sad, friendly	sweet	I am so happy [...] that my sweet kitty came back home to me, [...]
bite, pass, grab, claw, cross, hog	claws	My kitten claws my couch and attacks my baby [...]

iPhone6", and *"I'm on iPhone 6 in my bed"*.

H. Clustering Quality of AdjectiveVectors+Reddit vs KR15

Tables Va and Vb depict examples of clusters including events and messages extracted by KR15 and AdjectiveVectors+Reddit respectively. We observe that our AdjectiveVectors+Reddit is able to create clusters containing multiple words that are semantically homogeneous. However, the clusters created by KR15 are generally smaller and contain phrases that could represent unrelated meanings (e.g. "got my cat", "my cat hobbes"). Therefore, the representative message and the associated event selected by AdjectiveVectors+Reddit is a better representation for the cluster.

V. Concluding Remarks

We propose actions (verbs) and characteristics (adjectives) to select relevant messages. Also, we propose adjective vectors to represent adjectives and Wordnet entities to represent verbs. As a result, we create clusters of verbs/adjectives of higher quality. We then select a representative message and event for each cluster using SS. We also apply SS in ranking of the events in the final pros-and-cons table. According to precision and DCG on two data sets, the pros and cons discovered by our algorithm are more meaningful than those identified by an existing algorithm (KR15) [1].

References

[1] E. Kıcıman and M. Richardson, "Towards decision support and goal achievement: Identifying action-outcome relationships from social media," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 547–556.

[2] M. Myslín, S.-H. Zhu, W. Chapman, and M. Conway, "Using twitter to examine smoking behavior and perceptions of emerging tobacco products," *Journal of medical Internet research*, vol. 15, no. 8, p. e174, 2013.

[3] M. Richardson, "Learning about the world through long-term query logs," *ACM Transactions on the Web (TWEB)*, vol. 2, no. 4, p. 21, 2008.

[4] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," *Journal of Computational Science*, vol. 2, no. 1, pp. 1–8, 2011.

[5] A. Olteanu, O. Varol, and E. Kıcıman, "Distilling the outcomes of personal experiences: A propensity-scored analysis of social media," in *Proc. of The 20th ACM Conference on Computer-Supported Cooperative Work and Social Computing*, 2017.

[6] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American society for information science*, vol. 41, no. 6, p. 391, 1990.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[8] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[9] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit." in *ACL (System Demonstrations)*, 2014, pp. 55–60.

[10] C. Fellbaum, "Wordnet: an eletronic lexical database. cambridge, massachusetts, eua," 1998.

[11] M. Lesk, "Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone," in *Proceedings of the 5th Annual International Conference on Systems Documentation*, ser. SIGDOC '86. New York, NY, USA: ACM, 1986, pp. 24–26. [Online]. Available: http://doi.acm.org/10.1145/318723.318728

[12] S. Banerjee and T. Pedersen, "An adapted lesk algorithm for word sense disambiguation using wordnet," in *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, 2002, pp. 136–145.

[13] G. Hirst, D. St-Onge *et al.*, "Lexical chains as representations of context for the detection and correction of malapropisms," *WordNet: An electronic lexical database*, vol. 305, pp. 305–332, 1998.

[14] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[15] C. J. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," in *Eighth international AAAI conference on weblogs and social media*, 2014.

[16] N. Jindal and B. Liu, "Mining comparative sentences and relations," in *AAAI*, vol. 22, 2006, pp. 1331–1336.