Constructing Explainable Opinion Graphs from Reviews

Nofar Carmeli*
Technion
Haifa, Israel
snofca@cs.technion.ac.il

Xiaolan Wang
Megagon Labs
Mountain View, USA
xiaolan@megagon.ai

Yoshihiko Suhara
Megagon Labs
Mountain View, USA
yoshi@megagon.ai

Stefanos Angelidis
University of Edinburgh
Edinburgh, UK
s.angelidis@ed.ac.uk

Yuliang Li
Megagon Labs
Mountain View, USA
yuliang@megagon.ai

Jinfeng Li
Megagon Labs
Mountain View, USA
jinfeng@megagon.ai

Wang-Chiew Tan†
Facebook AI
Menlo Park, USA
wangchiew@fb.com

ABSTRACT

The Web is a major resource of both factual and subjective information. While there are significant efforts to organize factual information into knowledge bases, there is much less work on organizing opinions, which are abundant in subjective data, into a structured format.

We present ExplainIt, a system that extracts and organizes opinions into an opinion graph, which are useful for downstream applications such as generating explainable review summaries and facilitating search over opinion phrases. In such graphs, a node represents a set of semantically similar opinions extracted from reviews and an edge between two nodes signifies that one node explains the other. ExplainIt mines explanations in a supervised method and groups similar opinions together in a weakly supervised way before combining the clusters of opinions together with their explanation relationships into an opinion graph. We experimentally demonstrate that the explanation relationships generated in the opinion graph are of good quality and our labeled datasets for explanation mining and grouping opinions are publicly available at https://github.com/megagonlabs/explainit.

CCS CONCEPTS

• **Information systems → Web mining**.

KEYWORDS

Opinion mining, explanation, opinion graph construction

ACM Reference Format:
Nofar Carmeli, Xiaolan Wang, Yoshihiko Suhara, Stefanos Angelidis, Yuliang Li, Jinfeng Li, and Wang-Chiew Tan. 2021. Constructing Explainable Opinion Graphs from Reviews. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3442381.3450081

*Work done during internship at Megagon Labs.
†Work done while at Megagon Labs.

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450081

1 INTRODUCTION

The Web is a major resource for people to acquire information, whether factual or subjective. In recent years, there have been significant advances in extracting facts in the form of subject-predicate-object triples and constructing knowledge bases of such facts [12, 28, 29, 39]. In comparison, there are much less efforts around constructing organized knowledge bases of opinions [5], which are abundant in subjective data, such as reviews and tweets. In fact, according to a recent study[1], more than 90% of customers read reviews before committing on visiting a business or making a purchase. A natural question is thus the following: is there a systematic way to organize opinions into knowledge bases that will make it easier for customers to understand the opinions found in subjective data?

Existing opinion mining techniques [19, 25, 32–34, 41] cannot be directly applied to organize the extracted opinions. First, they largely focus on improving the accuracy of opinion extraction and aspect-based sentiment analysis of the extracted opinions over a set of predefined aspects. They cannot be used, in particular, to determine the relationships between opinions. For example, while they can determine the sentiment of an extracted opinion "*very good location*", they cannot explain why the location is very good in relation to other extracted opinions. Furthermore, simply collecting all extracted opinions will result in a lot of redundancy and may also lead to incorrect conclusions. For example, if the list of all extracted opinions are {"*quiet room*", "*very noisy street*", "*loud neighborhood*", "*horrible city noise*", "*quiet room*"}, one can incorrectly conclude that "*quiet room*" is the most popular opinion if the opinions are not organized according to similarity. An early attempt [5] that produces knowledge bases of opinions does not fully address the above limitations since it does not de-duplicate similar opinions, nor considers the direction of explanation between opinions.

With the above observations, we asked ourselves the following question: *Can we go beyond opinion mining to represent opinions and the relationships among them uniformly into a knowledge base?* To understand how best to organize opinions into a knowledge base, we analyzed the properties of subjective information in reviews through a series of annotation tasks and confirmed that:

- *Opinions phrases*, or opinions in short, are pairs of the form (opinion term, aspect term) such as ("*very good*", "*location*"). Opinions are the most common expression for subjective information

[1] https://fanandfuel.com/no-online-customer-reviews-means-big-problems-2017/

Review #1, Score 85, Date: Feb. 30, 2020
Pros: Friendly and helpful staff. Great location. Walking distance to Muni bus stops. Not too far away from Fisherman's Wharf, Aquatic Park. Cons: extremely noisy room with paper thin walls.

Review #2, Score 80, Date: Feb. 30, 2019
Good location close to the wharf, aquatic park and the many other attractions. the rooms are ok but a bit noisy, loud fridge and AC.

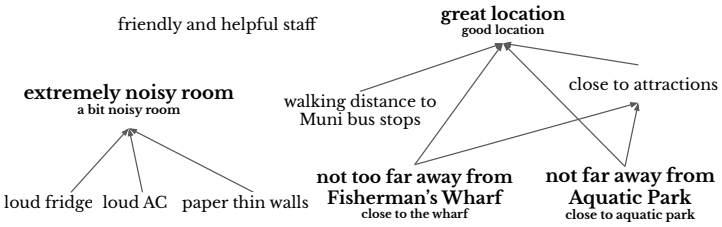


Figure 1: Opinion Graph (right) based on opinions extracted from reviews (left). A node explains its parent node.

in reviews. In 100 random review sentences that we annotated, we observed that 84.75% of subjective information is in this form.

• *Explanation*, or inference, is the most common relationship between opinions that are correlated in reviews. We annotated opinions that co-occur in 40K random review sentences[2] and observed that 12.3% of the opinions are correlated under some relationship (e.g., one explains/contradicts/paraphrases the other). Among these opinions, 74.2% of the opinions are related under the explanation relationship, which is the focus of this paper.

• Many opinions and relationships between opinions are oriented around specific entities, not across multiple entities. For example, the opinion "*close to main street*" explains "*very noisy room*" for a specific hotel in the review "Our room was very noisy as it is close to the main street". However, this explanation may not be true for arbitrary hotels.

Based on this analysis, we propose a graph representation for organizing opinions, called the *Opinion Graph* that organizes opinions around the explanation relationship based on reviews specific to an entity. A node is an opinion of the form (opinion term, aspect term) and consists of all opinions that are close to the node according to their semantic similarity. An edge (u,v) between two nodes u and v denotes that u explains v. We found this to be a versatile structure for organizing opinions of reviews because (a) the opinion graph is a concise and structured representation of the opinions over lots of reviews, (b) the nodes can aggregate and represent opinions at different levels of granularity, (c) the edges explain the opinions based on other opinions that appear in the reviews, (d) the provenance of opinions in nodes can be traced back to the input reviews where they are extracted from, and (e) the opinion graph is a useful abstraction that supports a series of downstream applications, from the generation of explainable review summaries to facilitating search over opinion phrases or criteria [22].

The right of Figure 1 shows an opinion graph that is generated from the hotel reviews on the left of the figure. Each node in the graph represents a set of semantically similar opinions. Each opinion consists of an opinion term, followed by an aspect term. For example, "*good location*", where *"good"* is an opinion term and *"location"* is an aspect term. Each edge represents the explanation relationship between opinions. For example, "*paper thin walls*" explains "*extremely noisy room*". This opinion graph enables one to easily create a customized summary of the reviews by using the entire graph, or only for portions of the graph, such as which attractions the hotel is in close proximity with. Moreover, end users or downstream applications can navigate aspects and opinions based on their specific needs and seek explanations of the extracted opinions, e.g., understand why the hotel is *"extremely noisy"* or why it is in a *"great location"*. A prototype based on this application has been demonstrated [38].

Opinion Graphs are constructed from reviews through a novel opinion graph construction pipeline ExplainIt, which we will present in this paper. To the best of our knowledge, ExplainIt is the first pipeline that can extract and organize both opinions and their explanation relationships from reviews. It is challenging to construct an opinion graph from reviews. First, the review sentences are inherently noisy and can be nuanced. Hence, mining opinions and the explanation relationships between them can be difficult. Second, all opinions and their predicted explanation relationships need to be integrated into one opinion graph, while taking into account potential inaccuracies and the noise and nuances inherent in languages. To summarize, we make the following contributions:

• We develop ExplainIt, a system that generates an opinion graph about an entity from a set of reviews about the entity. ExplainIt (a) mines opinion phrases, (b) determines the explanation relationships between them, (c) canonicalizes semantically similar opinions into opinion clusters, and (d) generates an opinion graph for the entity from the inferred explanation relationships and opinion clusters. In particular, our technical contributions include the explanation classifier, the in-domain training data, and the opinion phrase learning mechanism for opinion canonicalization.

• We evaluate the performance of ExplainIt through a series of experiments. We show that our explanation classifier performs 5% better than a fine-tuned BERT model [11] and 10% better than a re-trained textual entailment model [30]. We show that learned opinion phrase representations are able to improve existing clustering algorithms by up to 12.7% in V-measure. Finally, our user study shows that human judges agree with the predicted graph edges produced by our system in more than 77% of the cases.

• Our crowdsourced labeled datasets (in the hotel and restaurant domains) for two subtasks (mining explanation relationships and canonicalizing semantically similar opinion phrases) that we use for training and evaluation are publicly available at https://github.com/megagonlabs/explainit.

Outline We give an overview of ExplainIt in Section 2. We present the component for mining explanation relationships in Section 3. We demonstrate how we canonicalize similar opinion phrases in

[2]Under the Appen platform (https://appen.com/).

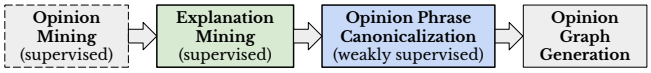


Figure 2: ExplainIt pipeline. The explanation Mining module and the Opinion Phrase Canonicalization module form our major technical contributions.

Section 4 and how we construct an opinion graph in Section 5. We evaluate ExplainIt in Section 6. We outline related work in Section 7 and conclude this paper in Section 8.

2 PRELIMINARIES

An *opinion phrase* is a pair $p = (o, a)$, where o is the *opinion term*, and a is the aspect term o is referring to. For example, the last sentence of Review #1 in Figure 1, "Cons: extremely noisy room with paper thin walls", contains two opinion phrases: p_1 = ("*extremely noisy*", "*room*"); and p_2 = ("*paper thin*", "*walls*"). An *explanation* $e = (p_i \rightarrow p_j)$ is a relationship between two opinion phrases, where p_i explains p_j. For example, (*"paper thin walls"* → *"extremely noisy room"*) and (*"close to attractions"* → *"great location"*) are two valid explanations.

Definition: An *Opinion Graph* $G = (N, E)$ for a set S of opinion phrases is such that (1) every opinion phrase $p \in S$ belongs to exactly one node $n \in N$, (2) each node $n \in N$ consists of *semantically consistent opinion phrases*, and (3) an edge $(n_i \rightarrow n_j) \in E$ represents a explanation relationship from n_i to n_j. That is, the member phrases of n_i explain the member phrases of n_j.

The right of Figure 1 depicts an opinion graph obtained from the opinion phrases mined from the given reviews. Observe that a "perfect node" would contain paraphrases and two perfect nodes n_i and n_j will be connected with an edge $(n_i \rightarrow n_j) \in E$ if and only if $(p_i \rightarrow p_j)\ \forall\ p_i \in n_i,\ p_j \in n_j$, as is the case in the example of Figure 1. In practice, however, we often deal with imperfect nodes, containing semantically *similar* phrases and we draw an edge between two nodes whenever an explanation relationship between the two nodes is very likely (i.e., when a significant number of explanation relationships exist between opinions in the two nodes).

Our goal is to build an opinion graph $G = (N, E)$ with optimal precision and recall for both the nodes (i.e., the clusters of opinion phrases) and the edges (i.e., the explanations between clusters). It is hard, if at all possible, to produce an opinion graph with a single end-to-end model because of the need for mining both opinion phrases and their relationships, as well as the scale of the problem, which often involves thousands of reviews. Therefore, just like the knowledge base construction pipelines [12, 14, 39], we decompose the opinion graph construction problem into several sub-problems and focus on optimizing each sub-problem individually.

2.1 Opinion Graph Construction Pipeline

Our opinion graph construction method is inspired by methods used in knowledge base construction. We break down the construction of an opinion graph into the four components as illustrated in Figure 2. We provide an overview of each component next.

Opinion Mining The first step mines opinion phrases from a set of reviews about an entity. For this, we can leverage Aspect-based Sentiment Analysis (ABSA) models [32, 33] and, in our pipeline, we use an open-source system [22]. The system also predicts the aspect category and sentiment associated with every opinion phrase. As we describe in Section 4, we exploit these additional signals to improve opinion phrase canonicalization.

Explanation Mining Next, ExplainIt discovers explanation relationships, if any, between pairs of extracted opinion phrases from reviews. We use crowdsourcing to obtain domain-specific labeled data, and develop a supervised multi-task classifier to discover the explanation relationship between two opinion phrases. Our model outperforms a series of baseline approaches [30, 35], including the fine-tuned BERT model [11].

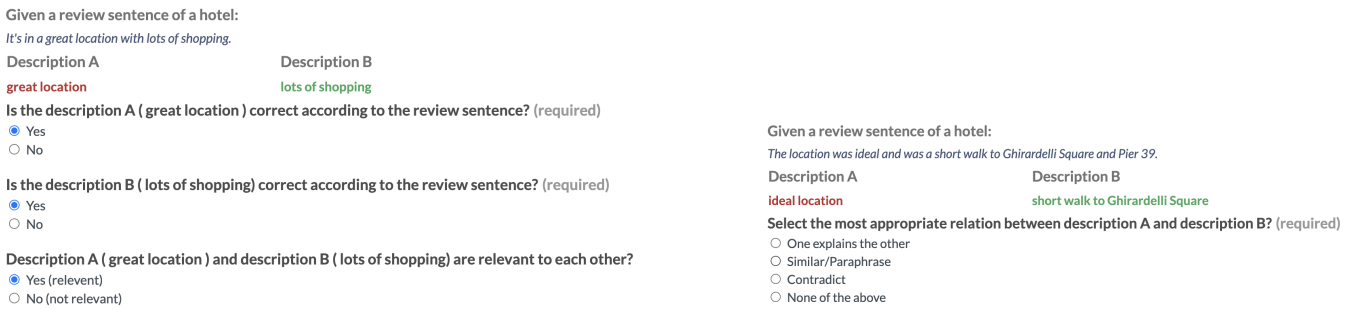
Opinion Phrase Canonicalization Semantically similar opinion phrases are grouped together (e.g., "*not far away from Fisherman's Wharf*" and "*close to the wharf*") to form a node in the opinion graph. This is necessary as reviews overlap significantly in content and, hence, contain many similar opinion phrases. To canonicalize opinion phrases, we develop a novel opinion phrase representation learning framework that learns opinion phrase embeddings using weak supervision obtained from the previous steps, namely predicted aspect categories, sentiment polarity scores, and explanation relationships. Similar to entity canonicalization techniques for open knowledge base construction [9, 37], we apply a clustering algorithm to the learned opinion phrase embeddings to cluster semantically similar opinion phrases to canonicalize those opinion phrases. We demonstrate improvements in the quality of the canonicalization outcome using our learned opinion phrase embeddings.

Opinion Graph Generation Finally, we present an algorithm to construct the final opinion graph. The algorithm constructs an opinion graph by connecting graph nodes according to the aggregated explanation predictions between opinion phrases in the respective nodes. Our user study shows that our method produces graphs that are both accurate and intuitive.

3 MINING EXPLANATIONS

A significant task underlying the construction of an opinion graph is to determine when one opinion phrase explains another. For example, *"close to Muni bus stops"* is an explanation of *"convenient location"*, but is not an explanation for *"close to local attractions"*. Similarly, *"on a busy main thoroughfare"* is an explanation for *"very noisy rooms"* but not necessarily an explanation for *"convenient location"*.

Mining explanations between opinion phrases from reviews is related to two problems: entity relation classification [40, 43] (or relation extraction) and recognizing textual entailment (RTE) [10]. The entity relation classification problem takes a sequence of text and a pair of entities as the input and learns to classify the relationship between the entities with domain-specific training data. As the models are trained and tailored by domain-specific tasks, it is infeasible to directly train the entity relation classification models for our explanation mining task. Recognizing textual entailment (RTE) problem, on the other hand, considers two sequences of text, often referred as premise and hypothesis, and determines whether the hypothesis can be inferred from the premise. Although it also considers the inference relationships between two pieces of text, RTE models trained over general text are still inadequate for mining



(a) Phase-one annotation task

(b) Phase-two annotation task

Figure 3: Human annotation tasks for explanation mining.

explanations from reviews. This is based on two observations: (a) domain-specific knowledge is often necessary to understand the nuances of opinion relationships; (b) in many cases, having access to the full review is crucial to judge potential explanations. In fact, we evaluated a state-of-the-art RTE model [30] trained on open-domain data [8] and observed a very low explanation accuracy of 34.3%. In Section 6, we re-trained both entity relation classification model and RTE models on the review domain and confirmed that they are still less accurate than our proposed model.

In what follows, we first describe how we collect domain-specific data for training through crowd-sourcing and then present our multi-task classifier.

3.1 Collecting Human Annotations

We use a two-phase procedure to collect two domain-specific training datasets for the hotel and restaurant domains. The goal of the first phase is to prune pairs of opinion phrases that are *irrelevant* to each other. In the second phase, the crowd workers label the remaining relevant pairs of opinion phrases. That is, given a pair of relevant opinion phrases, we ask crowd workers to determine if one opinion phrase explains another. If the answer is yes, we ask them to label the direction of the explanation. In both phases, we provide as context, the review where the opinion phrases co-occur to assist crowd workers in understanding the opinion phrases and hence, make better judgments. Figure 3 demonstrates two example tasks for each phase respectively.

We obtained our training data via the Appen crowdsourcing platform[3]. To control the quality of the labels, we selected crowd-workers who can achieve at least 70% accuracy on our test questions. For each question, we acquire 3 judgments and determine the final label via majority vote. For the first phase, we hired 832 crowdworkers and observed a 0.4036 Fleiss' kappa inter-annotator agreement rate [15]; for the second phase, we hired 322 crowdworkers with a 0.4037 inter-annotator agreement rate. As opposed to obtaining labels through a single phase, our two-phase procedure breaks down the amount of work for each label into smaller tasks and therefore renders higher quality annotated data. This is confirmed by our trial run of a single-phase procedure, which only recorded a 0.0800 inter-annotator agreement rate.

[3]https://appen.com/

We obtained $19K$ labeled examples this way with 20% positive examples (i.e., opinion phrases in an explanation relationship). For our experiment, we used a balanced dataset with $7.4K$ examples.

3.2 Explanation Classifier

We observe that the context surrounding the opinion phrases and the word-by-word alignments between the opinion phrases are very useful for our explanation mining task. For example, the opinion phrases *"noisy room"* and *"right above Kearny St"*, may appear to be irrelevant to each other since one is about *room quietness* and the other is about *location*. However, the context in the review where they co-occur, *"Our room was noisy. It is right above Kearny St."* allows us to conclude that *"right above Kearny St"* is an explanation for *"noisy room"*. In addition to context, the word-by-word alignments between opinion phrases can also be very beneficial for explanation mining. For example, from two phrases, *"easy access to public transportation"* and *"convenient location"*, the word-by-word alignments between *"easy access to"* and *"convenient"*, as well as *"public transportation"* and *"location"* makes it much easier to determine that the first phrase forms an explanation to the latter one.

However, existing models do not incorporate both types of information. Relation extraction models for constructing knowledge bases primarily focus on capturing the context between the given opinion phrases and they rarely explicitly consider word-by-word alignments between opinion phrases. RTE models mainly concentrate on aligning words between two pieces of text in the input and ignores the context.

Therefore, to mine the explanations more effectively, we design a multi-task learning model, which we call MaskedDualAttn, for two classification tasks: (1) *Review classification*: whether the review contains explanations; (2) *Explanation classification*: whether the first opinion phrase explains the second one (Figure 4). Intuitively, we want the model to capture signals from the context and the opinion phrases. Our technique, which accounts for both the context surrounding the opinion phrases and the word-by-word alignments between opinion phrases, is a departure from prior methods in open-domain RTE and entity relation classification, which do not consider both information at the same time. Our ablation study confirms that both tasks are essential for mining explanations effectively (Section 6.1). Table 1 summarizes the notations used in this section.

Input and Phrase Masks. The input to the classifier consists of a review $r = (w_1, ..., w_L)$ with L words, and two opinion phrases, p_i and p_j. For each phrase $p = (o, a)$, we create a binary mask, $\mathbf{m} = (m_1, ..., m_L)$, which allows the model to selectively use the relevant parts of the full review encoding:

$$m_i = \begin{cases} 1, & \text{if } w_i \in a \cup o \\ 0, & \text{otherwise.} \end{cases}$$

We denote the binary masks for p_i and p_j as $\mathbf{m}_i$ and $\mathbf{m}_j$ respectively.

Encoding. We first encode tokens in the review r through an embedding layer, followed by a BiLSTM layer. We denote the output vectors from the BiLSTM layer as $H = [h_1, ..., h_L] \in \mathbb{R}^{k \times L}$, where k is a hyperparameter of the hidden layer dimension. We do not encode the two opinion phrases separately, but mask the review encoding using $\mathbf{m}_i$ and $\mathbf{m}_j$. Note that we can also replace the first embedding layer with one of the pre-trained models, e.g., BERT [11]. Our experiment demonstrates that using BERT is able to further improve the performance by 4% compared to a word2vec embedding layer.

Self-attention. There are common linguistic patterns for expressing explanations. A simple example is the use of connectives such as "because" or "due to". To capture the linguistic features used to express explanations, we use the self-attention mechanism [2], which is a common technique to aggregate hidden representations for classification:

$$\mathrm{M} = \tanh(\mathrm{W}^{\mathrm{H}} H + \mathrm{b}^{\mathrm{H}}) \qquad \mathrm{M} \in \mathbb{R}^{k \times L} \tag{1}$$

$$\alpha = \mathrm{softmax}(\mathrm{w}^{\mathrm{T}} \mathrm{M}) \qquad \alpha \in \mathbb{R}^{L} \tag{2}$$

$$\mathrm{h}_r^* = \tanh(H \alpha^{\mathrm{T}}) \qquad \mathrm{h}_r^* \in \mathbb{R}^{k}, \tag{3}$$

where $\mathrm{W}^{\mathrm{H}} \in \mathbb{R}^{k \times k}$, $\mathrm{b}^{\mathrm{H}}, \mathrm{w} \in \mathbb{R}^{k}$ are three trainable parameters. We obtain the final sentence representation as h_r^*.

Alignment attention. Although the self-attention mechanism has a general capability of handling linguistic patterns, we consider it is insufficient to accurately predict the explanation relationship between opinion phrases. Thus, we implement another *alignment attention* layer to directly capture the similarity between opinion phrases. Our alignment attention only focuses on opinion phrases, which is different from the self-attention layer that considers all input tokens, and it has a two-way word-by-word attention mechanism [35] to produce a soft alignment between words in the input opinion phrases. To align p_i with p_j, for each word $w_t \in p_i$, we get a weight vector α_t over words in p_j as follows:

$$d_t = \mathrm{U}^{\mathrm{h}} h_t + \mathrm{U}^{\mathrm{r}} r_{t-1} \qquad d_t \in \mathbb{R}^{k} \tag{4}$$

$$\mathrm{M}_t = \tanh(\mathrm{U}^{\mathrm{H}} H + \underbrace{[d_t; ...; d_t]}_{L \text{ times}}) \qquad \mathrm{M}_t \in \mathbb{R}^{k \times L} \tag{5}$$

$$\alpha_t = \mathrm{softmax}(\mathrm{u}^{\mathrm{T}} \mathrm{M}_t - c\overline{\mathbf{m}_j}) \qquad \alpha_t \in \mathbb{R}^{L} \tag{6}$$

$$r_t = H \alpha_t + \tanh(\mathrm{U}^{\mathrm{t}} r_{t-1}) \qquad r_t \in \mathbb{R}^{k}, \tag{7}$$

where $\mathrm{U}^{\mathrm{H}}, \mathrm{U}^{\mathrm{h}}, \mathrm{U}^{\mathrm{r}}, \mathrm{U}^{\mathrm{t}} \in \mathbb{R}^{k \times k}$ and $\mathrm{u} \in \mathbb{R}^{k}$ are five trainable parameters; $h_t \in \mathbb{R}^{k}$ is the t-th output hidden state of H; $r_{t-1} \in \mathbb{R}^{k}$ is the representation of the previous word; and $\overline{\mathbf{m}_j}$ is the reversed binary mask tensor for p_j. The final presentation of opinion phrase

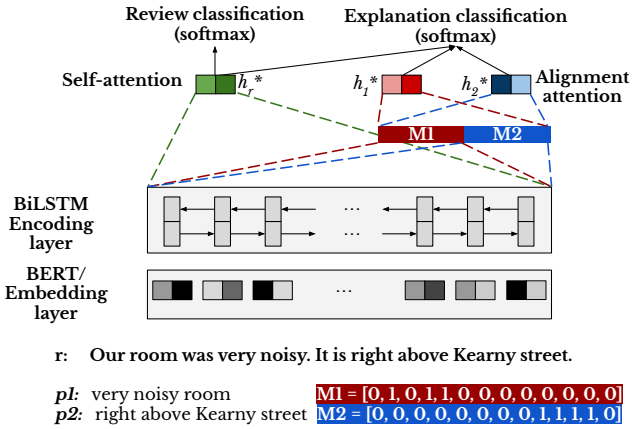


Figure 4: Model architecture of MaskedDualAttn.

Notation	Meaning
$p = (o, a)$	Opinion phrase, opinion term, and aspect term
L	Sequence length of a review r
$\mathbf{m}_i, \mathbf{m}_j$	Binary masks of the phrases p_i and p_j
k	Size of hidden states
$H = [h_1, ..., h_L]$	Output hidden states from BiLSTM layer
$\mathrm{W}^{\mathrm{H}}, \mathrm{b}^{\mathrm{H}}, \mathrm{w}$	Self-attention trainable weights
$\mathrm{U}^{\mathrm{H}}, \mathrm{U}^{\mathrm{h}}, \mathrm{U}^{\mathrm{r}}, \mathrm{U}^{\mathrm{t}}, \mathrm{u}, \mathrm{U}^{\mathrm{x}}, \mathrm{U}^{\mathrm{y}}$	Alignment-attention trainable weights

Table 1: Notations for explanation classifier.

p_i is obtained from a non-linear combination of p_i's last hidden state $h_{|p_i|}$ and last output vector $r_{|p_i|}$:

$$\mathrm{h}_i^* = \tanh(\mathrm{U}^{\mathrm{x}} r_{|p_i|} + \mathrm{U}^{\mathrm{y}} h_{|p_i|}) \qquad \mathrm{h}_i^* \in \mathbb{R}^{k}, \tag{8}$$

where $\mathrm{U}^{\mathrm{x}}, \mathrm{U}^{\mathrm{y}} \in \mathbb{R}^{k \times k}$ are two trainable parameters. Similar to the above mentioned procedure for aligning opinion phrase p_i from p_j, we also align opinion phrase p_j from p_i and obtain its final representation h_j^*.

Prediction and Training. The probability distributions for the review classification ($\mathbf{s}_r$) and explanation classification ($\mathbf{s}_e$) tasks are obtained from two softmax classifiers respectively:

$$\mathbf{s}_r = \mathrm{softmax}(\mathrm{W}^{\mathrm{r}} \mathrm{h}_r^* + \mathrm{b}^{\mathrm{r}}) \qquad \mathbf{s}_r \in \mathbb{R}^{2} \tag{9}$$

$$\mathbf{s}_e = \mathrm{softmax}(\mathrm{W}^{\mathrm{e}} \mathrm{h}_e^* + \mathrm{b}^{\mathrm{e}}) \qquad \mathbf{s}_e \in \mathbb{R}^{2}, \tag{10}$$

where $\mathrm{h}_e^* = [\mathrm{h}_r^*; \mathrm{h}_i^*; \mathrm{h}_j^*]$ is the concatenation of the sentence's and opinion phrases' representations; $\mathrm{W}^{\mathrm{r}} \in \mathbb{R}^{2 \times k}, \mathrm{b}^{\mathrm{r}} \in \mathbb{R}^{2}$ and $\mathrm{W}^{\mathrm{e}} \in \mathbb{R}^{2 \times k}, \mathrm{b}^{\mathrm{e}} \in \mathbb{R}^{2}$ are the classifiers' weights and biases respectively. Finally, we define the training objective J as follows:

$$J = J_o + \lambda J_r, \tag{11}$$

where J_r and J_o are the cross-entropy loss for the first and second classification task, respectively; λ is a tunable hyper-parameter.

4 CANONICALIZING OPINION PHRASES

The goal of this component is to group duplicates or very similar opinion phrases together in order to build a concise opinion graph. We call this process *canonicalizing opinion phrases*. This is a necessary step as reviews contain a variety of linguistic variations to

express the same or similar opinions. For example, *"one block from beach"*, *"close to the pacific ocean"*, *"unbeatable beach access"*, and *"very close to the sea"* are different phrases used in hotel reviews to describe the same opinion. Other examples (e.g., *"great location"* and *"good location"*) are shown on the right of Figure 1.

A widely used method for representing a phrase is the average word embeddings of the phrase based on a pre-trained word embedding model (e.g., GloVe) [21, 37].However, a serious limitation of this approach is that it may wrongly cluster opinion phrases that share the same opinion/aspect term but are not necessarily similar. For example, the average word embeddings for the opinion phrase "*very close to the ocean*" is closer to an irrelevant opinion phrase "*very close to the trams*" than a semantically similar opinion phrase "*2 mins walk to the beach*". See Figure 8(a) for more examples.

To account for semantic similarity, we consider opinion phrase representation learning to learn opinion phrase embeddings before applying a clustering algorithm such as k-means to group similar opinion phrases together. Our method offers two major benefits. First, it utilizes only weak supervision: our model does not require any additional labels for training as it leverages the outputs of previous components, namely the aspect categories and sentiment polarity of opinion phrases (in Opinion Mining), and mined explanations (in Explanation Mining). Second, our method allows us to use existing clustering algorithms, which we improve by simply using our opinion phrase representations as features.

4.1 Opinion Phrase Representation Learning

We develop an opinion phrase representation learning framework **W**eakly-**S**upervised **O**pinion **P**hrase **E**mbeddings (WS-OPE), which has two key properties: (1) different embeddings are used for opinion and aspect terms separately, which are then merged into an opinion phrase embedding, and (2) it uses weak supervision to incorporate the semantic meaning of opinion phrases into the opinion phrase embeddings by minimizing the vector reconstruction loss, as well as additional losses based on predicted aspect category, sentiment polarity, and explanations obtained from the previous steps of ExplainIt. The first idea makes it easier for the model to learn to distinguish lexically similar but semantically different opinion phrases. For example, the model can distinguish *"very close to the trams"* and *"very close to the ocean"* based on the aspects *"tram"* and *"ocean"*, which have different representations. The second idea enables the learning of opinion phrase embeddings without additional cost. With the additional loss functions based on signals extracted in the previous steps, the model can incorporate sentiment information into opinion phrase embeddings while retaining the explanation relationship between opinion phrases in the embedding space.

Figure 5 illustrates the learning framework of WS-OPE. The model encodes an opinion phrase into an embedding vector, which is the concatenation of an opinion term embedding and an aspect term embedding. Then, the opinion phrase embedding is used as input to evaluate multiple loss functions. The total loss is used to update the model parameters, including the opinion phrase embeddings themselves (i.e., opinion and aspect embeddings). We describe each component and loss function next. Table 2 summarizes the notations, which we will use in this section.

Notation	Meaning
$P = \{p_i\}_{i=1}^{N_p}$	Input opinion phrases
$E = \{e_i\}_{i=1}^{N_e}$	Explanations extracted by Section 3
$\mathrm{v}_p = [\mathrm{v}_a; \mathrm{v}_o]$	Embeddings for opinion phrase, opinion term, aspect term
W^{a}	Trainable parameter for opinion phrase encoding
W^{R}, b^{R}	Trainable parameters for opinion phrase reconstruction
J_R	Reconstruction loss
$\mathrm{W}^{\mathtt{asp}}$, $\mathrm{b}^{\mathtt{asp}}$	Trainable parameters for aspect classification
$\mathrm{W}^{\mathtt{pol}}$, $\mathrm{b}^{\mathtt{pol}}$	Trainable parameters for polarity classification
J_{asp}, J_{pol}	Aspect and polarity classification loss
J_E	Intra-cluster explanation loss

Table 2: Notations for opinion representation learning.

Input. The input to the model is a set of N_p opinion phrases $P = \{p_i\}_{i=1}^{N_p}$ that are extracted from reviews about a single entity (e.g., a hotel), and a set of N_e explanations $E_e = \{e_i\}_{i=1}^{N_e}$ for the opinion phrases. Recall that each opinion phrase[4] p consists of two sequences of tokens (o, a) of the opinion term and the aspect term. We use `asp` and `pol` to denote the aspect category and sentiment labels of a phrase, predicted by the ABSA model during the opinion mining stage.

Opinion Phrase Encoding. Given an opinion phrase (o, a), we first use an embedding layer with the self-attention mechanism [17] to compute an aspect embedding v_a and an opinion embedding v_o respectively[5]. The aspect embedding v_a is obtained by attending over the aspect term tokens $a = (w_1, \ldots, w_n)$:

$$u_i = \mathrm{v}_{w_i}^T \mathrm{W}^{\mathrm{a}} \mathrm{v}'_a \qquad u_i \in \mathbb{R} \tag{12}$$

$$c_i = \frac{\exp(u_i)}{\sum_{j=1}^{m} \exp(u_j)} \qquad c_i \in \mathbb{R} \tag{13}$$

$$\mathrm{v}_a = \sum_{i=1}^{m} c_i \mathrm{v}_{w_i} \qquad \mathrm{v}_a \in \mathbb{R}^d, \tag{14}$$

where $\mathrm{v}_{w_i} \in \mathbb{R}^d$ is the output of the embedding layer for word w_i, $\mathrm{v}'_a \in \mathbb{R}^d$ is the average word embedding for words in a, and $\mathrm{W}^{\mathrm{a}} \in \mathbb{R}^{d \times d}$ is a trainable parameter used to calculate attention weights. We encode the opinion term o into v_o in the same manner. Then, we concatenate the two embedding vectors into a single opinion phrase embedding v_p:

$$\mathrm{v}_p = [\mathrm{v}_a; \mathrm{v}_o] \qquad \mathrm{v}_p \in \mathbb{R}^{2d}. \tag{15}$$

Reconstruction loss. As in the standard auto-encoder paradigm, the main idea behind the reconstruction loss is to learn input vectors v_p so that they can be easily reconstructed from a representative matrix R. Following previous studies [1, 17], we set the K rows of $\mathrm{R} \in \mathbb{R}^{K \times 2d}$ using a clustering algorithm (e.g., k-means) over initial opinion phrase embeddings, such that every row corresponds to a cluster centroid[6]. To reconstruct the phrase vector v_p, we first feed

[4] We omit the index i of the opinion phrase and explanation below since it is clear from the context.

[5] The embedding architecture is similar to that of Attention-Based Auto-Encoder (ABAE) [17] but our model has different encoders for aspect and opinion terms, whereas ABAE does not distinguish aspect and opinion terms and directly encodes an opinion phrase into an embedding vector.

[6] We freeze R during training after initialization to facilitate training stability, as suggested in [1].

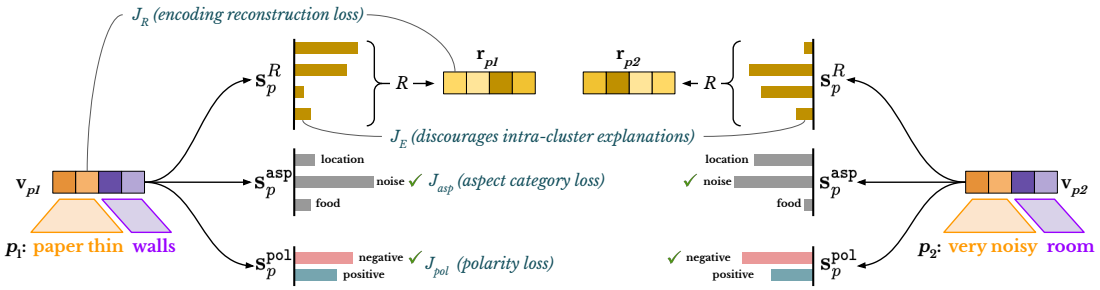


Figure 5: Overview and loss functions of our opinion phrase representation learning framework (WS-OPE).

it to a softmax classifier to obtain a probability distribution over the K rows of R:

$$\mathbf{s}_p^R = \text{softmax}(\mathrm{W}^{\mathrm{R}}\mathrm{v}_p + \mathrm{b}^{\mathrm{R}}) \qquad \mathbf{s}_p^R \in \mathbb{R}^K, \tag{16}$$

where $\mathrm{W}^{\mathrm{R}} \in \mathbb{R}^{K\times 2d}, \mathrm{b}^{\mathrm{R}} \in \mathbb{R}^K$ are the weight and bias parameters of the classifier respectively. We get the *reconstructed* vector r_p for opinion phrase p as follows:

$$\mathrm{r}_p = \mathrm{R}^T \mathbf{s}_p^R \qquad \mathrm{r}_p \in \mathbb{R}^{2d}. \tag{17}$$

We use the triplet margin loss [3] as the cost function, which moves the input opinion phrase v_p closer to the reconstruction r_p, and further away from k_n randomly sampled negative examples:

$$J_R = \sum_{p\in P}\sum_{i=1}^{k_n} \max(0, 1 - \mathrm{r}_p\mathrm{v}_p + \mathrm{r}_p\mathrm{v}_{n_i}), \tag{18}$$

where $n_i \in P$ are randomly selected negative examples. The sampling procedure tries to sample opinion phrases that are not similar to the input opinion phrase with respect to the probability distribution of Eq. (16). For an opinion phrase p, the probability of another opinion phrase p' being selected as a pseudo negative example is *inversely* proportional to the cosine similarity between $\mathbf{s}_p^R$ and $\mathbf{s}_{p'}^R$.

Aspect category and polarity loss. We also leverage additional signals that we collected from the previous steps to obtain better representations. For example, we would like to avoid opinion phrases such as "*friendly staff*" and "*unfriendly staff*" from being close in the embedding space. Hence, we incorporate sentiment polarity and aspect category into the framework to learn better opinion phrase embeddings with respect to sentiment information.

In WS-OPE, we add two classification objectives to learn the parameters. Specifically, we feed an opinion phrase embedding v_p into two softmax classifiers to predict the probability distributions of the aspect category and the sentiment polarity respectively:

$$\mathbf{s}_p^{\mathsf{asp}} = \text{softmax}(\mathrm{W}^{\mathsf{asp}}\mathrm{v}_p + \mathrm{b}^{\mathsf{asp}}) \tag{19}$$

$$\mathbf{s}_p^{\mathsf{pol}} = \text{softmax}(\mathrm{W}^{\mathsf{pol}}\mathrm{v}_p + \mathrm{b}^{\mathsf{pol}}). \tag{20}$$

The distributions $\mathbf{s}_p^{\mathsf{asp}}$ and $\mathbf{s}_p^{\mathsf{pol}}$ are used to compute cross-entropy losses J_{asp} and J_{pol} against *silver-standard* aspect and sentiment labels, predicted for each extracted phrase during opinion mining.

Intra-cluster explanation loss. Our mined explanations should also provide additional signals to learn better embeddings. Essentially, if an opinion phrase p_i explains an opinion phrase p_j, they should belong to different clusters (i.e., they should not belong to the same cluster). To reduce intra-cluster explanations, we define the *intra-cluster explanation loss* by the Kullback-Leibler divergence (KL) between the probability distributions $\mathbf{s}_{p_i}^R$ and $\mathbf{s}_{p_j}^R$:

$$J_E = -\sum_{e\in E} \mathrm{KL}(\mathbf{s}_{p_i}^R, \mathbf{s}_{p_j}^R),\ e = (p_i \rightarrow p_j), \tag{21}$$

where E is a set of pairs of opinion phrases in the mined explanations, and $\mathrm{KL}(\cdot,\cdot)$ is the KL divergence between two distributions.

When opinion phrases p_i and p_j are in an explanation relationship, we would like to penalize the case where $\mathrm{KL}(\mathbf{s}_{p_i}^R, \mathbf{s}_{p_j}^R)$ is small (likely to be in the same cluster). As a result, we are able to push the embeddings of p_i and p_j of opinion phrases that have an explanation relationship apart from each other to discourage having intra-cluster explanations.

Training objective. We define the final loss function by combining the four loss functions defined above:

$$J_{\text{WS-OPE}} = J_R + \lambda_{\mathsf{asp}}J_{\mathsf{asp}} + \lambda_{\mathsf{pol}}J_{\mathsf{pol}} + \lambda_E J_E, \tag{22}$$

where λ_{asp}, λ_{pol}, and λ_E are three hyper-parameters to control the influence of each corresponding loss. In practice, we prepare two types of mini-batches; one for single opinion phrases and one for explanation pairs. For each training step, we create and use these mini-batches separately: we use the single phrase mini-batch to evaluate the reconstruction, aspect category, and polarity losses; we use the explanation mini-batch to evaluate the explanation loss. At the end of every training step, we accumulate the loss values following Eq. (22) and update the model parameters.

4.2 Clustering Opinion Phrases

After the opinion phrase representation learning, we apply a clustering algorithm over the learned opinion phrase embeddings to obtain opinion clusters. Each opinion cluster is a node (i.e., *canonicalized* opinion) of the final opinion graph. Note that our opinion canonicalization module is not tied to any specific clustering algorithm. We will show in Section 6, our two-stage method for generating opinion clusters performs well regardless of the choice of clustering algorithms, which also demonstrates the strength of opinion phrase representation learning.

We could consider directly using a score distribution $\mathbf{s}_p^R$ for clustering instead of applying a clustering algorithm to the learned opinion embeddings. However, we found that $\mathbf{s}_p^R$ does not perform well compared to our approach. This is expected, as the classifier responsible for producing $\mathbf{s}_p^R$ has only been trained via the reconstruction loss, whereas the phrase embeddings have used all of the four signals, thus producing much richer representations. We

Group	Models	Acc.
RTE	Two-way attention	74.78
	Decomposable attention	76.26
	RTE-BERT	79.75
RelCLS	Sent-BiLSTM	75.41
	Sent-BERT	81.79
Proposed	MaskedDualAttn-GloVe	82.20
	MaskedDualAttn-BERT	**86.23**
Ablated	MaskedDualAttn-GloVe; single-task	78.67 (3.53 ↓)
	MaskedDualAttn-BERT; single-task	80.57 (5.66 ↓)

Table 3: Explanation mining accuracy of different models. Our ablated single-task model (GloVe and BERT) are trained without the review classification objective (i.e., $\lambda = 0$).

conduct further analysis on the contributions of the multiple loss functions in 6.2.5.

5 GENERATING OPINION GRAPHS

Based on mined explanations and canonicalized opinions from Sections 3 and 4, the final step for generating an opinion graph is to predict edges between nodes. In theory, when using perfectly accurate explanations and opinion clusters, generating such edges is trivial. Intuitively, when an opinion phrase explains another opinion phrase, opinion phrases that are paraphrases of the first phrase should also explain phrases that paraphrase the latter one. In other words, given a set of explanations E and two groups of opinion phrases, n_i and n_j, there should be an edge from n_i to n_j if there exists an edge between two opinion phrases in n_i and n_j respectively:

$$e = (n_i \rightarrow n_j) \text{ is true, if } \exists\, e = (p \rightarrow p') \in E | p \in n_i, p' \in n_j$$

For example, when we know that *"close to the beach"* → *"good location"*, we are able to conclude (*"close to the beach"*, *"near the beach"*, *"walking distance to the beach"*) → (*"good location"*, *"great location"*, *"awesome location"*).

However, in practice, our obtained explanations and nodes are not perfect. Thus, we may get a lot of false positive edges based on the above criteria. To minimize the false positives, we use a simple heuristic to further prune the edges, which is based on the observation that two groups of opinions seldom explain each other at the same time.

$$e = (n_i \rightarrow n_j) \text{ is true, if } \sum_{e \in E_{ij}} \mathrm{p}_e - \sum_{e' \in E_{ji}} \mathrm{p}'_e > 0\,,$$

where $E_{ij} = \{e = (p_i \rightarrow p_j) \mid p_i \in n_i, p_j \in n_j\}$ and $E_{ji} = \{e = (p_j \rightarrow p_i) \mid p_i \in n_i, p_j \in n_j\}$ are the explanations from n_i to n_j and n_j to n_i respectively; and p_e and p'_e are the explanation probabilities obtained from our explanation mining classifier.

Deriving edges between canonicalized opinions is a difficult problem in general. There are many ways to optimize this step further, and we leave this as part of our future work.

6 EVALUATION

We evaluate ExplainIt with three types of experiments. We use two review datasets for evaluation: a public Yelp corpus of 642K restaurant reviews and a private Hotel corpus[7] of 688K hotel reviews. For the mining explanations and canonicalizing opinion phrases, we perform automatic evaluation over crowdsourced gold labels[8]. To evaluate the quality of the generated opinion graph, we conducted a user study.

6.1 Mining Explanations

6.1.1 Dataset and metric. Based on the data collection process we described in 3.2, we used a dataset with 7.4K balanced examples in Hotel domain. We further split the labeled data into training, validation, and test sets with ratios of $(0.8, 0.1, 0.1)$. We evaluate the models by their prediction accuracy.

6.1.2 Methods. We compare our explanation classifier model and baseline methods, which we categorized into three groups. The first group (RTE) consists of three different models for RTE, the second group (RelCLS) consists of two models for relation classification, the third group (Proposed) consists of different configurations of our model, and the last group (Ablated) is for ablation study. We trained all the models on the same training data with Adam optimizer [20] (learning-rate=$1e-3$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and decay factor of 0.01) for 30 epochs. All models except BERT used the same word embedding model (`glove.6B.300d`) [31] for the embedding layers. For the RTE group, the input to the models is a pair of opinion phrases. The review context information associated with the pairs is ignored.

Two-way attention: The two-way attention model [35] is a BiLSTM model with a two-way word-by-word attention, which is used in our proposed model. This can be considered a degraded version of our proposed model only with the alignment attention, which takes opinion phrases without context information.

Decomposable attention: The decomposable attention model [30] is a widely used and the best non-pre-trained model for RTE tasks.

RTE-BERT: BERT [11] is a pre-trained self-attention model, which is known to achieve state-of-the-art performance in many NLP tasks. We fine-tuned the $\text{BERT}_{\text{base}}$ model with our training data.

For the RelCLS group, we follow existing relation classification techniques [23, 40, 43] and formulate the explanation classification problem as a single sentence classification task. We "highlight" opinion phrases in a review with special position indicators [43] `[OP1]` and `[OP2]`. For example, "`[OP1]` *Good location* `[OP1]` *with* `[OP2]` *easy access to beach* `[OP2]`" highlights two opinion phrases: "*good location*" and "*easy access to beach*". With this input format, we can train a sentence classification model that takes into account context information while it recognizes which are opinion phrases.

Sent-BiLSTM: We trained a BiLSTM model with self-attention [24], which was originally developed for sentence classification tasks. The model architecture can be considered a degraded version of our model without the two-way word-by-word attention. Because we use special position indicators, this model classifies whether the opinion phrases are in the explanation relationship or not.

[7] Data was collected from multiple hotel booking websites.
[8] We release the labeled datasets at https://github.com/megagonlabs/explainit.

		Homogeneity (Precision)			Completeness (Recall)			V-measure (F1)		
		k-means	GMM	Cor. Cluster.	k-means	GMM	Cor. Cluster.	k-means	GMM	Cor. Cluster.
Hotel	AvgWE	0.6695	0.6785	0.7240	0.7577	0.7728	0.6756	0.7102	0.7219	0.6985
	ABAE	0.6626	0.6628	0.6964	0.7609	0.7522	0.7113	0.7075	0.7039	0.6966
	WS-OPE (ours)	**0.7073**	**0.7177**	**0.7460**	**0.8115**	**0.8184**	**0.8370**	**0.7551**	**0.7641**	**0.7848**
Restaurant	AvgWE	0.5854	0.5509	0.5851	0.8168	0.7801	0.8103	0.6778	0.6413	0.6761
	ABAE	0.5563	0.5553	**0.6256**	0.7927	0.7779	0.7819	0.6492	0.6432	0.6918
	WS-OPE (ours)	**0.5920**	**0.5572**	0.6158	**0.8333**	**0.8111**	**0.8155**	**0.6877**	**0.6555**	**0.6985**

Table 4: Opinion phrase canonicalization performance on Hotel and Restaurant datasets.

Sent-BERT: We fine-tuned the $\text{BERT}_{\text{base}}$ model for the sentence classification task with the training data. Different from RTE-BERT, Sent-BERT takes an entire review, enriched by opinion phrase markers, as the input so it can take context information into account.

The last group (Proposed) include two variations of our model:
MaskedDualAttn-GloVe: The default model with an embedding layer initialized with the GloVe (`glove.6B.300d`) model.
MaskedDualAttn-BERT: We replace the embedding layer with the $\text{BERT}_{\text{base}}$ model to obtain contextualized word embeddings.

6.1.3 Result analysis. As shown in Table 3, our proposed model achieves significant improvement over baseline approaches: we largely outperform non-pre-trained textual entailment models and sentence classification models by 5.94% to 7.42%. Furthermore, to mine explanations, models that consider context information tend to perform better. We found that BERT over sentences is 2% more accurate than BERT over opinion phrases only. Lastly, leveraging pre-trained model can further improve the performance: by replacing the embedding layer with BERT, the accuracy is further improved by 4%.

We also conducted an ablation analysis to verify our multi-task learning framework. We tested variants of MaskedDualAttn-GloVe and MaskedDualAttn-BERT that were trained without the review classification objective (i.e., $\lambda = 0$). The other configurations were the same as MaskedDualAttn-GloVe and MaskedDualAttn-BERT. The results are shown in Table 3 (Ablated). From the results, we confirm that the multi-task learning significantly contributes to the performance of both of the MaskedDualAttn models.

Since our model has both of the alignment attention and self-attention, only with the single objective function (i.e., explanation classification), the model may not be optimized well. In fact, by turning off the multi-task learning, we observe lower performance by MaskedDualAttn-BERT than Sent-BERT, while MaskedDualAttn-GloVe shows better performance than the BiLSTM-based baseline models (Sent-BiLSTM). Therefore, we consider the issue can be resolved by incorporating multiple objectives as our final models, regardless of the choice of the base model (i.e., GloVe, BERT) achieves the best performance in the explanation mining task.

6.2 Canonicalizing Opinions Phrases

6.2.1 Dataset and metrics. For both Hotel and Restaurant domains, we first exclude entities with too few/many reviews and randomly select 10 entities from the remaining ones. We also develop a non-trivial process to collect the gold clusters using crowdsourcing.

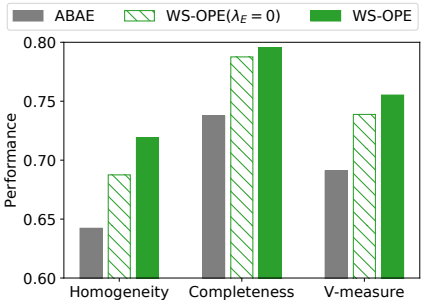


Figure 6: Ablation study on the usefulness of the intra-cluster explanation loss. Excluding the intra-cluster explanation loss (i.e., $\lambda_E = 0$) hurts the opinion phrase canonicalization performance, while it still performs better than ABAE.

We evaluate the performance with three metrics: (1) homogeneity, (2) completeness, and (3) V-measure, in the same manner as precision, recall, and F1-score. Homogeneity measures the *precision* of each cluster and scores 1.0 if each cluster contains only members of a single class. Completeness measures the *recall* of each true class and scores 1.0 if all members of a given class are assigned to the same cluster. The V-measure is the harmonic mean between homogeneity and completeness scores.

6.2.2 Methods. To understand the benefits of our learned opinion phrase embeddings, we evaluate whether they can consistently improve the performance of existing clustering algorithms. Here we select three representative clustering algorithms, k-means, Gaussian Mixture Models (GMM) [6], and Correlation Clustering over similarity score [4, 13]. For k-means and GMM, we set $k = 50$ and $k = 20$ for Hotel and Restaurant datasets, respectively; we set $\theta = 0.85$ for Correlation Clustering for both datasets. We compared the following methods, which use the same word embedding model (`glove.6B.300d`):
AvgWE: We first calculate the average word embeddings for opinion term and aspect term using GloVe, and then concatenate the aggregated average embedding as the final opinion phrase embedding.
ABAE: We fine-tune the word embedding model without additional labels (i.e., $\lambda_{\text{asp}} = \lambda_{\text{pol}} = \lambda_E = 0$), which can be considered an ABAE model [17].
WS-OPE: We learn opinion phrase embeddings based on GloVe word embeddings with weak-supervision from aspect category,

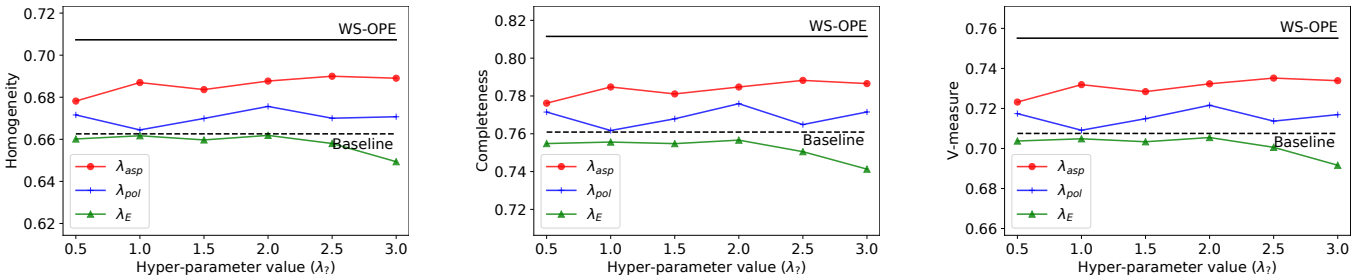


Figure 7: Sensitivity analysis on hyper-parameters λ_{asp}, λ_{pol}, and λ_E. The solid line is the performance of WS-OPE and the dotted line is that of the best-performing baseline model (from Table 4).

polarity, and explanation with the following hyper-parameters: $\lambda_{asp} = 1.0, \lambda_{pol} = 0.5, \lambda_E = 0.2$. We obtained the explanations by our explanation classifier (Section 3)[9].

6.2.3 Result analysis. As shown in Table 4, our learned opinion phrase representations (WS-OPE) achieve the best performance among all settings and consistently boost the performance of existing clustering algorithms compared to the baseline methods in both HOTEL and RESTAURANT domains. In addition, we confirm that our model significantly benefits from the additional weak supervision from opinion and explanation mining as our method significantly improves the performance compared to ABAE, which does not use the weak supervision.

6.2.4 Usefulness of mined explanations. To verify if the explanations mined in the previous step contribute to the performance of opinion phrase canonicalization, we conducted an ablation study. We evaluated our method without the intra-cluster loss (i.e., $\lambda_E = 0$), so the learned opinion phrase representations do not consider any explanation relationships between opinion phrases. Figure 6 shows that the performance of our model degrades without the intra-cluster loss (i.e., mined explanations) but is still significantly better than the baseline ABAE model. The results also confirm that the intra-cluster loss based on mined explanations can boost the performance of opinion phrase canonicalization.

6.2.5 Hyper-parameter sensitivity. We also conducted the sensitivity analysis on the hyper-parameters λ_{asp}, λ_{pol}, and λ_E, which balance the multiple loss functions in Eq. (22), to evaluate the robustness of our model with respect to those hyper-parameters. Specifically, we evaluated our model with different $\lambda_?$ ($? = \{asp, pol, E\}$) $\in \{0.5, 1.0, 1.5, 2.0, 2.5, 3.0\}$ while fixing the other two hyper-parameters as 0. Thus, we can test the contribution of each loss function with different weights when combined with the base reconstruction loss.

Figure 7 shows the results for three evaluation metrics (Homogeneity, Completeness, and V-measure). From the results, we confirm that by using either the aspect category loss or the polarity loss, our model consistently outperforms the baseline models. Although only using the intra-cluster loss ($\lambda_E > 0$) does not outperform the baseline, we have shown the usefulness of the intra-cluster loss when combined with the other loss functions in 6.2.4.

6.2.6 Embedding space visualization. We also present a qualitative analysis of how our WS-OPE helps to canonicalize opinion phrases. Figure 8 shows a two-dimensional t-SNE projection [27] of embeddings for a fraction of the opinion phrases about a hotel. The opinion phrase embeddings obtained before and after representation learning are shown on the left and right side of the figure, respectively. The color codes denote true cluster assignments.

We observe that the original vectors appear more uniformly dispersed in the embedding space, and hence, the cluster boundaries are less prominent. Additionally, we annotated the figure with a number of particularly problematic cases. For example, each of the following pairs of opinion phrases (*"very close to tram"*, *"very close to ocean"*), (*"2 mins walk to beach"*, *"2 mins walk to zoo"*), and (*"good location near the ocean"*, *"good location near the zoo"*), appear in close proximity when they should belong to different clusters. After learning the representations, the problematic pairs of vectors are now clearly separated in their respective clusters.

6.3 Opinion Graph Quality: User Study

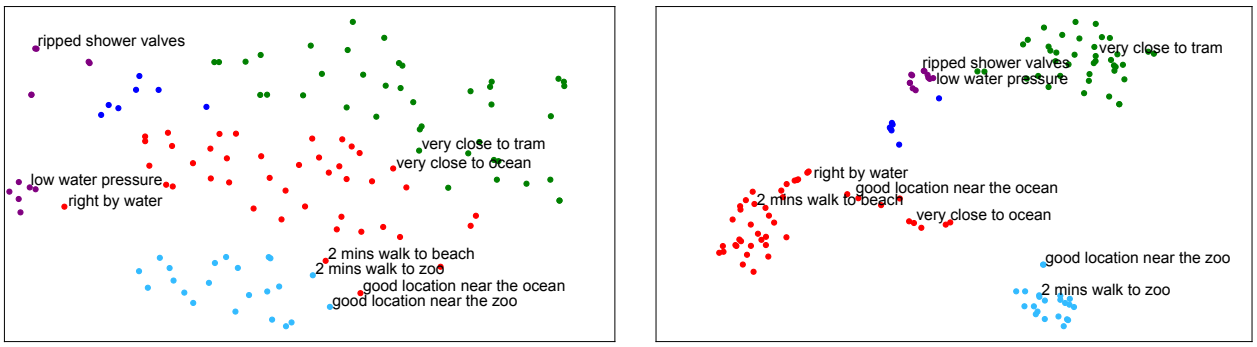
In addition to the automatic evaluation of the explanation mining and opinion phrase canonicalizing modules, we designed a user study to verify the quality of the final opinion graphs produced by EXPLAINIT. Assessing the quality of an entire opinion graph at once is impractical due to its size and complexity. Instead, we broke down the evaluation of each generated graph into a series of pairwise tests, where human judges were asked to verify the explanation relation (or lack thereof) between pairs of nodes in the graph.

More specifically, given a predicted graph $G = (N, E)$ about an entity, we sampled node pairs (n_i, n_j), so that we get a balanced number of pairs for which we predicted the existence or absence of an explanation relation. For every pair, we present the two nodes to the user and show five member opinion phrases from each one. We further show the predicted relation between the nodes (*"explains"* or *"does not explain"*) and ask the users if they agree with it. An example is shown in Figure 9.

We generated examples for 10 hotels (i.e., their constructed opinion graphs), amounting to 166 node pairs (or questions) in total. This user study was done via Appen's highest accuracy (Level-3) contributors, who obtained no less than 80% accuracy on our test questions. Every question was shown to 3 judges and we obtained a final judgment for it using a majority vote[10]. The judges *agreed* with our predicted relation in 77.1% of cases.

[9] We use the same trained model for both HOTEL and RESTAURANT.

[10] The inter–annotator agreement between contributors is 85.6%.



(a) Opinion phrase embeddings of AvgWE (i.e., before representation learning)

(b) Opinion phrase embeddings of WS-OPE (i.e., after representation learning)

Figure 8: Embedding space comparison. Color-coding denotes true cluster assignments. After learning representations, semantically similar opinion phrases are significantly closer to each other and irrelevant ones are further apart in the embedding space, allowing easier opinion phrase canonicalization

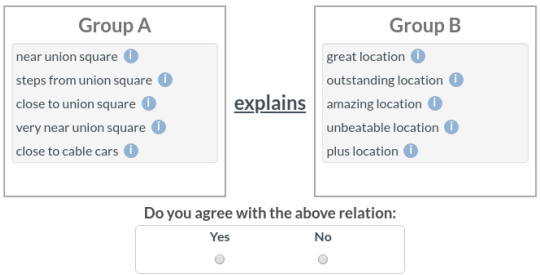


Figure 9: Example question for user study. Reviews can be seen by hovering over the (i) icon for each opinion phrase.

6.4 Opinion Graph Usefulness

To evaluate the usefulness of the generated summaries, we further presented the predicted clusters and explanation relations to crowd workers and let them judge the usefulness of such produced information. Note that our clusters of near-synonym opinions and the explanation relations are largely absent from existing travel/service booking websites. We again used Appen's highest accuracy (Level-3) contributors for this task. We presented 164 clusters-explanation pairs to 1006 workers and observed that 80.12% of workers found our clusters and explanations useful.

Furthermore, based on ExplainIt, we built a summary explorer that visualizes the generated opinion graphs [38]. By leveraging the provenance of extracted opinion phrases and explanations, this summary explorer also allows users to access the *"evidence"* (original reviews) of the nodes/edges, thus making ExplainIt more transparent and interpretable to end-users. Lastly, with the help of an additional abstractive summarization system [36], ExplainIt can further generate textual summaries.

7 RELATED WORK

Opinion Mining: There has been work of mining opinions from online reviews since [18, 19, 25, 32–34, 41]. Those studies developed opinion extraction systems using association mining techniques to extract frequent noun phrases from reviews and then aggregates sentiment polarity scores. These form the aspect-based opinions of online product reviews. Opinion Observer [26] extended the method to build a system that visualizes the polarity information of each aspect with bar plots. ExplainIt goes one step further from existing opinion mining techniques. Based on extracted opinions, it organizes opinions into an opinion graph such that it includes (a) explanation relationships between opinions and (b) canonicalizes opinions to exclude redundancy.

Explanation Classifier: Recognizing Textual Entailment (RTE) [10] and Natural Language Inference (NLI) [8] are the tasks to judge if given two statements, whether one statement can be inferred from the other. These tasks are usually formulated as a sentence-pair classification problem where the input is two sentences. A major difference between RTE models and our explanation classifier is that our classifier judges if an opinion phrase explains another opinion phrase in the same review text. The two opinion phrases may appear in the same sentence or may appear in different sentences. Hence, as described in Section 3, we added another task, i.e., explanation existence judgment, in addition to the opinion-phrase classification task to improve the performance using a multi-task learning framework. The goal of relation classification [23, 40, 43] is to classify the relationship between a pair of entities. For example, determining the relation between entity "Ms. Ruhl" and entity "Chicago" given a context sequence "Ms. Ruhl, 32, grew up in suburban Chicago.". Similar to the explanation mining problem, the input of relation classification also includes both the context sequence and a pair of entities. However, different from MaskedDualAttn, existing relation classification models do not incorporate word-by-word alignments between entities. This is because for relation classification, such alignment (e.g., alignment between "Ms. Ruhl" and "Chicago") is not very useful compared to the context sequence that connects the given pair of entities (e.g., "grew up in suburban").

Opinion Phrase Canonicalization: Aspect-based auto-encoder [17] is an unsupervised neural model that clusters sentences while learning better word embeddings for sentiment analysis. It showed better performance than conventional topic models (e.g., LDA [7], Biterm

Topic Model [42]) in aspect identification tasks. Our WS-OPE extends their approach by (1) having an opinion phrase encoder that consists of two encoders for aspect and opinion terms, and (2) incorporating additional sentiment signals such as sentiment polarity and aspect category in a weakly-supervised manner.

Opinion phrase canonicalization is closely related to KB canonicalization [16, 37], which canonicalizes entities or relations (or both) by merging triples consisting of two entities and a relation, based on the similarity. Galárraga et al. [16] proposed several manually-crafted features[11] for clustering triples. CESI [37] uses side information (e.g., entity linking, WordNet) to train better embedding representations for KB canonicalization. The difference from KB canonicalization is that ExplainIt does not rely on external structured knowledge such as WordNet or KBs, which were used for those models. This is mainly because it is not straightforward to construct a single KB that reflects a wide variety of subjective opinions written in reviews. Instead, we aim to construct an opinion graph for each entity.

8 CONCLUSION

We present ExplainIt, a system that extracts opinions and constructs an explainable opinion graph from reviews.

ExplainIt consists of four components including a novel explanation mining component and an opinion phrase canonicalization component, which we developed to construct opinion graphs from online reviews. Our experimental results show that our methods significantly perform better than baseline methods in both the task of classifying explanations and canonicalizing opinion phrases by up to 5.4% and, respectively, 12.2%. In addition, our user study confirmed that human judges agree with the explanation relationships depicted in our opinion graph in more than 77% of the cases. We created labeled datasets for explanation mining and opinion phrase canonicalization tasks and we made these datasets publicly available for future research.

REFERENCES

[1] Stefanos Angelidis and Mirella Lapata. 2018. Summarizing Opinions: Aspect Extraction Meets Sentiment Prediction and They Are Both Weakly Supervised. In *EMNLP*. 3675–3686.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. ICLR '15*.

[3] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. 2016. Learning local feature descriptors with triplets and shallow convolutional neural networks.. In *Bmvc*, Vol. 1. 3.

[4] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. 2004. Correlation Clustering. *Mach. Learn.* 1–3 (2004), 89–113.

[5] Nikita Bhutani, Aaron Traylor, Chen Chen, Xiaolan Wang, Behzad Golshan, and Wang-Chiew Tan. 2020. SAMPO: Unsupervised Knowledge Base Construction for Opinions and Implications. In *Automated Knowledge Base Construction*.

[6] Christopher M Bishop. 2006. *Pattern recognition and machine learning*. Springer.

[7] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* 3, null (March 2003), 993–1022.

[8] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

[9] Jiaoyan Chen, Ernesto Jiménez-Ruiz, and Ian Horrocks. 2019. Canonicalizing Knowledge Base Literals. In *Proc. ISWC '19*. 110–127.

[10] Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges Workshop*. Springer, 177–190.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL HLT*. 4171–4186.

[12] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *SIGKDD*. 601–610.

[13] Micha Elsner and Warren Schudy. 2009. Bounding and Comparing Methods for Correlation Clustering Beyond ILP. In *NAACL HLT*. 19–27.

[14] Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *EMNLP*. 1535–1545.

[15] Joseph L Fleiss, Bruce Levin, and Myunghee Cho Paik. 2013. *Statistical methods for rates and proportions*. john wiley & sons.

[16] Luis Galárraga, Geremy Heitz, Kevin Murphy, and Fabian M. Suchanek. 2014. Canonicalizing Open Knowledge Bases. In *Proc. CIKM '14*. 1679–1688.

[17] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An Unsupervised Neural Attention Model for Aspect Extraction. In *ACL*. 388–397.

[18] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *SIGKDD*. 168–177.

[19] Minqing Hu and Bing Liu. 2004. Mining Opinion Features in Customer Reviews. In *AAAI*. 755–760.

[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[21] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *ICML*. 957–966.

[22] Yuliang Li, Aaron Feng, Jinfeng Li, Saran Mumick, Alon Halevy, Vivian Li, and Wang-Chiew Tan. 2019. Subjective Databases. *Proc. VLDB Endow.* 12, 11 (2019), 1330–1343.

[23] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *ACL*. 2124–2133.

[24] Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A Structured Self-attentive Sentence Embedding. In *Proc. ICLR '17*.

[25] Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool.

[26] Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion Observer: Analyzing and Comparing Opinions on the Web. In *WWW*. 342–351.

[27] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[28] Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. 2018. Never-ending learning. *Commun. ACM* 61, 5 (2018), 103–115.

[29] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2015. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2015), 11–33.

[30] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A Decomposable Attention Model for Natural Language Inference. In *EMNLP*. 2249–2255.

[31] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*. 1532–1543.

[32] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, AL-Smadi Mohammad, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proc. SemEval '16*. 19–30.

[33] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proc. SemEval '15*. 486–495.

[34] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion Word Expansion and Target Extraction through Double Propagation. *COLING* 37, 1 (2011), 9–27.

[35] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *Proc. ICLR '16*.

[36] Yoshihiko Suhara, Xiaolan Wang, Stefanos Angelidis, and Wang-Chiew Tan. 2020. OpinionDigest: A Simple Framework for Opinion Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 5789–5798. https://doi.org/10.18653/v1/2020.acl-main.513

[37] Shikhar Vashishth, Prince Jain, and Partha Talukdar. 2018. CESI: Canonicalizing Open Knowledge Bases Using Embeddings and Side Information. In *WWW*. 1317–1327.

[38] Xiaolan Wang, Yoshihiko Suhara, Natalie Nuno, Yuliang Li, Jinfeng Li, Nofar Carmeli, Stefanos Angelidis, Eser Kandogann, and Wang-Chiew Tan. 2020. ExtremeReader: An interactive explorer for customizable and explainable review summarization. In *WWW*. 176–180.

[39] Gerhard Weikum and Martin Theobald. 2010. From information to knowledge: harvesting entities and relationships from web sources. In *PODS*. 65–76.

[40] Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *CIKM*. 2361–2364.

[41] Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2019. BERT Post-Training for Review Reading Comprehension and Aspect-based Sentiment Analysis. In *NAACL HLT*.

[11] [37] showed that word-embedding features using GloVe outperformed the methods in [16]. Thus, we consider GloVe word embeddings as a baseline for the opinion phrase canonicalization task.

2324–2335.
[42] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2013. A Biterm Topic Model for Short Texts. In *WWW*. 1445–1456.
[43] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *ACL*. 207–212.