Joint Aspect-Sentiment Analysis with Minimal User Guidance

Honglei Zhuang, Fang Guo, Chao Zhang*, Liyuan Liu, Jiawei Han
University of Illinois at Urbana-Champaign, *Georgia Institute of Technology
{hzhuang3,fangguo1,ll2,hanj}@illinois.edu,chaozhang@gatech.edu

ABSTRACT

Aspect-based sentiment analysis is a substantial step towards text understanding which benefits numerous applications. Since most existing algorithms require a large amount of labeled data or substantial external language resources, applying them on a new domain or a new language is usually expensive and time-consuming. We aim to build an aspect-based sentiment analysis model from an *unlabeled* corpus with *minimal guidance* from users, *i.e.*, only a small set of seed words for each aspect class and each sentiment class. We employ an autoencoder structure with attention to learn two dictionary matrices for aspect and sentiment respectively where each row of the dictionary serves as an embedding vector for an aspect or a sentiment class. We propose to utilize the user-given seed words to regularize the dictionary learning. In addition, we improve the model by joining the aspect and sentiment encoder in the reconstruction of sentiment in sentences. The joint structure enables sentiment embeddings in the dictionary to be tuned towards the aspect-specific sentiment words for each aspect, which benefits the classification performance. We conduct experiments on two real data sets to verify the effectiveness of our models.

CCS CONCEPTS

• **Information systems** → **Sentiment analysis**.

KEYWORDS

Aspect-based sentiment analysis; weakly-supervised; autoencoder

ACM Reference Format:
Honglei Zhuang, Fang Guo, Chao Zhang, Liyuan Liu, Jiawei Han. 2020. Joint Aspect-Sentiment Analysis with Minimal User Guidance. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401179

1 INTRODUCTION

Sentiment analysis, which aims to identify the subjective opinion (*e.g.* positive vs. negative) of a given piece of text, is an essential task towards text understanding with a broad range of applications, including recommendations [3, 4] and stock prediction [28, 35]. Aspect-based sentiment analysis [34] takes a further step to identify the target aspect of sentiment in a given sentence. For example, sentences from restaurant reviews "*The food is good*" and "*The servers are friendly*" both convey a positive sentiment, while the first sentence should be identified as a comment on the *Food* aspect and the second on the *Service* aspect.

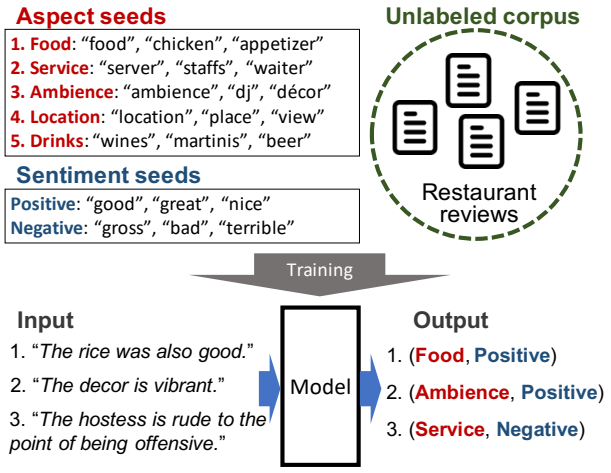


Figure 1: An example of aspect-sentiment analysis with minimal guidance. The user only provides small sets of aspect and sentiment words. The algorithm outputs the identified aspect and sentiment class for each sentence.

Most existing studies on aspect-based sentiment analysis adopt a supervised framework [7, 10, 15, 18, 22, 36, 40, 41, 45], where a significant number of labeled sentences are required to train the model. Nevertheless, such labels are extremely expensive and difficult to obtain, especially for a new domain or a new language. Another thread of studies [2, 9, 12, 19, 21, 27, 31, 32, 44] focus on weak supervision or distant supervision to perform aspect-based sentiment analysis. However, some of them either rely on external language resource such as thesaurus information [2, 9, 12, 21, 25, 27, 31] or syntactic structures generated by well-trained NLP tools [19, 32]. In reality, such information is not always available or accurate in new domains or low-resource languages.

To alleviate the reliance on external language resource and massive annotations, we develop an aspect-based sentiment analysis model which requires only a few user-defined seed words and an unlabeled text collection. More specifically, users only need to provide a small set of seed words for each aspect class and each sentiment class. The objective is to build a model to identify the aspect class and sentiment class for any sentence expressing an opinion on an aspect class. Comparing to most previous studies on weakly supervised aspect-based sentiment analysis, our setting requires significantly less effort or resource from users.

To better convey our intuition, we use Figure 1 as an example. Suppose a restaurant owner without machine learning knowledge needs to conduct aspect-based sentiment analysis on reviews of

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07.
https://doi.org/10.1145/3397271.3401179

her restaurants. The user is fully aware of the aspect classes in the restaurant domain, which are *Food*, *Service*, *Ambience*, *Location* and *Drinks*. The user only needs to provide a small set of seed words for each aspect class and each sentiment class as well as the unlabeled corpus. For example, the user can specify {"food", "chicken", "appetizer"} for *Food* aspect, {"server", "staffs", "waiter"} for *Service* aspect *etc.*, while {"good", "great", "nice"} and {"gross", "bad", "terrible"} for *Positive* and *Negative* sentiment class respectively. Based on the user-provided seed words and the corpus, our goal is to train a model to identify the aspect and sentiment class for a given sentence. For example, given a sentence "The rice was also good", the model should output its aspect class as *Food* and sentiment class as *Positive*.

There are several research challenges in this problem. The first is how to model the aspect and sentiment perspectives of sentences in the unlabeled corpus. The second is how to utilize the user-provided seed words to guide the aforementioned modeling process, such that the learned model can be well aligned with user intention. In addition, we observe that some sentiment words are only used for one specific aspect (*e.g.* "delicious" is only used in the *Food* aspect). Hence, another challenge is how to leverage the correlation between aspect words and sentiment words in a sentence to further improve the performance.

Our idea to attack the problem is to utilize an autoencoder to discover the aspect and sentiment structure hidden in sentences. By training the autoencoder to reconstruct sentences in the unlabeled corpus, we can learn a "dictionary" where each aspect and each sentiment can be characterized by a vector in the embedding space, reflecting the frequent words of the corresponding aspect and sentiment. We also design a regularization on the dictionary to instill the user guidance, such that the learned vectors in the dictionary remain close to the seed words in the embedding space. Moreover, we adapt the autoencoder structure so that the model is capable of learning a sentiment dictionary for each aspect, characterizing the aspect-specific sentiment in the embedding space.

More concretely, we make the following contributions:

- *Modeling the aspect and sentiment of sentences with user guidance.* We employ a structure with two parallel autoencoders to learn the aspect dictionary and sentiment dictionary of the corpus by reconstructing unlabeled sentences. We propose a regularization method to integrate user guidance into the modeling process. We also attach an attention layer to identify aspect and sentiment words in sentences.
- *Characterizing aspect-specific sentiment words.* We adapt the model by joining the aspect and sentiment encoders to reconstruct the sentiment of sentences. Thereby we can learn a sentiment dictionary for each aspect which captures the signals from aspect-specific sentiment words.
- *Conducting experiments on real data sets.* We verify the effectiveness of our proposed methods on two real world data sets from different domains.

2 RELATED WORK

In this section, we review unsupervised and weakly supervised effort in aspect-based sentiment analysis.

Lexicon-based methods. A series of studies on aspect-based sentiment analysis utilize aspect and/or sentiment lexicons. Some methods directly leverage an existing lexicon from an external source, such as [27] which uses a sentiment lexicon. Other methods develop algorithms to automatically build the aspect and/or sentiment lexicons.

Frequency-based methods construct the lexicons by counting the frequencies of each word in a given corpus and developing reasonable measures to distinguish aspect/sentiment words from others. Hu *et al.* [9] use a frequency-based method to identify frequent nouns to build the aspect lexicon. Then, they extract adjectives adjacent to the identified aspect words to build the sentiment lexicon. This method relies on part-of-speech (POS) tags of words in sentences. Popescu *et al.* [31] develop another frequency-based method and achieve improvement from [9], but they rely on more external resources such as the web statistics data. Scaffidi *et al.* [33] also propose a frequency-based method developed from a statistical test to construct the aspect lexicon but still requires the POS tags.

Syntax-based methods further leverage the syntactic structure of each word occurrence in the lexicon construction process. Qiu *et al.* [32] choose to build the lexicons from some seed aspect or sentiment words by syntactic rules. They parse the dependency structure of each sentence and construct the lexicon from the given seed words. However, the quality of their method heavily rely on the accuracy of the dependency parser, which can be low on a new domain without training data. Moreover, the method requires users to specify syntactic rules, while users are not necessarily familiar with linguistic knowledge. Although there are some follow-up studies to improve this algorithm, they still suffer from these drawbacks [19, 20]. Zhang *et al.* [42] also utilize similar ideas, with a different set of rules, as well as a HITS-based algorithm to rank the aspects. Zhao *et al.* [43] study to generalize some syntactic structures for better coverage on aspect extraction.

A recent study [44] also starts with seed aspect and sentiment words and use pattern-based methods. However, their final objective is to perform aspect-based sentiment analysis on documents, while our method focuses on sentence-level analysis.

Topic-model-based methods. Generative topic models are also frequently adopted to model the aspect and sentiment data.

A series of work by Wang *et al.* [38, 39] propose generative models to predict rating on each aspect. Nevertheless, their work rely on additional overall rating data for each review. Titov and McDonald [37] also propose a multi-aspect sentiment model to jointly characterize aspect occurrences and sentiment ratings of users. Similarly, they rely on the rating as supervision. Mei *et al.* [24] study a topic model for the general sentiment as well as the dynamics of topics. They focus more on corpus level summarization, while our objective is sentence-level aspect-based sentiment analysis.

Jo and Oh [11] propose a sentence-level generative topic model. Their model is capable of performing sentence-level aspect-based sentiment analysis from similar input as ours. However, they only leverage the co-occurrence signals between words for semantic proximity, without enjoying the benefit of recent progress on word embedding. Similarly, Lin and He [16] utilize a seed set of sentiment words with polarity labels, but with a much larger size.

Zhuang *et al.* [44] also propose a generative topic model to perform sentiment classification on top of the lexicons. Their model treats each word as an embedding vector and infer their distribution in the embedding space, thus is capable of utilizing the semantic proximity signals provided by word embedding. However, this model relies on co-occurrence information within documents, and thus cannot be directly applied to our sentence-level scenario.

Neural-network-based methods. Recently, there are also some neural-network-based methods focusing on unsupervised or weakly supervised aspect-based sentiment analysis. He *et al.* [6] propose an unsupervised neural model to extract aspects by an attention mechanism. However, the granularity of aspects generated by their method tends to be too fined and requires a manual step to map the learned aspects to the desired aspect classes, whereas our method can utilize user guidance to generate desired aspect classes. Moreover, they focus more on aspect extraction, and do not study sentiment analysis. Karamanolakis *et al.* [13] explore to use seed aspect words as user guidance for aspect extraction, but they do not perform sentiment analysis jointly. Angelidis *et al.* [1, 2] also explore to use seed aspect words as user guidance and perform aspect extraction, but their sentiment prediction module still requires overall sentiment ratings as training data. Our method only relies on an unlabeled corpus and some seed words.

3 PRELIMINARIES

In this section, we start by introducing basic notations of our data set. Then we move on to formalize the research problem.

3.1 Notations

We represent a given corpus as a set of documents $\mathcal{D} = \{d_i\}_{i=1}^{n}$, where each document d_i can be represented as a sequence of sentences $d_i = (s_1, \ldots, s_{|d_i|})$. A sentence s_j consists of a sequence of words $s_j = (w_1, \ldots, w_{|s_j|})$, where each word[1] w_k takes a value from a vocabulary $\mathcal{V}$.

For each word w in the vocabulary $\mathcal{V}$, one can derive an embedding vector from word embedding technique (*e.g.*, [26]). More precisely, the word embedding for each $w \in \mathcal{V}$ is a vector $\mathbf{e}_w \in \mathbb{R}^{\nu}$, where ν is the number of dimensions of the embedding space. The semantic proximity between two words should be reflected by the similarity of their embedding vectors. A popular similarity measure is cosine similarity, defined as:

$$\mathrm{sim}(\mathbf{e}_w, \mathbf{e}_{w'}) = \frac{\mathbf{e}_w \cdot \mathbf{e}_{w'}}{\|\mathbf{e}_w\| \times \|\mathbf{e}_{w'}\|} \tag{1}$$

In addition, there are K aspects in the given domain. For each document d_i, a subset of sentences contain aspect-specific description along with sentiment orientation. We refer to these sentences as aspect-sentiment-present sentences, while others are aspect-sentiment-absent sentences. Each aspect-sentiment-present sentence s_j can be associated with an aspect $a_j \in \{1, \ldots, K\}$ as well as a sentiment label y_j where $y_j \in \{0, 1\}$. 0 stands for *Negative* sentiment, and 1 stands for *Positive* sentiment.

[1]Notice that due to the phrase mining step and the tokenization step during the preprocessing of each corpus, each "word" w_k can actually be a unigram word, a multigram phrase (*e.g.*, "battery life", "chocolate cake") or a subword like "n't" in "don't". We simply use the term "word" for simplicity.

In principle, a complete pipeline should contain a classifier to determine whether a sentence is aspect-sentiment-present followed by a classifier to perform the aspect-based sentiment analysis. Neither of these sub-tasks have been studied in such a weakly-supervised scenario where users can only provide keywords. In this paper, we focus on solving the latter sub-task and leave the more challenging former sub-task to future work.

Below we will provide a formalized problem description for the joint analysis of aspect and sentiment.

3.2 Problem Formalization

First, an unlabeled corpus of reviews $\mathcal{D}$ from a specific domain should be given. A domain refers to a relatively consistent category of products or services, such as the hotel domain, the restaurant domain, and the laptop domain. We assume users have a complete set of aspects of interest in the given domain. For example, in the restaurant domain, the set of aspects would be {*Food*, *Service*, *Ambience*, *Location*, *Drinks*}, corresponding to aspects 1 to K respectively.

Users can provide some seed aspect words and seed sentiment words as guidance. Seed aspect words are a group of word sets $\mathcal{V}_{A_1}, \ldots, \mathcal{V}_{A_K}$, where each word set $\mathcal{V}_{A_t} \subset \mathcal{V}$ corresponds to an aspect. Similarly, seed sentiment words can be denoted as $\mathcal{V}_{S_0}$ and $\mathcal{V}_{S_1}$, corresponding to *Negative* and *Positive* sentiments.

Example 1. *In the example illustrated in Figure 1, aspect* 1 *is* Food *and its seed words given by users are* $\mathcal{V}_{A_1}$ = { *"food", "chicken", "appetizer"* }. *Similarly, aspect* 2 Service *has a set of seed words* $\mathcal{V}_{A_2}$ = {*"server", "staffs", "waiter"*}. *Seed sentiment words* $\mathcal{V}_{S_1}$ = {*"good", "great", "nice"*}, *and* $\mathcal{V}_{S_0}$ = {*"gross", "bad", "terrible"*}.

Notice that we do not require a ridiculously large set of seed words from users. For example, only 5 words for each aspect or each sentiment class would be sufficient. This setting can be easily fulfilled within minutes by a user with common sense knowledge about the data without any additional linguistic expertise, language resource or exhaustive labor.

The problem can be formalized as:

Problem 1. *Given a corpus of review documents* $\mathcal{D}$, *some seed aspect words* $\mathcal{V}_{A_1}, \ldots, \mathcal{V}_{A_K}$ *for each of the* K *aspects, and seed sentiment words* $\mathcal{V}_{S_0}, \mathcal{V}_{S_1}$ *for* Negative *and* Positive *sentiments, we aim to develop a classifier such that for any aspect-sentiment-present sentence* s_j, *we can output its aspect label and corresponding sentiment labels* (a_j, y_j).

It is worth noting that this problem aims to perform *sentence-level* aspect and sentiment analysis, which is a more challenging task from the *document-level* aspect-based sentiment analysis problem studied in [2, 44]. In document-level analysis, even if the algorithm misclassified a few sentences, the final output could still be correct if there are multiple sentences referring to the same aspect. In contrast, in sentence-level analysis, the algorithm needs to strive for the correctness of every sentences. The performance evaluation will also be based on sentence-level correctness. More importantly, if we can perform reasonable good sentence-level aspect-based sentiment analysis only with a few seed words provided by users as guidance, we can essentially perform document-level aspect-based sentiment analysis in the same manner.

4 ASPECT SENTIMENT AUTOENCODER

In this section, we describe a neural model to characterize the aspect and sentiment for each sentence in a given corpus.

4.1 Sentence Representation with Attention

For each sentence s_j, we first construct its vector representations for aspect and sentiment respectively, denoted as $\mathbf{z}_j^A$ and $\mathbf{z}_j^S$. The aspect vector representation $\mathbf{z}_j^A$ aims to summarize the aspect-relevant information from the sentence in the embedding space, while the sentiment vector representation $\mathbf{z}_j^S$ should capture the sentiment-related information.

Both vectors $\mathbf{z}_j^A$ and $\mathbf{z}_j^S$ are defined as weighted sums of word embedding vectors $\mathbf{e}_{w_k}$ for $w_k \in s_j$ where w_k is valid, *i.e.* not a stop word, a number or a punctuation. More precisely,

$$\mathbf{z}_j^A = \sum_k \alpha_k^A \mathbf{e}_{w_k}, \qquad \mathbf{z}_j^S = \sum_k \alpha_k^S \mathbf{e}_{w_k}$$

where the weights α_k^A and α_k^S are non-negative attention scores derived from attention models. Generally, the weights α_k^A and α_k^S can be regarded as the probabilities that the k-th word w_k should be utilized to determine the aspect and sentiment class respectively.

The mechanisms to derive aspect attention and sentiment attention in this model are similar. Therefore, we only present the aspect attention mechanism. The aspect attention weight α_k^A can be calculated based on the embedding of word w_k as well as the global context of the entire sentence. More concretely:

$$\alpha_k^A = \frac{\exp(u_k)}{\sum_k \exp(u_k)} \tag{2}$$

where u_k is obtained by:

$$u_k = \mathbf{e}_{w_k}^\top \mathbf{M}_A \mathbf{x}_j, \qquad \mathbf{x}_j = \frac{1}{|m_j|} \sum_k \mathbf{e}_{w_k}$$

The matrix $\mathbf{M}_A \in \mathbb{R}^{\nu \times \nu}$ can be learned during the training process. $\mathbf{x}_j$ is the unweighted average word embedding, and m_j is the number of valid words in s_j.

The attention mechanism for sentiment is similar, while the transformation matrix is replaced by another matrix $\mathbf{M}_S$.

4.2 Sentence Reconstruction

We have two vector representations $\mathbf{z}_j^A$ and $\mathbf{z}_j^S$ of the sentence s_j, which emphasize on the aspect and the sentiment of the sentence respectively. Now we build an autoencoder to extract the hidden knowledge from massive data by learning to encode these representations to a low dimension vector and then reconstruct them from a dictionary. A dictionary is a matrix where each row is a vector in the embedding space, representing an aspect or a sentiment class. Learning the dictionary that best recovers sentences in the corpus is essentially capturing the most frequent semantic regions in the embedding space, which serves as a good summary of the corpus. Moreover, since the dictionary shares the same embedding space with words, it is straightforward to introduce regularization on the dictionary from user-provided seed words. The regularization will be described in Section 4.3.

Reconstructing the aspect/sentiment vector. We only elaborate on the reconstruction of the aspect vector. The reconstruction

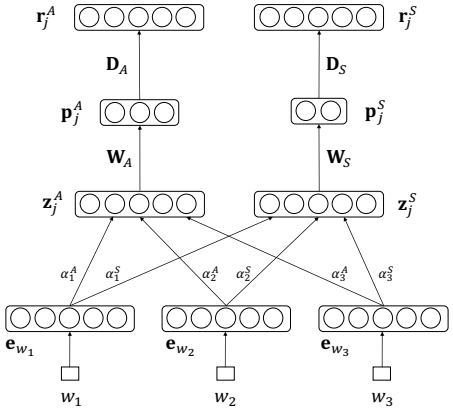


Figure 2: A graphical illustration of the aspect sentiment autoencoder model.

of sentiment vector is similar. We first reduce the aspect vector representation $\mathbf{z}_j^A$ to a K-dimension vector by:

$$\mathbf{p}_j^A = \text{softmax}(\mathbf{W}_A \cdot \mathbf{z}_j^A) \tag{3}$$

where $\mathbf{W}_A \in \mathbb{R}^{K \times \nu}$ are model parameters to be learned. The softmax activation introduces non-linearity and ensures the resulting compressed vector $\mathbf{p}_j^A$ is non-negative with the ℓ_1-norm of 1.

The vector $\mathbf{p}_j^A$ can be viewed as a distribution over different aspects where each element represents the probability that sentence s_j belongs to the corresponding aspect. In order to enforce this property, we try to reconstruct the original aspect vector representation of the sentence $\mathbf{z}_j^A$ from $\mathbf{p}_j^A$, with the help of an aspect dictionary $\mathbf{D}_A$:

$$\mathbf{r}_j^A = \mathbf{D}_A^\top \cdot \mathbf{p}_j^A \tag{4}$$

where $\mathbf{D}_A \in \mathbb{R}^{K \times \nu}$ is the aspect dictionary to be learned. Each row of $\mathbf{D}_A$ can be regarded as an embedding vector of an aspect in the embedding space. Ideally, the vector of an aspect should be close to representative words frequently mentioned in this aspect.

Discussion. Some studies [2, 6] also adopt a similar autoencoder structure to learn the aspect dictionary. However, instead of deriving and reconstructing a representation for the entire sentence [6], we derive and reconstruct two separate representations for aspect and sentiment respectively.

We also utilize regularization to confine the learned dictionary to be aligned with user guidance (described below). We extend the regularization-based method to both aspect and sentiment analysis and we believe the user guidance required by our method (just seed words) is substantially less than the guidance required by [2] (seed words and overall sentiment ratings).

4.3 Regularization

We place several regularization terms on the decoder parameters $\mathbf{D}_A$ and $\mathbf{D}_S$ to leverage the user-provided seed words.

Seed regularization. We leverage the information from seed words by applying a regularization on the dictionary parameters.

First, we describe the regularization on the aspect dictionary matrix $\mathbf{D}_A$. We create a "prior" matrix $\mathbf{R}_A \in \mathbb{R}^{K \times \nu}$ with the same

size as parameter $\mathbf{D}_A$. The t-th row of $\mathbf{R}_A$ is assigned with the average embedding of seed words in the corresponding aspect:

$$\mathbf{R}_A^{(t)} = \frac{\sum_{w \in \mathcal{V}_{A_t}} \mathbf{e}_w^\top}{\| \sum_{w \in \mathcal{V}_{A_t}} \mathbf{e}_w^\top \|} \tag{5}$$

The objective is to penalize when the learned embedding of the t-th aspect (represented by the t-th row of $\mathbf{D}_A$) deviates too far away from the average embedding of seed words. Accordingly, the regularization term can be written as:

$$C_A(\theta) = \sum_{t=1}^{K} \left[1 - \text{sim}\left(\mathbf{R}_A^{(t)}, \mathbf{D}_A^{(t)} \right) \right] \tag{6}$$

where $\mathbf{R}_A^{(t)}$ and $\mathbf{D}_A^{(t)}$ represents the t-th row of $\mathbf{R}_A$ and $\mathbf{D}_A$ respectively. The prior matrix and regularization term for sentiment dictionary can be obtained in similar way, denoted as $\mathbf{R}_S^{(y)}$ and $C_S(\theta)$.

Redundancy regularization. Another regularization typically adopted is the penalty on redundancy of learned dictionaries. The intuition is to prevent the dictionary from having almost identical rows. For a learned dictionary matrix $\mathbf{D}$, the regularization term is:

$$X(\mathbf{D}) = \left\| \mathbf{D}_n \mathbf{D}_n^\top - \mathbf{I} \right\| \tag{7}$$

where $\mathbf{D}_n$ is $\mathbf{D}$ with each row normalized to a unit length vector, and $\mathbf{I}$ is the identity matrix. This term is also utilized in [6].

We apply the redundancy regularization term on all the dictionary matrices. We define $X_A(\theta) = X(\mathbf{D}_A)$ and $X_S(\theta) = X(\mathbf{D}_S)$.

4.4 Training Objective

The overall objective is to minimize the loss between the reconstructed vectors and the sentence representation vectors for both aspect and sentiment. The similarity between two vectors are measured by cosine similarity. For each sentence, we aim to maximize the similarity between the derived sentence representation vectors and the reconstructed vectors.

In addition, we randomly sample m sentences as *negative* samples. We take the average embedding of all the words in the i-th negative sentence, denoted as $\mathbf{x}_{n_i}$. We also try to minimize the similarity between the vector representations of negative samples and the reconstructed vectors.

Thus, the loss is formalized in a similar way to the objective functions proposed in [6]:

$$L_A(\theta) = \sum_{s_j \in \mathcal{D}} \sum_{i=1}^{m} \max\left(0, 1 - \text{sim}\left(\mathbf{r}_j^A, \mathbf{z}_j^A\right) + \text{sim}\left(\mathbf{r}_j^A, \mathbf{x}_{n_i}\right)\right) \tag{8}$$

$$L_S(\theta) = \sum_{s_j \in \mathcal{D}} \sum_{i=1}^{m} \max\left(0, 1 - \text{sim}\left(\mathbf{r}_j^S, \mathbf{z}_j^S\right) + \text{sim}\left(\mathbf{r}_j^S, \mathbf{x}_{n_i}\right)\right) \tag{9}$$

The final training objective is to minimize the overall loss along with the regularization term:

$$\begin{aligned} L(\theta) = L_A(\theta) + L_S(\theta) + \lambda_1 \big(C_A(\theta) + C_S(\theta)\big) \\ + \lambda_2 \big(X_A(\theta) + X_S(\theta)\big) \end{aligned} \tag{10}$$

where λ_1 and λ_2 are two hyperparameters given by users to weigh the effect of different regularization terms.

The model can be trained end-to-end with both the attention module and the sentence reconstruction module learned together.

5 JOINT ASPECT SENTIMENT AUTOENCODER

The model described above consists of two parallel autoencoder structures for aspect and sentiment respectively. However, the occurrences of aspect and sentiment words in sentences are correlated.

In this section, we describe a joint autoencoder for aspect and sentiment in sentences. The model is designed based on the observation that some sentiment words are specifically used for a certain aspect. For example, "delicious" is specifically used to express positive sentiment on *Food* aspect, while "rude" is often used to express negative sentiment on *Service* aspect.

We capture the aspect-specific sentiment words by expanding the universal sentiment dictionary into several aspect-based sentiment dictionaries. To learn the dictionaries, we join the aspect and sentiment encoder to generate the distributions over the space of aspect-sentiment pairs. Then we accordingly reconstruct the sentiment representation of sentences.

Since the attention and aspect reconstruction part is identical to the model described in Section 4, we only present the different parts of the model.

5.1 Sentence Reconstruction

In this subsection, we only focus on the joint reconstruction of sentiment representation vector.

The intuition is to capture the aspect-specific sentiment words in sentiment dictionaries of each aspect. The occurring probability of an aspect-specific word in a sentence s_j is not only related to the sentiment class of the current sentence, but also the aspect class of the current sentence. Therefore, We start by joining the aspect distribution $\mathbf{p}_j^A$ and the sentiment distribution $\mathbf{p}_j^S$.

We derive the joint encoded vector $\mathbf{p}_j^{AS}$ by taking the outer product of $\mathbf{p}_j^S$ and $\mathbf{p}_j^A$ and then flattening it into a vector:

$$\mathbf{p}_j^{AS} = \text{vec}\left(\mathbf{p}_j^S \otimes \mathbf{p}_j^A\right) \tag{11}$$

where $\mathbf{p}_j^{AS}$ will be a $2K$-dimension vector. The $(2t-1)$-th element of $\mathbf{p}_j^{AS}$ can be interpreted as the probability that sentence s_j expresses *Negative* sentiment on the t-th aspect, while the $(2t)$-th corresponds to *Positive* sentiment on the t-th aspect.

In order to reconstruct the sentiment representation from the joint aspect and sentiment distribution $\mathbf{p}_j^{AS}$, we need to learn a larger sentiment dictionary. More specifically, the sentiment dictionary has to be aspect-specific. We denote the aspect-specific dictionary as $\mathbf{D}_{AS} \in \mathbb{R}^{2K \times \nu}$, where the $(2t-1)$-th and $(2t)$-th row form a sentiment dictionary for the t-th aspect.

Now we can reconstruct the sentiment vector $\mathbf{z}_j^S$ from the joint distribution over aspect and sentiment by:

$$\mathbf{r}_j^S = \mathbf{D}_{AS}^\top \cdot \mathbf{p}_j^{AS} \tag{12}$$

By minimizing the loss between $\mathbf{r}_j^S$ and $\mathbf{z}_j^S$, the $(2t-1)$-th and $(2t)$-th row of $\mathbf{D}_{AS}$ should become the vector representation of the *Negative* and *Positive* sentiment for the t-th aspect in the embedding space respectively. Notice that the learned sentiment dictionary for each aspect summarizes *all* possible sentiment words co-occurred with the aspect, which include both aspect-specific sentiment words

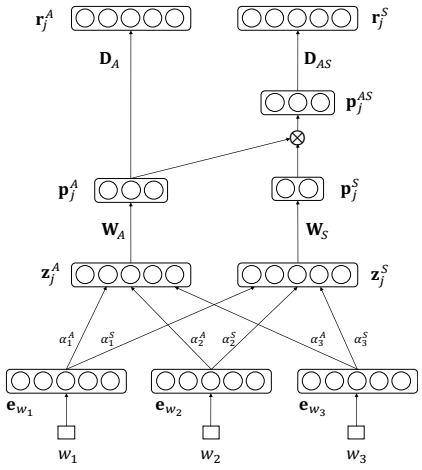


Figure 3: A graphical illustration of the joint aspect sentiment autoencoder model.

as well as general sentiment words. For example, the embedding for *Positive* sentiment on *Food* aspect should still be close to general words like "good" and "great", while also relatively close to aspect-specific words like "delicious" and "yummy".

Discussion. An alternative design is to concatenate the aspect and sentiment vector representation $\mathbf{z}_j^A$ and $\mathbf{z}_j^S$ and directly derive the joint probability distribution $\mathbf{p}_j^{AS}$ over aspect and sentiment. However, our preliminary experiments show that the model cannot learn meaningful attention and dictionaries. It could be due to the over-complicated interaction between aspect and sentiment representation vectors in this model which introduces a larger number of parameters to be learned.

5.2 Regularization

Since we are learning the aspect-specific sentiment dictionary $\mathbf{D}_{AS}$, the regularization on the dictionary needs to be adapted accordingly. We construct an expanded prior matrix $\mathbf{R}_{AS} \in \mathbb{R}^{2K \times \nu}$ by repetitively tiling the prior matrix we utilized in Equation (5) into the new prior matrix $\mathbf{R}_{AS}$. More specifically,

$$\mathbf{R}_{AS} = [\mathbf{R}_S^{(0)\top}\ \mathbf{R}_S^{(1)\top}\ \mathbf{R}_S^{(0)\top}\ \mathbf{R}_S^{(1)\top}\ \cdots\ \mathbf{R}_S^{(1)\top}]^\top \tag{13}$$

Essentially, if we regard every two rows in $\mathbf{D}_{AS}$ as a sentiment dictionary for each aspect, then each of them is regularized in the same way as the general sentiment dictionary $\mathbf{D}_S$. However, if users are willing to give some seed words for each aspect-specific sentiment, one can easily instantiate a more informative prior regularization within this framework. We leave this idea for future extensions.

The regularization term on $\mathbf{D}_{AS}$ is similar:

$$C_{AS}(\theta) = \sum_{r=1}^{2K} \left[1 - \text{sim}\left(\mathbf{R}_{AS}^{(r)}, \mathbf{D}_{AS}^{(r)}\right)\right] \tag{14}$$

The overall loss function is similar to Equation (10) with updated regularization terms.

Although the major difference of the model structure is only reflected in the sentiment reconstruction part, the training results show substantial differences in the aspect reconstruction. The current model structure allows the aspect encoder to contribute substantially to the reconstruction of sentiment representations, especially those with aspect-specific sentiment words. As the model minimizes the reconstruction loss, the aspect autoencoder will also shift some attention to such words and utilize them in the aspect classification. We present some concrete examples in Section 6.

6 EXPERIMENTAL RESULTS

In this section, we conduct experiments to answer the following three research questions:

- **RQ1.** How do the proposed methods perform compared to baseline methods with similar user guidance?
- **RQ2.** How do different parameter configurations in the proposed methods affect the performance?
- **RQ3.** What do the proposed models learn and why do they perform better?

6.1 Data Sets

We introduce the data sets used in our experiments.

Restaurant. We collect 47,239 unlabeled reviews on restaurants from a public Yelp data set[2]. For the purpose of evaluation, we utilize sentences from SemEval-2016 [30] in the restaurant domain as ground-truth, where each sentence is labeled with target entities, entity types and attributes, as well as corresponding sentiment polarity (*Positive*, *Negative* and *Neutral*). We regard the entity types as aspect classes, while neglecting the entity type "Restaurant" since such sentences do not express aspect-specific opinions and are not our targets. We ignore the attributes of entities since they provide more fine-grained information. We also remove sentences with *Neutral* sentiment class to simplify the problem, but it can be seamlessly added with an extra set of seed words.

Laptop. We utilize 14,683 unlabeled Amazon reviews on merchandises in the laptop category, collected by [8, 23]. We also use labeled sentences on the laptop domain from SemEval-2016 [30] for evaluation. Similar to the Restaurant data set, each sentence is labeled with target entities, entity types, attributes and sentiment polarity. There are originally 21 different entity types. We remove some rare entity types and only keep the following 8 entity types as aspect classes: *Support*, *OS*, *Display*, *Battery*, *Company*, *Mouse*, *Software* and *Keyboard*. Again, we ignore the attributes and remove sentences with neutral sentiment.

6.2 Experiment Setup

Preprocessing. The unlabeled review documents serve as the training corpus $\mathcal{D}$. We use the sentence tokenizer and word tokenizer provided by *NLTK*[3]. We also adopt a phrase mining technique, *SegPhrase* [17], to discover phrases such as "mini bar" and "front desk", such that they can be treated as a single semantic unit instead of several. SegPhrase can automatically segment sentences into chunks of unigram words and multigram phrases.

We then derive the word embedding by training *word2vec* [26] on our unlabeled set of documents. For each data set, we train a

[2]https://www.yelp.com/dataset/challenge
[3]https://www.nltk.org/

different set of word embedding. Notice that the corpus used for word embedding contains multigram phrases from SegPhrase. Thus each multigram phrase has its own embedding. This is coherent with [17]. Notice that our method does not rely on a specific word embedding algorithm so it can be seamlessly replaced by any other embedding methods.

Methods evaluated. We compare the following methods.

- *Cosine similarity (CosSim).* Performing aspect and sentiment classification by simply calculating the cosine similarity between the average embedding of all words in the given sentence and the average embedding of the seed words of each aspect/sentiment class. Classifying the sentence into the aspect/sentiment class with the highest cosine similarity.
- *Aspect Sentiment Unification Model (ASUM).* A topic model specifically proposed for aspect-based sentiment analysis by Jo *et al.* [11]. The model also takes seed aspect and seed sentiment words as guidance.
- *BERT with weakly-labeled data (BERT).* We utilize a pre-trained language model BERT [5] (12-layer, 768-hidden, uncased) to perform aspect and sentiment analysis individually. We fine-tune the model based on "pseudo-training data" from the unlabeled corpus by labeling sentences containing any seed word to the corresponding class.
- ⋆ *Aspect Sentiment Autoencoder (ASA).* Our model described in Section 4 with a parallel autoencoder structure.
- ⋆ *Joint Aspect Sentiment Autoencoder (JASA).* Our proposed model in Section 5 with a joint autoencoder.

Evaluation measures. We evaluate the performance of aspect and sentiment classification respectively. For both the aspect and sentiment classification task, we evaluate the performance by accuracy, precision, recall and F_1-score. To clarify, for the multi-class aspect classification task, we employ macro-averaged precision, macro-averaged recall and macro-averaged F_1-score as the evaluation measures.

For our neural network model, we run the experiments 10 times for each method on each data set and report the average performance to reduce the effect of randomness.

Configurations. For both data sets, two annotators are asked to read the unlabeled corpus and write down 10 seed words for each aspect and each sentiment. Then the two annotators need to discuss their selections and finally agree on 5 seed words for each aspect and each sentiment. We will test baseline methods and our proposed models based on this set of seed words.

We utilize Adam [14] with default parameter setting to optimize the model. For both data sets, we set the weights for prior regularization and redundancy regularization to $\lambda_1 = 10$ and $\lambda_2 = 0.1$ respectively. The model is trained for 15 epochs, where each epoch contains $1,000$ batches. Each batch contains 50 randomly sampled sentences. Each randomly sampled sentence is paired with $m = 20$ negative sentences as negative samples.

6.3 Results

Performance comparison (RQ1). We proceed to evaluate the performance of aspect classification and sentiment classification separately on both Restaurant and Laptop data sets. The overall performance of aspect classification results are in Table 1 and the sentiment classification results are in Table 2.

Table 1: Performance of aspect classification (%).

Data set	Method	Accuracy	Precision	Recall	F_1
Restaurant	CosSim	78.67	**65.47**	57.39	59.83
	ASUM	30.79	27.01	26.25	24.75
	BERT	75.98	62.30	80.26	60.83
	ASA	80.19	62.95	82.50	66.96
	JASA	**80.83**	63.67	**82.57**	**67.70**
Laptop	CosSim	55.81	67.46	60.35	56.81
	ASUM	34.24	26.81	32.01	28.21
	BERT	56.21	59.49	56.72	54.13
	ASA	76.03	76.70	78.94	76.87
	JASA	**77.16**	**77.76**	**79.81**	**77.97**

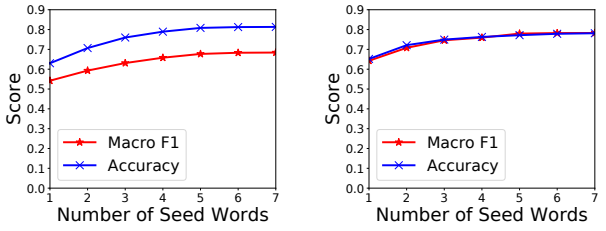
Table 2: Performance of sentiment classification (%).

Data set	Method	Accuracy	Precision	Recall	F_1
Restaurant	CosSim	77.23	75.57	74.92	75.20
	ASUM	71.31	69.47	70.16	69.72
	BERT	79.50	77.72	77.99	77.85
	ASA	82.06	80.73	80.64	80.68
	JASA	**82.34**	**80.95**	**81.31**	**81.12**
Laptop	CosSim	73.03	74.31	74.40	73.03
	ASUM	67.26	66.67	66.75	66.70
	BERT	59.68	64.10	71.26	57.52
	ASA	74.30	**74.83**	**75.23**	74.26
	JASA	**74.42**	74.46	74.94	**74.30**

In the task of aspect classification, both of our proposed methods ASA and JASA are able to achieve the best overall performances in terms of accuracy. ASA and JASA achieve over 80% of accuracy in Restaurant data set and 76-77% of accuracy in Laptop data set. The baselines based on word embedding (CosSim) or pre-trained neural language model (BERT) also show good performance. Surprisingly, BERT does not always outperform simple baseline like CosSim in spite of the extra knowledge it utilizes from the pre-training corpus. This highlights the challenge of applying pre-trained language models like BERT on this task since the "pseudo-training data" for fine-tuning is not necessarily accurate. The topic-model-based baseline ASUM fails to identify most aspects and therefore has poor overall performance on both data sets.

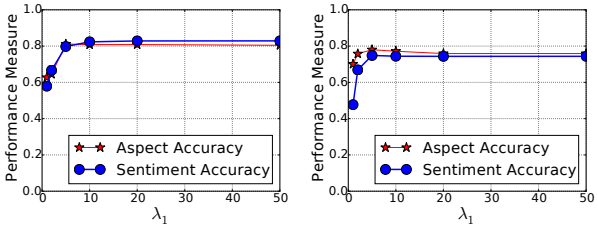
Since the aspect analysis module of ASA shares some similarity to the aspect analysis module in [2], we further compare the performance between ASA and JASA. It can be observed that JASA with the joint autoencoder structure improves the aspect classification accuracy on both data sets for +0.7% to +1.1% from the ASA model with parallel autoencoder structures. We perform a paired Student's t-test showing that the improvement from ASA to JASA is statistically significant on Restaurant and Laptop data set with significance level 0.1% and 0.5% respectively. This verifies the benefit of exploiting the correlation between aspect and sentiment words in sentences.

For sentiment classification, our methods ASA and JASA still outperform all the other baselines in both data sets, in terms of all the evaluation measures. Both methods achieve more than 82% of accuracy in Restaurant data set and 74% of accuracy on Laptop data set, with +1% to +5% improvement from baseline methods. BERT



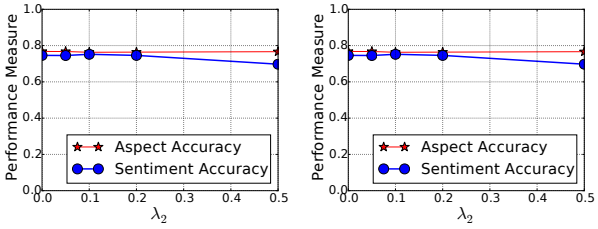
(a) Restaurant (b) Laptop

Figure 4: Performance with different number of seeds.



(a) Restaurant (b) Laptop

Figure 5: Performance of JASA with different values of λ_1.



(a) Restaurant (b) Laptop

Figure 6: Performance of JASA with different values of λ_2.

baseline achieves noticeably low performance on Laptop data set for sentiment classification, because sentences with negative sentiment in Laptop data set usually contain more complicated expressions (*e.g.* "be careful" is *Negative* whereas "carefully handled" is *Positive*). These sentences are likely to be wrongly labeled in the generated "pseudo-trainig data" and thereby introduce more noises during the fine-tuning of pre-trained BERT model.

Notice that our reported results are average measures of 10 different results. The reported F_1-score is not directly calculated from the reported precision and recall.

Number of seeds (RQ2). Seed words play an essential role in the training of our model. We conduct an experiment to understand how the model performance changes with the number of seed words. We use the list of 10 seed words for each aspect from one annotator on both data sets. We randomly pick n seed words from each list, run our JASA model for 10 times and report the average performance of macro-F_1 and accuracy scores. We vary n from 1 to 7 and plot the results in Figure 4.

From Figure 4(a), we can see that on the Restaurant data set, the performance of our model improves significantly when $n < 5$ and gradually saturates when $n \geq 5$. The similar trend can be observed on Laptop data set, as shown in Figure 4(b). This verifies that our model only needs around 5 seed words for each aspect to achieve reasonable performance, which is affordable for most users.

Selection of hyperparameter λ_1 and λ_2 (RQ2). We explore the ideal selection of the weighting hyperparameter of prior regularization λ_1. We test the performance of JASA on both data sets with λ_1 set to different values from 2 to 50 while keeping $\lambda_2 = 0.1$.

Figure 5(a) shows that on Restaurant data set, when λ_1 is smaller than 5, both aspect accuracy and sentiment accuracy would substantially drop. When λ_1 is set to value from 5 to 50, the performance remains relatively stable. We can find similar patterns on Laptop data set, as shown in Figure 5(b). There is a slight performance drop in aspect accuracy when λ_1 is larger than 10.

The results are in line with our expectation as when λ_1 is small, our seed words provide limited regularization to the model which will lead to poor performance of both aspect analysis and sentiment analysis. On the other hand, the slight performance drop when λ_1 is larger than 10 on Laptop data set also shows the effectiveness of our model training on the unlabeled corpus. Based on the above analysis, we can set parameter λ_1 to any value between 5 to 10.

We also conduct experiments for the selection of weighting hyperparameter for redundancy regularization λ_2. We compare the performance of JASA by varying lambda λ_2 from 0.01 to 0.5 on both data sets, while λ_1 is set to 10. Figure 6 shows that on both data sets, the performance of both tasks is not very sensitive to the value of λ_2. This might imply that the redundancy regularization, although also adopted by other studies [2, 6], does not necessarily play a role in our models. In the future, we may consider a different redundancy regularization or simply remove this regularization.

Visualizing aspect-specific sentiment dictionary (RQ3). We compare learned aspect-specific sentiment dictionary $\mathbf{D}_{AS}$ to the seed words to understand our dictionary learning module. For the t-th aspect with the sentiment $y \in \{0, 1\}$, we denote the average embedding of seed sentiment word $\mathbf{R}_S^{(y)}$ as $\mathbf{v}$, and the learned embedding in the aspect-specific dictionary $\mathbf{D}_{AS}^{(2t-1+y)}$ as $\mathbf{v}'$.

Figure 7 shows the visualization of $\mathbf{v}$ and $\mathbf{v}'$ for *Food* aspect on both *Positive* and *Negative* sentiment. It can be observed that the learned embedding $\mathbf{v}'$ in the aspect-specific dictionary shifts towards the embedding of aspect-specific sentiment. For *Positive* sentiment, as presented in Figure 7(a), $\mathbf{v}'$ drifts towards words like "delicious", "tasty" and "yummy", which are specifically used to compliment *Food* aspect. For *Negative* sentiment (Figure 7(b)), $\mathbf{v}'$ is more close to words like "bland" and "flavorless", which are common words to criticize on *Food* aspect. On the other hand, $\mathbf{v}'$ does not completely distance itself from the average seed embedding $\mathbf{v}$ due to the regularization. This is because $\mathbf{v}'$ still needs to reflect general sentiment words like "good" or "terrible".

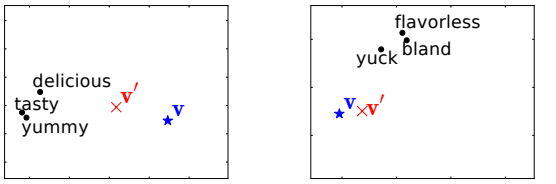
We also try to identify the sentiment words that gain the most "density" by learning the aspect-specific sentiment dictionary. To be specific, we find words with the most boosted density from a kernel function centered at $\mathbf{v}$ to another kernel function centered at $\mathbf{v}'$. We instantiate the kernel function $K_{\mathbf{v}}(\mathbf{e}) = \exp\Big(\text{sim}(\mathbf{e}, \mathbf{v})\Big)$, which is proportional to the probability density function of a von Mises-Fisher distribution with the concentration parameter as 1. For each word $w \in \mathcal{V}$, we calculate the difference of densities by:

$$\delta_w = K_{\mathbf{v}'}(\mathbf{e}_w) - K_{\mathbf{v}}(\mathbf{e}_w)$$

and rank the words by δ_w from the largest to the smallest.

Table 3: Comparing the learned embedding in dictionary D_{AS} and the average seed embedding. The table lists words with the most boosted densities δ_w in a cosine-similarity-based kernel function by moving the center from the average embedding of seed words v to the learned embedding in the dictionary v′.

Aspect	Sentiment	Words with largest δ_w
Location	Positive	cozy, sultry, nice
	Negative	ventured, seattle, resident
Drinks	Positive	awesome, huckleberry, boozy
	Negative	vomit, substance, clump
Food	Positive	delicious, tasty, yummy
	Negative	yuck, bland, flavorless
Ambience	Positive	nice, cool, clean
	Negative	vomit, enemy, dishonest
Service	Positive	friendly, courteous, personable
	Negative	frustrated, time, acknowledgement



(a) *Food* aspect, *Positive* sentiment (b) *Food* aspect, *Negative* sentiment

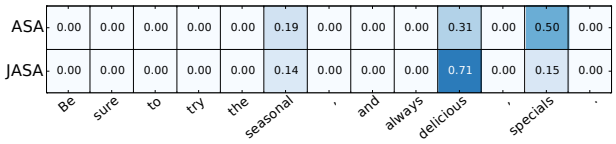
Figure 7: Visualization of learned embedding v′ in the dictionary D_{AS} on certain aspect and sentiment comparing to the average embedding of seed words v.

Table 3 presents the top-3 words for each aspect-sentiment pair. As one can see, many words are aspect-specific sentiment words. For example, for aspect *Food* with *Positive* sentiment, the top-ranked words are "delicious", "tasty" and "yummy", while for its *Negative* sentiment, words like "bland", "flavorless" are ranked high.

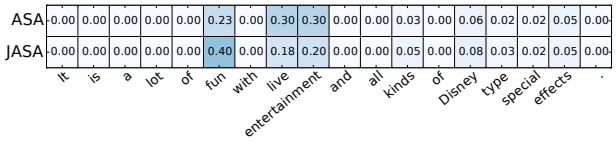
Case study (RQ3). In case study, we show the attention output of JASA and ASA to take an in-depth look at how JASA outperforms ASA. Figure 8 shows the aspect attention weights α_k^A's on sentences where JASA makes the correct classification but ASA fails.

In the first example shown in Figure 8(a), ASA places the most attention on the word "specials", which is sufficient to narrow down the possible aspect classes to *Food* or *Drinks*, but not enough to pinpoint the correct aspect class. In contrast, JASA places much more attention on the word "delicious", which is usually a sentiment word, but can also imply the aspect class since it is specifically used to describe food. Hence, JASA correctly classifies the sentence into *Food* aspect class with a high confidence (with predictive probability of 0.99). This is because the joint structure of aspect and sentiment in JASA allows the aspect attention mechanism and reconstruction mechanism to fit the training objective of sentiment representation reconstruction, which enables the aspect-specific sentiment words to benefit the aspect classification in such cases.

The second example is shown in Figure 8(b). This sentence is labeled with the *Ambience* aspect and the *Positive* sentiment. It can be observed that ASA does not have a particularly strong attention on one specific word. Instead, its attention scores are almost evenly



(a) Aspect attention derived by ASA and JASA of a sentence on *Food* aspect with *Positive* sentiment. ASA mistakenly classifies the sentence into *Drinks* aspect.



(b) Aspect attention derived by ASA and JASA of a sentence on *Ambience* aspect with *Positive* sentiment. ASA mistakenly classifies the sentence into *Location* aspect.

Figure 8: Comparative case study on aspect attention generated by ASA and JASA respectively.

distributed on words "fun", "live" and "entertainment". It mistakenly classifies the sentences into *Location* aspect. Again, JASA shifts more attention on the word "fun". Since "fun" is more frequently related to the *Ambience* aspect, JASA is able to output the correct aspect label *Ambience*.

7 CONCLUSION

In this paper, we study to develop sentence-level aspect-based sentiment analysis with minimal user guidance, where users only need to provide a small set of seed words for each aspect class and each sentiment class as well as an unlabeled corpus of reviews. We utilize the autoencoder structure with its dictionary regularized by user-provided seed words for both aspect and sentiment modeling. We also propose a model with joint aspect and sentiment encoder to capture aspect-specific sentiment words. The experimental results show the effectiveness of our model on two data sets.

Our methods can be further extended to leverage more signals from the unlabeled corpus that can potentially benefit the aspect-based sentiment analysis. We provide some possible future directions: 1) Utilizing correlation of aspect-based sentiment within document: A basic intuition is that the same review usually expresses the same sentiment on the same aspect. Integrating the document structure into the model to capture this signal may benefit the sentiment modeling. 2) Leveraging user guidance in an interactive way: Allowing users to further modify and improve their seed words based on learned models. Our method can be utilized to further improve the performance of some existing prototype system [29].

Acknowledgments. Research was sponsored in part by US DARPA KAIROS Program No. FA8750-19-2-1004 and SocialSim Program No. W911NF-17-C-0099, National Science Foundation IIS 16-18481, IIS 17-04532, and IIS-17-41317, and DTRA HDTRA11810026. Any opinions, findings, and conclusions or recommendations expressed herein are those of the authors and should not be interpreted as necessarily representing the views, either expressed or implied, of DARPA or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright annotation hereon.

REFERENCES

[1] Stefanos Angelidis and Mirella Lapata. 2018. Multiple Instance Learning Networks for Fine-Grained Sentiment Analysis. *TACL* 6 (2018), 17–31.

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

[3] Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. 2017. Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In *KDD*.

[4] Li Chen, Guanliang Chen, and Feng Wang. 2015. Recommender systems based on user reviews: the state of the art. *User Modeling and User-Adapted Interaction* (2015).

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

[6] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An unsupervised neural attention model for aspect extraction. In *ACL*.

[7] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Exploiting Document Knowledge for Aspect-level Sentiment Classification. In *ACL*.

[8] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*.

[9] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *KDD*.

[10] Wei Jin, Hung Hay Ho, and Rohini K Srihari. 2009. OpinionMiner: a novel machine learning system for web opinion mining and extraction. In *KDD*.

[11] Yohan Jo and Alice H Oh. 2011. Aspect and sentiment unification model for online review analysis. In *WSDM*.

[12] Jaap Kamps, Maarten Marx, Robert J. Mokken, and Maarten de Rijke. 2004. Using WordNet to Measure Semantic Orientations of Adjectives. In *LREC*.

[13] Giannis Karamanolakis, Daniel Hsu, and Luis Gravano. 2019. Leveraging Just a Few Keywords for Fine-Grained Aspect Detection Through Weakly Supervised Co-Training. In *EMNLP-IJCNLP*. 4603–4613.

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *ICLR*.

[15] Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Ying-Ju Xia, Shu Zhang, and Hao Yu. 2010. Structure-aware review mining and summarization. In *COLING*.

[16] Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *CIKM*.

[17] Jialu Liu, Jingbo Shang, Chi Wang, Xiang Ren, and Jiawei Han. 2015. Mining quality phrases from massive text corpora. In *SIGMOD*.

[18] Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *EMNLP*.

[19] Qian Liu, Zhiqiang Gao, Bing Liu, and Yuanlin Zhang. 2015. Automated Rule Selection for Aspect Extraction in Opinion Mining. In *IJCAI*.

[20] Qian Liu, Bing Liu, Yuanlin Zhang, Doo Soon Kim, and Zhiqiang Gao. 2016. Improving opinion aspect extraction using semantic similarity and aspect associations. In *AAAI*.

[21] Yue Lu, Malu Castellanos, Umeshwar Dayal, and ChengXiang Zhai. 2011. Automatic construction of a context-aware sentiment lexicon: an optimization approach. In *WWW*.

[22] Diego Marcheggiani, Oscar Täckström, Andrea Esuli, and Fabrizio Sebastiani. 2014. Hierarchical multi-label conditional random fields for aspect-oriented opinion mining. In *ECIR*.

[23] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *SIGIR*.

[24] Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. In *WWW*.

[25] Donatas Mešklė and Flavius Frasincar. 2019. ALDONA: a hybrid solution for sentence-level aspect-based sentiment analysis using a lexicalised domain ontology and a neural attention model. In *SAC*. 2489–2496.

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.

[27] Samaneh Moghaddam and Martin Ester. 2010. Opinion digger: an unsupervised opinion miner from unstructured product reviews. In *CIKM*.

[28] Thien Hai Nguyen, Kiyoaki Shirai, and Julien Velcin. 2015. Sentiment analysis on social media for stock movement prediction. *Expert Systems with Applications* (2015).

[29] Oren Pereg, Daniel Korat, Moshe Wasserblat, Jonathan Mamou, and Ido Dagan. 2019. ABSApp: A Portable Weakly-Supervised Aspect-Based Sentiment Extraction System. In *EMNLP-IJCNLP: System Demonstrations*.

[30] 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 *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*. 19–30.

[31] Ana-Maria Popescu and Orena Etzioni. 2007. Extracting product features and opinions from reviews. In *Natural language processing and text mining*.

[32] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational linguistics* (2011).

[33] Christopher Scaffidi, Kevin Bierhoff, Eric Chang, Mikhael Felker, Herman Ng, and Chun Jin. 2007. Red Opal: product-feature scoring from reviews. In *EC*.

[34] Kim Schouten and Flavius Frasincar. 2016. Survey on Aspect-Level Sentiment Analysis. *TKDE* (2016).

[35] Jianfeng Si, Arjun Mukherjee, Bing Liu, Qing Li, Huayi Li, and Xiaotie Deng. 2013. Exploiting topic based twitter sentiment for stock prediction. In *ACL*.

[36] Duyu Tang, Bing Qin, and Ting Liu. 2016. Aspect Level Sentiment Classification with Deep Memory Network. In *EMNLP*.

[37] Ivan Titov and Ryan T. McDonald. 2008. A Joint Model of Text and Aspect Ratings for Sentiment Summarization. In *ACL*.

[38] Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent aspect rating analysis on review text data: a rating regression approach. In *KDD*.

[39] Hongning Wang, Yue Lu, and ChengXiang Zhai. 2011. Latent aspect rating analysis without aspect keyword supervision. In *KDD*.

[40] Jin Wang, Liang-Chih Yu, K Robert Lai, and Xuejie Zhang. 2016. Dimensional sentiment analysis using a regional CNN-LSTM model. In *ACL*.

[41] Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for Aspect-level Sentiment Classification. In *EMNLP*.

[42] Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O'Brien-Strain. 2010. Extracting and ranking product features in opinion documents. In *COLING*.

[43] Yanyan Zhao, Bing Qin, Shen Hu, and Ting Liu. 2010. Generalizing syntactic structures for product attribute candidate extraction. In *NAACL*.

[44] Honglei Zhuang, Timothy Hanratty, and Jiawei Han. 2019. Aspect-Based Sentiment Analysis with Minimal Guidance. In *SDM*.

[45] Cäcilia Zirn, Mathias Niepert, Heiner Stuckenschmidt, and Michael Strube. 2011. Fine-grained sentiment analysis with structural features. In *IJCNLP*.