WEAKLY SUPERVISED ASPECT EXTRACTION FOR DOMAIN-SPECIFIC TEXTS

BY

FANG GUO

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2020

Urbana, Illinois

Adviser:

Professor Jiawei Han

ABSTRACT

Aspect extraction, identifying aspects of text segments from a pre-defined set of aspects, is one of the keystones in text understanding. It benefits numerous applications, including sentiment analysis and product review summarization. Most existing aspect extraction methods heavily rely on human-curated aspect annotations of massive text segments, thus making them expensive to be applied in specific domains. Recent attempts leveraging clustering methods can alleviate such annotation effort, but they require domain-specific knowledge and effort to further filter, aggregate, and align the clustering results to desired aspects. Therefore, in this paper, we explore to extract aspects from the domain-specific raw texts with very limited supervision – only a few user-provided seed words per each aspect. Specifically, our proposed neural model is equipped with multi-head attention and self-training. The multi-head attention is learned from the seed words to ensure that the aspect-related words in text segments are weighted higher than those unrelated ones. The self-training mechanism provides more pseudo labels in addition to limited supervision. Extensive experiments on real-world datasets demonstrate the superior performance of our proposed framework, as well as the effectiveness of both the attention module and the self-training mechanism. Case studies on the attention weights further shed lights on the interpretability of our aspect extraction results.

To my parents, for their love and support.

ACKNOWLEDGMENTS

I would love to express my great appreciation towards my advisor Professor Jiawei Han of the Department of Computer Science at University of Illinois at Urbana-Champaign. During my three-year master study, he was so kind and patient to guide me in research and live. Being a freshman in research field, I was illuminated and inspired by Professor Han and other data mining group members. During my thesis work, a lot of people in our group helped me a lot, especially Honglei Zhang, Jingbo Shang, Qi Zhu and Yu Meng.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

Aspect extraction (a.k.a. aspect discovery) refers to the task of identifying aspects of text segments from a pre-defined set of aspects. Being one of the fundamental tasks in text understanding, accurate aspect extraction benefits various downstream applications, including sentiment analysis and product review summarization. For instance, understanding aspects of a product's review sentences can help to deliver a holistic summary of this product without missing any important aspect.

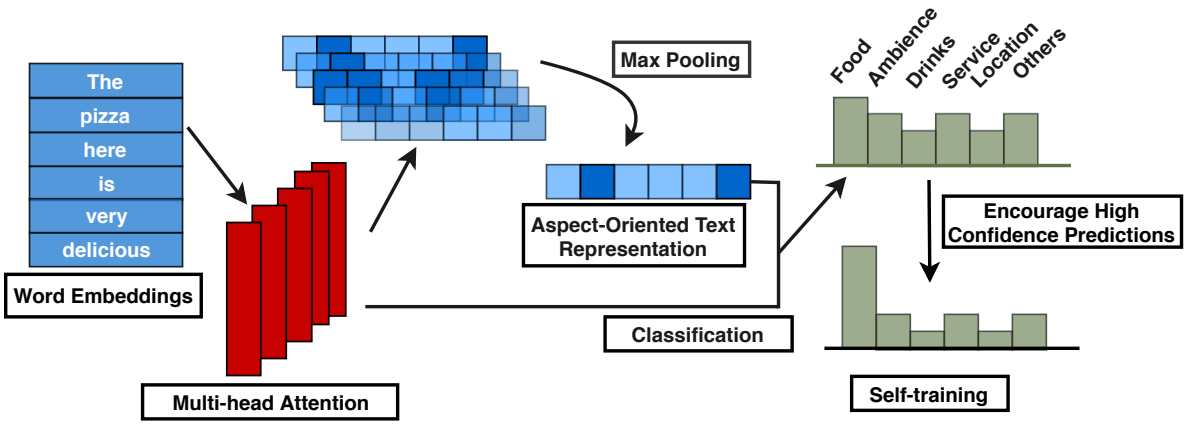
It is worth noting that the pre-defined aspects are highly domain-specific. For example, to extract aspects from restaurant reviews, the pre-defined aspects would be *Food*, *Service*, and *Price*; While aspects of smartphone reviews would be *Battery Life*, *Screen Size*, etc. The aspect extraction tool developed for one domain cannot be directly applied to another domain as they have totally different label spaces. Therefore, for every single domain, one has to develop a domain-specific aspect extraction tool.

Unfortunately, most of the existing aspect extraction methods are proposed within a supervised learning paradigm, which heavily rely on a large number of labeled data. Typically, labeled data of aspect extraction tasks have to be generated by extensive human effort, which makes such methods extremely expensive and time-consuming to deploy on a new domain. Although there are a few unsupervised studies, they also suffer from substantial shortcomings. Some [1] require carefully designed rules. Others [2, 3] can produce some clusters, but domain-specific knowledge and effort are required later when filtering, aggregating, and aligning such clusters to the pre-defined aspects.

Another challenge of this problem lies in how to handle the *misc* aspect. The *misc* aspect is designed to capture two types of noisy text segment: (1) text segments about some specific aspects out of the pre-defined scope, which are quite common in the real world, and (2) text segments talking nothing about any specific aspect (, "*This is one of my favorite restaurants.*"). Due to this noisy nature, even domain experts have difficulties to nominate seed words for the *misc* aspect.

In this paper, we explore to extract aspects or detect the *misc* aspect from the domain-specific raw texts with very limited supervision. We focus on short chunks of text segments, which contains at most one aspect [3, 2, 4]. Specifically, under this setting, the input contains raw texts and a few user-provided seed words for each aspect. For instance, given the pre-defined aspects in the restaurant review domain, the user only needs to provide a small set of seed words for each aspect (e.g., {"*food*", "*chicken*", "*steak*"} for the *Food* aspect and {"*server*", "*staffs*", "*waiter*"} for the *Service* aspect). Based on the user-provided

Figure 1.1: Graphical Illustration of the Weakly-Supervised Aspect Extraction Task and the Workflow of Our Model.



seed words and the corpus, we aim to train a model to identify the aspect of any (unseen) text segment in the same domain. For example, given a text segment "*The pizza here is delicious*", the model should be able to extract its aspect as *Food*.

Our problem, to some extent, is similar to the weakly-supervised text classification problem. There are also attempts to build document classification models based on the guidance of user-given seed words [5]. The major difference lies in the fact that the text segments in our problem are much shorter than the documents in the text classification problem, which makes our problem more challenging and requires special model designs. More comparisons can be found in our experiments.

We propose a novel neural model AutoAspect, for automatic aspect extraction. AutoAspect is equipped with multi-head attention and self-training, as shown in Figure 1.1. The multi-head attention is designed to emphasize the aspect-related words during aspect extraction. The basic idea to attack the problem is to capture the relationship between aspects and words. Inspired by recent works on label embedding based text classification [6], we embed both words and aspects in the same latent space and calculate the text-aspect compatibility based on cosine similarity. Such compatibility values are utilized for attention weights. Then we can obtain the aspect-oriented text representations and classify each of them into one of the user-given aspects or label it as *misc* by examining the confidence of the soft assignment. The self-training mechanism is employed to bootstrap more pseudo labels to better support the neural model's training. This module will put more focus on labels assigned with high confidence and iteratively refine the results.

Our major contributions are highlighted as follows.

- We explore to solve the domain-specific aspect extraction problem with

minimal supervision – only a small set of seed words per aspect.

- We design a novel neural model equipped with multi-head attention, aspect extraction with confidence thresholding and self-training, which are specially tailored to this problem setting.
- We conduct extensive experiments on three real-world datasets from different domains. The results show that our proposed model substantially outperforms existing methods in predicting aspects of text segments.

CHAPTER 2: PRELIMINARIES

In this section, we first present the notations and then formulate the problem rigorously.

2.1 NOTATIONS

We denote the given input corpus as a set of n documents $D = \{d_i\}_{i=1}^{n}$. Each document d_i can be further split into a sequence of text segments $d_i = \langle S_1, \ldots, S_{|d_i|} \rangle$, where $|d_i|$ represents the number of segments in the document d_i. Each text segment S consists of a sequence of tokens $S = \langle w_1, \ldots, w_{|S|} \rangle$, where $|S|$ is the number of tokens in this text segment. Please note that the terminology "token" here includes not only single-word words and punctuation, but also multi-word phrases (e.g., "battery life", "chocolate cake") and subword pieces (e.g., "n't" in "don't"). The tokens are pre-processed from raw texts by applying both tokenization and phrasal segmentation [7].

Let V be the vocabulary set of all possible tokens. For each token w in the vocabulary V, one can derive an embedding vector from word embedding technique ([8]). More precisely, the word embedding for each w, the word embedding for each w is a vector $e_w \in \mathbb{R}^{\nu \times 1}$, where ν is the number of dimensions in 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:

$$\mathrm{sim}\big(e_w, e_{w'}\big) = \frac{e_w \cdot e_{w'}}{\|e_w\| \times \|e_{w'}\|} \tag{2.1}$$

Therefore, the embedding representation matrix $X \in \mathbb{R}^{\nu \times |S|}$ of text segment S is constructed by concatenating each row vector. That is, $X = (e_{w_1}, \ldots, e_{w_{|S|}})$. In addition, there are K aspects $A_1, \ldots, A_K$ in the given domain. For each text segment S in the corpus, we use y_j to denote its aspect label. If S can be classified to one aspect, we set y_j as the aspect's identifier (i.e., $\subset \{1, \ldots, K\}$). If S is a general one without any specific aspect, we set y_j as $K + 1$.

In our problem setting, a text segment is a more fine-grained unit compared to a sentence. So we assume that each text segment can be classified to at most one aspect, which is generally true in the real-world data. In principle, a common pipeline should contain a classifier to determine whether a sentence is aspect-present followed by a classifier to perform the aspect extraction. Neither of the two sub-tasks have been studied in such a weakly-supervised scenario where users can only provide keywords. However, in this paper, we

focus on solving the two sub-tasks at the same time.

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

2.2 PROBLEM FORMULATION

Based on the above notations, we formulate our problem as follows.

The input of our problem contains two parts. First, an unlabeled review corpus D about a specific domain is 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. Second, we assume users have a relatively complete set of K aspects of interest in the given domain.

Users will provide some seed aspect words as guidance. Seed aspect words are small subsets of the vocabulary set V, i.e., $V_{A_1}, \ldots, V_{A_K}$.

Here is an example of the input.

> **Example (Input):** As illustrated in Figure 1.1, the given corpus is about the restaurant reviews. The pre-defined aspects are {*Food*, *Service*, *Ambience*, *Location*, *Drinks*}, corresponding to aspects A_1 to A_K respectively. Suppose A_1 is *Food*. Its seed words given by users are $V_{A_1} = \{$ *"food"*, *"chicken"*, *"appetizer"* $\}$. Similarly, the aspect A_2 *Service* has a set of seed words $V_{A_2} = \{$ *"server"*, *"staffs"*, *"waiter"*$\}$.

Notice that we do not require a ridiculously large set of seed words from users. In practice, two or three seed words per aspect should be able to produce satisfactory aspect extraction results. 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:** Given a corpus of review documents D, the pre-defined aspects $A_1, \ldots, A_K$, and seed aspect words $V_{A_1}, \ldots, V_{A_K}$, our problem is to build an aspect classifier for text segments. That is, for any input text segment S, the classifier can predict its corresponding aspect label a or tell us it focuses on none of the pre-defined aspects.

It is worth noting that this problem aims to perform *sentence-level* aspect analysis, which is a more challenging task from the *document-level* aspect-based sentiment analysis problem studied in [9, 5]. 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 analysis only with a few seed words provided by users as guidance, we can essentially perform document-level aspect analysis in the same manner.

CHAPTER 3: OUR MODEL

In this section, we present an overview of our proposed model and introduce the details about its three major components: (1) multi-head attention, (2) aspect extraction with confidence thresholding, and (3) self-training mechanism.

3.1 OVERVIEW

The first module, multi-head attention, creates attention heads for each aspect to construct the aspect-oriented text representations based on the word-aspect compatibility. For example, in the sample text segment "*The pizza here is delicious*", the learned text representation will emphasize more on words *pizza* and *delicious*, which are related to the aspect *Food*.

The second module, aspect extraction with confidence thresholding, formulates aspect extraction as a prediction problem where each text segment can be classified into one of the aspects based on the similarities between text representations and the aspect embedding. We also propose a confidence thresholding method to handle the text segments without mentioning any specific aspect.

The third component is a self-training mechanism adopted to train a better, more robust classifier. One can for sure build a classifier just based on the text representations from the attention module and the initial aspect embedding. Going beyond, we propose to improve the aspect extraction by two steps. Specifically, we train the model in a bootstrap manner — gradually adjusting the word and aspect embedding according to the model's own high-confidence prediction results.

3.2 MULTI-HEAD ATTENTION FOR ASPECT-ORIENTED REPRESENTATION

Our aspect extraction module features a multi-head attention mechanism where each attention head focuses on a specific aspect. It will assign different attention weights in different attention head for each aspect, where the aspect embedding serves as the query, and each word's embedding is used as both the key and the value. Then the attention weights in each head will be assigned by comparing how indicative a word is to the corresponding aspect. The outputs from all attention heads are finally aggregated to derive the prominent aspect of the text segment. The intuition behind is the fact that not all words contribute equally to identifying the aspect of a text segment. The multi-head attention mechanism

helps our model focus on aspect indicative words and ignore irrelevant ones, and obtaining aspect-oriented text representation.

Specifically, for each aspect A_i, we assume that there is a latent vector $a_i \in \mathbb{R}^{\nu \times 1}$ in the word embedding space that represents the semantics of the aspect. Higher embedding similarity between a word and an aspect implies the word is more closely related to the aspect and should be paid greater attention to.

Based on the above assumption, we initialize aspect embedding as the average word embedding of the user-provided aspect seed words. We compute the attention score between a word w and an aspect A_i as the cosine similarity between the word embedding and aspect embedding, , $\cos(a_i, e_w)$. The final attention score β_w in the text segment of word w will be its maximum attention score across all aspects, , $\max_{1 \leq i \leq K} \cos(a_i, e_w)$, and is normalized over the entire text segment via softmax, ,

$$\beta_w = \frac{\exp\left(\max_{1 \leq i \leq K} \cos(a_i, e_w)\right)}{\sum_{w' \in S} \exp\left(\max_{1 \leq i \leq K} \cos(a_i, e_{w'})\right)}. \tag{3.1}$$

Now we can obtain the aspect-oriented text representation $z \in \mathbb{R}^{\nu \times 1}$ for the text segment S as the weighted average of word embedding according to their attention weights:

$$z = \sum_{w \in S} \beta_w e_w. \tag{3.2}$$

By applying the same computation process, we will be able to get an aspect-oriented text representation z_i for each text segment S_i.

3.3 ASPECT EXTRACTION WITH CONFIDENCE THRESHOLDING

Once we have the aspect-oriented text representation z_i for each input text segment S_i and the aspect embedding a_j that represents the semantics of each aspect A_j, a soft assignment of text segments to aspects can be derived based on the similarity between text segment embedding z_i and aspect embedding a_j:

$$q_{ij} = \frac{\exp(\cos(a_j, z_i))}{\sum_{j'} \exp(\cos(a_{j'}, z_i))}. \tag{3.3}$$

If all text segments were known to mention exactly one of the aspects, then aspect extraction could be performed by simply classifying each text segment into its most relevant aspect, , $\arg\max_j q_{ij}$. However, as we mentioned in Chapter 2, it is common in real-world review

texts that one segment does not mention any of the aspects and therefore it is sub-optimal to force every text segment to be classified into one of the aspects. We want our model is able to detect text segments that should belong to *misc* aspect. In aspect extraction, two types of text segments belong to the *misc* aspect: (1) text segments about some specific aspects different from the K pre-defined aspects; and (2) text segments talking nothing about any specific aspects. These text segments are expected to have a relatively flat distribution in the predictions of the K-aspect classifier. Therefore, it is intuitive to leverage normalized entropy H_{norm}, which measures how chaotic the distribution is, to estimate the likelihood of S_i belonging to the *misc* aspect, i.e., P_{misc}. To tackle this issue, we propose the following confidence thresholding method to identify plain segments that do not mention any user-given aspects.

Specifically, we examine the confidence of the soft assignment in Equation (3.3) for text segment S_i using the normalized entropy metric:

$$H_{\text{norm}}(S_i) = -\frac{1}{\log K} \sum_{j=1}^{K} q_{ij} \log q_{ij}, \tag{3.4}$$

where the constant $-\frac{1}{\log K}$ normalizes the entropy value into the range $[0, 1]$.

Since entropy measures the uncertainty of the prediction, a higher $H_{\text{norm}}(S_i)$ indicates a less confident assignment of text segment S_i to any of the aspects. Indeed, when the text segment does not mention any of the aspects, the text segment embedding will be dissimilar with the embedding of all aspects, and q_{ij} will be close to a uniform distribution across all aspects, resulting in a high entropy value. Therefore, we use a threshold value γ ($0 \leq \gamma \leq 1$), which is a pre-defined hyperparameter, to identify plain text segments. Specifically, we mark a text segment S_i as plain (not mentioning any aspects) and avoid classifying them into any of the aspects when $H_{\text{norm}}(S_i) > \gamma$.

3.4 THE SELF-TRAINING MECHANISM

The results directly obtained from a soft assignment are not the best one can hope for, mainly for the following two reasons: (1) The attention embedding a_i's are initialized to be the average embedding of seed words but do not incorporate other aspect-relevant terms in the corpus. Thus, the quality of attention embedding could be biased towards user inputs; And (2) the initial word embeddings are generic representations encoding other properties of words that are not dedicated for aspect extraction. For example, the word *"delicious"* is a strongly indicative word for the aspect *"Food"*, but it also has sentiment

Algorithm 3.1: Overall Algorithm.

Input: A text collection $D = \{d_i\}|_{i=1}^{N}$ where $d_i = \langle S_1, \ldots, S_{|d_i|}\rangle$ can be split into several segments; seed aspect words $V_{A_1}, \ldots, V_{A_K}$ for each aspect.

Output: Aspect label of each text segment $y \in \{1, \ldots, K, K+1\}$ where $K+1$ represents the *NULL* aspect.

$\{e_w\} \leftarrow$ train unsupervised word embedding on D;
Initialize $a_i \leftarrow \frac{1}{|V_{A_i}|}\sum_{w' \in V_{A_i}} e_{w'}$;
Build model $M \leftarrow$ Equations (3.1), (3.2), (3.3);
$y', y \leftarrow$ randomly initialized such that $\Delta(y, y') > \rho\%$;
while $\Delta(y, y') > \rho\%$ **do**
    $y' \leftarrow y$;
    $Q \leftarrow$ Equation (3.3);
    $H \leftarrow$ Equation (3.4);
    $y \leftarrow$ threshold Q based on H;
    $P \leftarrow$ Equation (3.5);
    $M \leftarrow$ self-train according to Equation (3.6);
Return y;

polarity indication encoded in the unsupervised embedding representation. One can expect better aspect extraction performance by purifying word embedding to only contain aspect-indicative signals.

To address the aforementioned issues, we propose a self-training mechanism that makes full use of the unlabeled corpus to bootstrap our model by adjusting the attention embedding and word embedding for better aspect extraction performance. Self-training has been widely adopted in semi-supervised [10, 11] and weakly-supervised models [5]. The philosophy of self-training is bootstrapping the model by training on its own high-confident predictions in the previous iteration.

Specifically, during our self-training process, pseudo labels will be generated using the same formula as in [12]:

$$p_{ij} = \frac{q_{ij}^2/f_j}{\sum_{f_{j'}} q_{ij}^2/f_{j'}}, \tag{3.5}$$

where q_{ij} comes from Equation (3.3) and $f_j = \sum_i q_{ij}$ is the soft frequency for aspect j. The objective here is to minimize the KL divergence between soft-assignments Q and its corresponding pseudo labels P:

$$L = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{3.6}$$

This process will terminate when less than $\rho\%$ of the text segments in the corpus change their aspect assignments, and then the aspect prediction results will be the final one of the model. Here, ρ is a pre-defined hyper-parameter.

3.5 ALGORITHM SUMMARY

Algorithm 3.1 summarizes the overall training of the model. The model is first initialized with unsupervised word embeddings and averaged seed embeddings as attention weights. After learning the aspect-oriented text embedding by utilizing the multi-head attention module, we can perform K+1 classification with confidence thresholding. Then the model will be updated iteratively via self-training.

CHAPTER 4: EXPERIMENTS

In this section, we evaluate the empirical performance of our method for weakly supervised aspect extraction.

4.1 DATASETS

We conduct experiments on two real-world datasets to evaluate the performance of our proposed model. Table 4.1 presents you some statistics.

- **Restaurant:** For training, we have collected 16,061 unlabeled restaurant reviews from a public Yelp dataset[1]. For evaluation, we utilize reviews from SemEval-2016 [13] in the restaurant domain as ground-truth. These reviews are labeled with target entities, which are regarded as aspect types. There are in totally 5 aspects in this dataset: *Food*, *Service*, *Ambience*, *Price*, and *Location*.
- **Laptop:** For training, we are using 14,683 unlabeled Amazon reviews on laptop, collected by [14, 15]. For evaluation, we utilize labeled reviews on the laptop domain from SemEval 2016 [13]. There are originally 21 different entity types. In our experiments, the pre-defined aspect set contains the top-8 popular entity types as aspects. Specifically, they are *Support*, *OS*, *Display*, *Battery*, *Company*, *Mouse*, *Software*, and *Keyboard*.

On both datasets, we use the class label *NULL* to denote the text segments without mentioning any specific aspect in the pre-defined aspect set.

4.2 COMPARED METHODS

We compare our model with a wide range of baseline models, described as follows.

- **Cosine Similarity.** It assigns the most similar aspect to each text segmentation according to the cosine similarity between the average embedding of all words in the given text segment and the average embedding of all seed words of each aspect.

[1]https://www.yelp.com/dataset/challenge

Table 4.1: Dataset Statistics

Dataset	Unlabeled Segments	Test Segments
Restaurant	16,061	1,166
Laptop	14,683	780

- **ABAE** [3]. The original ABAE model is an unsupervised neural topic model. To start with, it utilizes an attention mechanism to construct new text segment embedding. Then, it will learn the aspect dictionary via an auto-encoder framework. We extend the ABAE by utilizing user-provided seed aspect words to guide the dictionary learning process.
- **MATE** [4]. It is an extended version of ABAE, which accepts seed information for guidance and replaces ABAE's aspect dictionary with seed matrices.
- **WeSTClass** [5]. This is a weakly supervised text classification model, which accept seed words as supervision as well. It first generates pseudo-documents by leveraging seed information and then use a self-training module to refine the model.
- **Dataless** [9]. This method accepts aspect names as supervision and leverages Wikipedia and Explicit Semantic Analysis (ESA) to derive vector representation of both aspects and documents. The class is assigned based on the vector similarity between aspects and documents.
- **BERT** [16] is the recent pre-trained language model, presenting state-of-the-art performance in a wide variety of classic NLP tasks. Under the weak supervision setting, we use seed words matching to generate sentence labels. If there are multiple seed words in one sentence, we assign the aspect label based on majority voting. Then we fine-tune BERT under the supervised text classification setting.
- **AutoAspect, No-Threshold**. This is a variant of our model without the confidence thresholding method. That is, we force our proposed model to assign exactly one aspect to each text segmentation. The *NULL* aspect handling part is dropped in this ablated version.
- **AutoAspect, No-Self-Train**. This is a variant of our model without the self-training module.
- **Best+Threshold**. Since none of the above baseline methods can handle

Table 4.2: Example Seed Words for the Restaurant Dataset.

Aspect	**Seed Word List**
Location	street, convenient, block, avenue, river, subway, neighborhood, downtown, bus
Drinks	drinks, beverage, wines, margaritas, sake, beer, wine list, cocktail, vodka, soft drinks
Food	food, spicy, sushi, pizza, tasty, steak, delicious, bbq, seafood, noodle
Ambience	romantic, atmosphere, room, seating, small, spacious, dark, cozy, quaint, music
Service	tips, manager, wait, waitress, servers, fast, prompt, friendly, courteous, attentive

NULL aspect, for fair comparison, we append our confidence thresholding method to each of the baseline models in order for them to identify *NULL* aspect. We report the best performances among all baselines with the confidence thresholding method.

- **AutoAspect**. This is the full version of our proposed framework, with both the self-training module and the confident thresholding method.

4.3 EXPERIMENT SETUP

Pre-processing. In both datasets, unlabeled review documents serve as the training data and labeled text segments serve as test data, we use NLTK[2] to tokenize them into a list of words. Then we apply phrase mining [7], to discover phrases like "caesar salad" and "hard drive" so that these phrases will be treated as single semantic unit.

We train word2vec [8] on our training corpus and obtain embeddings for words and phrases. Notice that our method does not rely on any specific word embedding so that it can seamlessly adapt to any other pre-trained embedding as well.

User-Provided Seed Words. For both datasets, three professional annotators are asked to read the unlabeled corpus and write down 10 seed words for each aspect. Then we will test our model based on these three sets of user-provided seed words separately and report the average of the test result. These three sets of seed words will also be used in the baseline models. Table 4.2 shows the seed word list provided by one annotator for the Restaurant

[2]https://www.nltk.org/

dataset. By default, we will randomly choose 5 seed words from them to train the model. It is worth noting that the seed words in both datasets are very diverse. Although the seed words picked by the annotators in our paper seem to be representative, about 73% of the sentences in Restaurant dataset do not contain any seed words in the list, and this number rises to 74% for Laptop dataset.

Configurations. We set the number of latent dimensions $\nu = 200$, the plain text segment threshold value $\gamma = 0.9$ and the self-training terminating criteria $\rho = 0.001$.

Evaluation Metrics. We evaluate the performance by Accuracy (Acc), Precision (Prec), Recall and F_1 score. To clarify, we employ macro-averaged precision, macro-averaged recall and macro-averaged F_1 score as the evaluation metrics.

Based on each seed word list provided by three annotators, we run the experiments 10 times for each method on each data set and report the average performance to reduce the effect of randomness.

4.4 EXPERIMENTAL RESULTS

In this section, we present and discuss the experimental results of all methods on the two datasets.

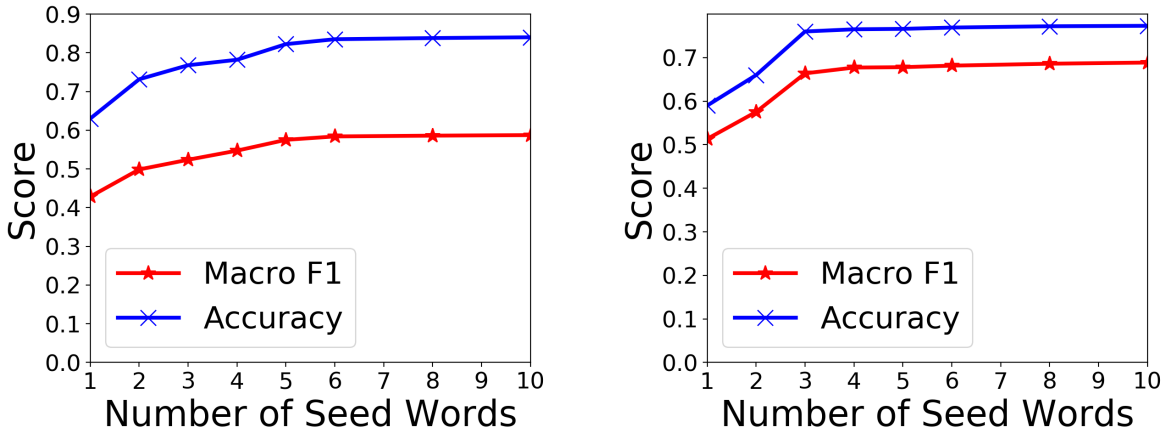
4.4.1 Performance Comparison.

In the first set of experiments, we compare the aspect extraction performance of our method against all the baseline methods on both datasets. As shown in Table 4.3, our proposed framework achieves the overall best performance among all the baselines on two datasets.

The simple baseline CosSim achieves a competitive accuracy of 59% on the Restaurant dataset, but only reaches an accuracy of 49% on the Laptop dataset, which has more aspects. It is in line with our expectation because the average word embedding is effective for short text segments to tell different aspects. However, when there are more aspects, we can not capture the subtle difference only relying on the embedding of words.

It is not surprising to see weakly supervised models achieve good performance. On the Restaurant dataset, the second best method is MATE, while on the Laptop dataset, the second best method is WeSTClass. However, they are not designed to handle the *NULL* aspect, because no one can give accurate seed words for this *NULL* aspect. Even compared with the variant No-Threshold of our model, which does not handle the *NULL* aspect too, these weakly supervised models still perform much worse. Surprisingly, pre-trained neural

Figure 4.1: Performance study with varying number of seed words.



(a) Restaurant (b) Laptop

language model (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.

In order to deliver a more fair comparison, we further applied our confidence thresholding technique to the best baseline method on each dataset. The best baseline methods are MATE and WeSTClass on the Restaurant and Laptop datasets, respectively. Looking at the results in Table 4.3, there are still significant margins between our model and the enhanced best baseline methods. More importantly, if one checks the improvements of plugging in the confidence thresholding technique, it is obvious that this technique fits our model better, thus demonstrating more improvements with our proposed model.

Moreover, our model consistently beat its variant without the self-training mechanism on both datasets. Therefore, we conclude that self-training can help us to further boost the aspect extraction accuracy.

4.4.2 Performance study with different seed word lists

The selection of seed words is another crucial part of our model. We want to find out how our model will be influenced when given different annotator-picked seed word lists. Therefore we do additional experiments with two different seed word lists in the Restaurant dataset. Firstly, we use the same annotator-picked seed word list as shown in Table 4.2. Secondly, we ask the annotator to pick another seed word list that is totally different from the first one. For both seed word lists, we randomly pick 3 seed words for each aspect and report the

Table 4.3: Empirical Evaluation of Aspect Extraction Performance. The scores are all percentages (%).

Dataset	Method	Acc	Prec	Recall	F_1
Restaurant	CosSim	59.00	54.55	47.82	49.85
	Dataless	45.47	52.25	44.67	42.65
	WeSTClass	52.36	61.53	52.59	48.72
	ABAE	60.51	54.94	49.04	51.12
	MATE	62.56	56.13	51.27	51.77
	BERT	54.55	59.55	52.85	47.51
	No-Threshold	67.83	65.49	51.74	52.88
	No-Self-Train	66.98	64.85	43.88	49.08
	Best+Threshold	64.56	58.64	53.73	52.56
	AutoAspect	**69.81**	**67.90**	**57.77**	**57.50**
Laptop	CosSim	49.60	59.96	53.64	50.49
	Dataless	53.2	55.46	56.34	55.04
	WeSTClass	62.49	64.41	65.22	63.71
	ABAE	56.53	60.07	59.88	57.21
	MATE	60.48	61.05	62.99	61.20
	BERT	56.20	59.49	56.72	54.12
	No-Threshold	65.84	66.01	60.29	63.16
	No-Self-Train	66.43	69.65	66.11	66.75
	Best+Threshold	63.04	66.58	66.77	65.81
	AutoAspect	**67.49**	**70.64**	**66.97**	**67.80**

average accuracy over 10 runs. Due to space limitations, we use seed words from *Ambience* as an example. From Table 4.4, we can see that our model achieves comparable performance under two totally different seed word lists.

4.4.3 Performance study with varying number of seed words

Seed words are important parts of the input to our model. Evaluating the sensitivity of the number of seed words would be interesting. So we conduct experiments to check the performance of our proposed model under different numbers of given seed words per aspect. Given a list of seed words provided by a certain annotator, we randomly select n seed words for each aspect, where n varies from $\{1, 2, 3, 4, 6, 8, 10\}$. Again, on each dataset, we run our model 10 times with the three lists of seed words and report the average performance with macro-F_1 and accuracy scores. From Figure 4.2(a), we can see that on the Restaurant

Table 4.4: Performance on two seed word lists

Seed Word List 1	Acc	Seed Word List 2	Acc
room, romantic, atmosphere	0.629	mood, vibe, aroma	0.615
cozy, dark, spacious	0.630	semblance, style, surroundings	0.626
quaint, small, music	0.620	circumstance, ambient, layout	0.618

dataset, the performance of our model improves significantly when n is less than 5 and gradually becomes stable after that. The similar trend can be observed on the Laptop dataset, as shown in Figure 4.2(b). In most real-world cases, it is not burdensome for any user to provide 5 seed words per aspect. Therefore, we believe that our model should be effective in the real world with the limited amount of seed information.

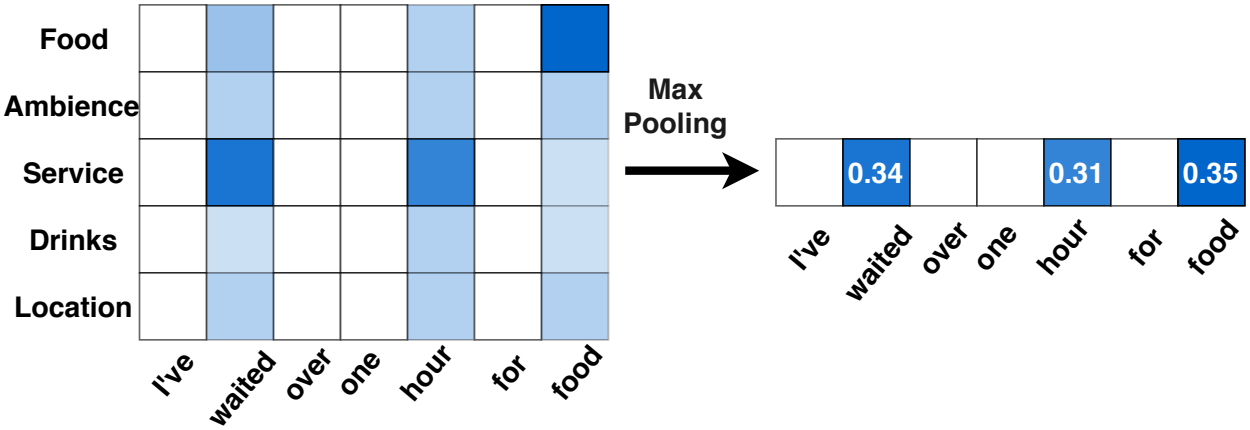
4.4.4 Misc Text Segment Examples

We present two successfully classified text segments of the different types of misc aspect.

The first example is from the Restaurant dataset, "There is nothing more pleasant than that.". This text segment does not talk about any specific aspect and it can refer to *service* or *ambience*. Eventually, predicts the probabilities of this segment belong to *misc*, *service*, and *ambience* as 0.38, 0.29, and 0.25 respectively. Therefore *misc* wins in the end.

The second example is from the Laptop dataset: "the only problem is that i had to add 1 gb RAM, the computer was kinda slow.", about the out-of-pre-defined *hardware* aspect. predicts it as *misc* and *os* with chances 0.47 and 0.19 respectively, mainly because the word "slow" is widely used to complain about OS.

Figure 4.2: Case Studies about Multi-Head Attentions.



4.5 CASE STUDY

We would like to confirm the effectiveness of the multi-head attention in the case study. Figure 4.2 shows the attention weights generated from our model on one interesting text segment: "*I've waited over one hour for food*". It contains three aspect-related words: "food", "waited" and "hour". After Max Pooling, our multi-head attention gives comparative scores to these three words. It is in line with our expectation because even if "food" is a seed word for *Food* aspect, it does not get too much weight since "waited" and "hour" are both very related to *Service* aspect as well.

This example shows that our multi-head attention is able to focus only on aspect indicative words and mitigate the effect from irrelevant ones.

CHAPTER 5: RELATED WORK

Aspect extraction was originally a task focusing on extracting aspects for each document. Rule-based methods [1, 17, 18, 19, 20, 21] are the pioneers along this direction. All these methods reply on either pre-defined rules carefully designed by human experts or the assumption that the aspect terms should frequently appear in the corpus. Therefore, this type of methods has several limitations that are hard to be adapted to new domains.

Later, researchers started to utilize unsupervised methods to extract possible aspects for each document. Traditional unsupervised methods are mostly based on the LDA topic model and its variants [22, 23, 24, 25] by treating extracted topics as aspects. Mixed models, such as LDA-IG [26] and ELDA [27], are further proposed to add the word co-occurrence information into the topic models. More recently, a neural model ExtRA [2] is proposed to further improve the aspect extraction at the document level. However, as our problem setting focuses on a much shorter unit (i.e., text segment) than a full document, these models are no longer applicable here.

Recently, unsupervised neural models estimating the aspects for each text segment have been developed, such as ABAE [3]. ABAE employs an attention module to learn embedding for text segments and an auto-encoder framework to build aspect dictionaries. However, ABAE cannot produce end-to-end aspect results as output. It requires users to first set the number of topics as a much larger number than the number of desired aspects, and then manually map the extracted topics back to the aspects. Our problem setting is designed to avoid such significant burdens posed on users. Building upon ABAE, Angelidis and Lapata [4] further proposed a multi-seed aspect extractor MATE using seed aspect words as guidance. This model keeps the human effort at a minimal degree and fits our problem setting well. However, even with its multi-task counterpart, the reconstruction objective in MATE model is not able to provide adequate training signals. Our proposed method leverages the self-training mechanism to overcome this issue, thus outperforming MATE significantly in extensive experiments.

Our problem can also be viewed as a weakly-supervised text classification problem. Existing methods can build document classifiers by taking either hundreds of labeled training documents [28, 29, 30], class/category names [9, 31], or user-provided seed words [5] as the source of weak supervision. Because the text segments in our problem setting are mostly short, while these models are good at handling document-level classification, their performance on text segments becomes not satisfactory. Moreover, all these methods assume that users can always provide seeds for all classes, while overlooking the noisy *misc* aspect in our

problem. We incorporate the *misc* aspect systematically into our framework.

Supervised feature-based methods treat the aspect extraction as a sequence labeling problem in sentence-level. Relying on hand-curated features to enhance performance, traditional sequential models like hidden Markov models [32] and Conditional Random Fields based models [33, 34] are proposed. Later, as neural networks and representation learning become popular, some methods were proposed to extract aspects by automatically learning features in CRF [35, 36, 37]. Despite the success of the sequential models, they could still be easily affected by the selection of features. Thus recently a number of deep-learning based models are proposed like LSTM-based approaches[38, 39, 40]and CNN-based approaches[41, 42]. Supervised models mainly perform on document/sentence level aspect extraction and a significant number of labeled sentences are required to train the supervised models.

CHAPTER 6: CONCLUSIONS AND FUTURE WORK

In this paper, we explore to build an aspect extraction model for text segments using only a few user-provided seed words per aspect. We develop a novel neural model with specially designed multi-head attention, aspect extraction with confidence thresholding and self-training. The multi-head attention is able to locate the aspect-related words in each text segment while the aspect extraction with thresholding module is able to detect pre-defined aspects and *misc* aspect. In addition, the self-training generates more "supervision" from the most confident model predictions. Extensive experiments have demonstrated the effectiveness of our proposed model. Ablation studies and case studies have verified the intuition and expectation of our model designs.

In the future, we would like to integrate the extracted aspect information with downstream tasks, such as sentiment analysis and opinion summarization. Building a unified optimization framework for both aspect extraction and the downstream tasks would be another interesting direction to move. With the downstream applications, more supervision could be available and one can then build better aspect extraction models.

REFERENCES

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

[2] Z. Luo, S. Huang, F. F. Xu, B. Y. Lin, H. Shi, and K. Zhu, "Extra: Extracting prominent review aspects from customer feedback," in *EMNLP*, 2018, pp. 3477–3486.

[3] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier, "An unsupervised neural attention model for aspect extraction," in *ACL*, 2017, pp. 388–397.

[4] S. Angelidis and M. Lapata, "Summarizing opinions: Aspect extraction meets sentiment prediction and they are both weakly supervised," *arXiv:1808.08858*, 2018.

[5] Y. Meng, J. Shen, C. Zhang, and J. Han, "Weakly-supervised neural text classification," in *CIKM*, 2018, pp. 983–992.

[6] G. Wang, C. Li, W. Wang, Y. Zhang, D. Shen, X. Zhang, R. Henao, and L. Carin, "Joint embedding of words and labels for text classification," *arXiv:1805.04174*, 2018.

[7] J. Shang, J. Liu, M. Jiang, X. Ren, C. R. Voss, and J. Han, "Automated phrase mining from massive text corpora," *TKDE*, vol. 30, no. 10, pp. 1825–1837, 2018.

[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NeurIPS*, 2013.

[9] Y. Song and D. Roth, "On dataless hierarchical text classification," in *AAAI*, 2014.

[10] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training," in *CIKM*, vol. 5, 2000, p. 3.

[11] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-supervised self-training of object detection models." *WACV/MOTION*, vol. 2, 2005.

[12] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *ICML*, 2016, pp. 478–487.

[13] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, A.-S. Mohammad, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq et al., "Semeval-2016 task 5: Aspect based sentiment analysis," in *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, 2016, pp. 19–30.

[14] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *SIGIR*, 2015.

[15] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *WWW*, 2016.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv:1810.04805*, 2018.

[17] B. Liu, M. Hu, and J. Cheng, "Opinion observer: analyzing and comparing opinions on the web," in *WWW*, 2005, pp. 342–351.

[18] L. Zhuang, F. Jing, and X.-Y. Zhu, "Movie review mining and summarization," in *CIKM*, 2006, pp. 43–50.

[19] C. Scaffidi, K. Bierhoff, E. Chang, M. Felker, H. Ng, and C. Jin, "Red opal: product-feature scoring from reviews," in *Proceedings of the 8th ACM conference on Electronic commerce*, 2007, pp. 182–191.

[20] L. Zhang, B. Liu, S. H. Lim, and E. O'Brien-Strain, "Extracting and ranking product features in opinion documents," in *Proceedings of the 23rd international conference on computational linguistics: Posters*. Association for Computational Linguistics, 2010, pp. 1462–1470.

[21] G. Qiu, B. Liu, J. Bu, and C. Chen, "Opinion word expansion and target extraction through double propagation," *Computational linguistics*, vol. 37, no. 1, pp. 9–27, 2011.

[22] I. Titov and R. McDonald, "Modeling online reviews with multi-grain topic models," in *WWW*. ACM, 2008, pp. 111–120.

[23] W. X. Zhao, J. Jiang, H. Yan, and X. Li, "Jointly modeling aspects and opinions with a maxent-lda hybrid," in *EMNLP*, 2010, pp. 56–65.

[24] S. Brody and N. Elhadad, "An unsupervised aspect-sentiment model for online reviews," in *NAACL*, 2010, pp. 804–812.

[25] A. Mukherjee and B. Liu, "Aspect extraction through semi-supervised modeling," in *ACL*, 2012, pp. 339–348.

[26] C. Zhang, H. Wang, L. Cao, W. Wang, and F. Xu, "A hybrid term–term relations analysis approach for topic detection," *Knowledge-Based Systems*, vol. 93, pp. 109–120, 2016.

[27] M. Shams and A. Baraani-Dastjerdi, "Enriched lda (elda): combination of latent dirichlet allocation with word co-occurrence analysis for aspect extraction," *Expert Systems with Applications*, vol. 80, pp. 136–146, 2017.

[28] J. Tang, M. Qu, and Q. Mei, "Pte: Predictive text embedding through large-scale heterogeneous text networks," in *SIGKDD*, 2015, pp. 1165–1174.

[29] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification," *arXiv:1605.07725*, 2016.

[30] W. Xu, H. Sun, C. Deng, and Y. Tan, "Variational autoencoder for semi-supervised text classification," in *AAAI*, 2017.

[31] K. Li, H. Zha, Y. Su, and X. Yan, "Unsupervised neural categorization for scientific publications," in *SIAM Data Mining*. SIAM, 2018, pp. 37–45.

[32] W. Jin, H. H. Ho, and R. K. Srihari, "A novel lexicalized hmm-based learning framework for web opinion mining," in *ICML*, 2009, pp. 465–472.

[33] F. Li, C. Han, M. Huang, X. Zhu, Y.-J. Xia, S. Zhang, and H. Yu, "Structure-aware review mining and summarization," in *Proceedings of the 23rd international conference on computational linguistics*. Association for Computational Linguistics, 2010, pp. 653–661.

[34] N. Jakob and I. Gurevych, "Extracting opinion targets in a single-and cross-domain setting with conditional random fields," in *EMNLP*, 2010, pp. 1035–1045.

[35] W. Wang, S. J. Pan, D. Dahlmeier, and X. Xiao, "Recursive neural conditional random fields for aspect-based sentiment analysis," *arXiv:1603.06679*, 2016.

[36] Y. Yin, F. Wei, L. Dong, K. Xu, M. Zhang, and M. Zhou, "Unsupervised word and dependency path embeddings for aspect term extraction," *arXiv:1605.07843*, 2016.

[37] Y. Xiang, H. He, and J. Zheng, "Aspect term extraction based on mfe-crf," *Information*, vol. 9, no. 8, p. 198, 2018.

[38] X. Li and W. Lam, "Deep multi-task learning for aspect term extraction with memory interaction," in *EMNLP*, 2017, pp. 2886–2892.

[39] X. Li, L. Bing, P. Li, W. Lam, and Z. Yang, "Aspect term extraction with history attention and selective transformation," *arXiv:1805.00760*, 2018.

[40] J. Yu, J. Jiang, and R. Xia, "Global inference for aspect and opinion terms co-extraction based on multi-task neural networks," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 27, no. 1, pp. 168–177, 2019.

[41] S. Poria, E. Cambria, and A. Gelbukh, "Aspect extraction for opinion mining with a deep convolutional neural network," *Knowledge-Based Systems*, vol. 108, pp. 42–49, 2016.

[42] H. Xu, B. Liu, L. Shu, and P. S. Yu, "Double embeddings and cnn-based sequence labeling for aspect extraction," *arXiv:1805.04601*, 2018.