

# User-Guided Aspect Classification for Domain-Specific Texts

**Peiran Li<sup>1</sup>**

**Fang Guo<sup>3</sup>**

**Jingbo Shang<sup>1,2</sup>**

<sup>1</sup> Department of Computer Science and Engineering, University of California San Diego, CA, USA

<sup>2</sup> Halıcıoğlu Data Science Institute, University of California San Diego, CA, USA

<sup>3</sup> Department of Computer Science, University of Illinois at Urbana-Champaign, IL, USA

<sup>1,2</sup>{pel047, jshang}@ucsd.edu    <sup>3</sup>fangguo1@illinois.edu

## Abstract

Aspect classification, identifying aspects of text segments, facilitates numerous applications, such as sentiment analysis and review summarization. To alleviate the human effort on annotating massive texts, in this paper, we study the problem of classifying aspects based on only a few user-provided seed words for pre-defined aspects. The major challenge lies in how to handle the noisy “*misc*” aspect, which is designed for texts without any pre-defined aspects. Even domain experts have difficulties to nominate seed words for the “*misc*” aspect, making existing seed-driven text classification methods not applicable. We propose a novel framework, ARYA, which enables mutual enhancements between pre-defined aspects and the “*misc*” aspect via iterative classifier training and seed updating. Specifically, it trains a classifier for pre-defined aspects and then leverages it to induce the supervision for the “*misc*” aspect. The prediction results of the “*misc*” aspect are later utilized to filter out noisy seed words for pre-defined aspects. Experiments in two domains demonstrate the superior performance of our proposed framework, as well as the necessity and importance of properly modeling the “*misc*” aspect.

## 1 Introduction

Aspect classification is a fundamental task in text understanding, aiming at identifying aspects of text segments (He et al., 2017). It can facilitate 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 (Angelidis and Lapata, 2018).

Following the supervised paradigm to extract aspects requires extensive human effort on annotating massive domain-specific texts, because aspects

vary across domains. For example, in restaurant reviews, possible aspects include “*food*”, “*service*”, and “*location*”. When it comes to laptop reviews, aspects become “*battery*”, “*display*”, etc. Therefore, to alleviate such effort, we study the problem of *user-guided aspect classification*, which only relies on very limited supervision — only a small number (e.g., 5) of seed words per aspect.

The major challenge of this problem lies in how to handle the “*misc*” aspect. The “*misc*” aspect is designed to capture two types of text segments which makes it noisy: (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 (e.g., “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, making existing seed-driven text classification methods (Agichtein and Gravano, 2000; Riloff et al., 2003; Kuipers et al., 2006; Tao et al., 2015; Meng et al., 2018) not applicable here. In this paper, we aim to better incorporate the “*misc*” aspect into user-guided aspect extraction.

We make two intuitive, crucial observations, which shed light on the development of our proposed framework. First, given a text segment, if its distribution over pre-defined aspects is flat, it likely belongs to the “*misc*” aspect. This provides us a chance of inducing “*misc*”-aspect supervision from the classifier trained for pre-defined aspects. Second, given a word, if it is a strong indicator of the “*misc*” aspect, it is unlikely to be a good seed word of any pre-defined aspect. Excluding such words from the seed words of pre-defined aspects would reduce ambiguity, thus becoming a wise decision.

Acknowledging these observations, we propose a novel framework incorporating the “*misc*” aspect in a systematic manner. As shown in Figure 1, it is an iterative framework, alternatively training the

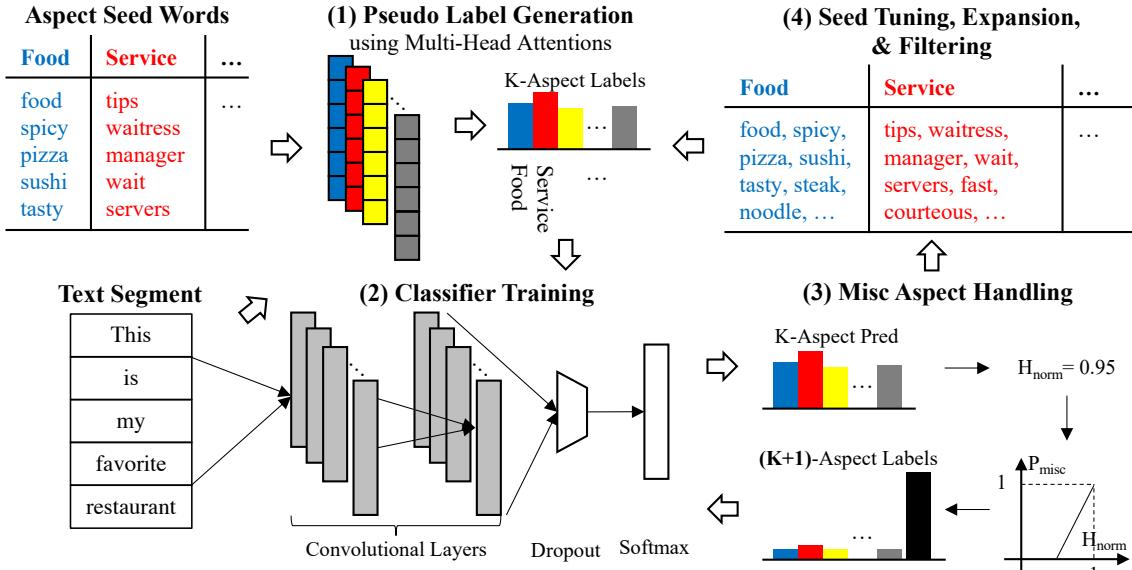


Figure 1: Overview of our proposed framework ARYA. It enables mutual enhancements between the pre-defined aspects and the “*misc*” aspect via iterative classifier training and seed updating. Pre-defined aspects help to induce supervision for the “*misc*” aspect; The “*misc*” aspect helps to filter out noisy seed words for pre-defined aspects.

classifier for all aspects and updating seed words of pre-defined aspects. We name it as ARYA.<sup>1</sup> More specifically, we first train a classifier for  $K$  pre-defined aspects based on user-provided seed words. This  $K$ -aspect classifier further induces supervision for the “*misc*” aspect based on normalized entropy estimation, enabling a  $(K+1)$ -aspect classifier. This  $(K+1)$ -classifier facilitates our comparative analysis, which updates seed words of pre-defined aspects using strong aspect-indicative words. The predicted “*misc*” aspect information is utilized to ensure those noisy words will never appear in seed words of pre-defined aspects. As one can see here, ARYA achieves mutual enhancements between pre-defined aspects and the “*misc*” aspect.

To our best knowledge, we are the first to systematically handle the “*misc*” aspect in user-guided aspect extraction. Our main contributions are:

- We identify the keystone towards user-guided aspect extraction as the noisy “*misc*” aspect.
- We develop ARYA based on two intuitive observations, making pre-defined aspects and the “*misc*” aspect mutually enhance each other.
- Experiments in two domains demonstrate the superiority of ARYA and the necessity of considering the “*misc*” aspect systematically.

**Reproducibility.** We will release our code and datasets in our GitHub repository<sup>2</sup>.

<sup>1</sup>Our framework is named after *Arya Stark* in Game of Thrones, who kills the Night King, bringing an end to the *Others* (i.e., White Walkers, wights, etc.) forever.

<sup>2</sup><https://github.com/peiranli/ARYA>

## 2 Overview

**Problem Formulation.** Given a domain-specific corpus  $\mathcal{D}$  of  $n$  text segments  $\{S_1, S_2, \dots, S_n\}$ ,  $K$  pre-defined aspects  $\{A_1, \dots, A_K\}$ , and a small number of seed words per aspect  $\{V_{A_1}, \dots, V_{A_K}\}$ , this paper aims to build an aspect classifier for domain-specific text segments. A domain here refers to a relatively consistent category of products or services, such as the *hotel* domain, the *restaurant* domain, and the *laptop* domain.

In this paper, we assume that there is at most one specific aspect in each text segments. In practice, one can always segment the text in a fine-grained way to ensure that this assumption holds. In other words, for any input text segment  $S_i$ , our classifier aims to predict its corresponding aspect label  $y_i$ .  $y_i$  is either an ID of the pre-defined aspects (between 1 and  $K$ ) or the number  $K+1$  denoting that  $S_i$  focuses on none of the pre-defined aspects.

**Our Framework.** ARYA is an iterative framework as illustrated in Figure 1 and Algorithm 1. In each iteration, we apply the following four steps in order.

- **Pseudo Label Generation.** Given seed words for  $K$  aspects, we generate  $K$ -aspect pseudo labels for all text segments in the raw corpus.
- **Classifier Training.** We train a  $K$ -aspect classifier based on the generated pseudo labels. Our framework is compatible with all text classifiers. As an illustration, we choose to use 1-D CNN in this paper. We will brief its neural architecture for the self-contained purpose.

---

**Algorithm 1:** Overall Algorithm

---

**Input:** A corpus  $\mathcal{D}$  of  $n$  text segments  $\{S_1, S_2, \dots, S_n\}$ , user-provided seed words for  $K$  pre-defined aspects  $\{V_{A_1}, \dots, V_{A_K}\}$ .  
**Output:** A  $(K+1)$ -aspect classifier.  
Train word embedding  $\mathbf{e}_w$  on  $\mathcal{D}$ .  
**while** Seed words are not converged **do**  
    Compute aspect embedding  $\mathbf{a}_j$  (Eq 1)  
    Get K-aspect supervision  $q_{i,j}$  (Eq 4)  
    Train K-aspect classifier  $M_K$  (Sec 4)  
    Get  $(K+1)$ -aspect supervision  $\hat{q}_{i,j}$  (Eq 7)  
    Tune, expand, and filter seed words (Sec 6)  
**Return** The last  $(K+1)$ -aspect classifier.

---

- **Misc Aspect Handling.** We leverage the predictions of the trained  $K$ -aspect classifier to produce pseudo labels for the “*misc*” aspect. After that, we train a new  $(K+1)$ -aspect classifier, which makes an end-to-end aspect extraction.
- **Seed Tuning, Expansion, and Filtering.** We conduct a comparative analysis to compare and contrast the text segments projected to different aspects to find new and discriminative seed words for each aspect. The “*misc*” aspect is utilized here to further filter out noisy seed words for pre-defined aspects.

We will discuss the details of the four major components in the following sections. Before that, here are some basic notations.

**Notations.** Each text segment consists of a sequence of tokens, i.e.,  $S_i = \langle w_1, \dots, w_{|S_i|} \rangle$ , where  $|S_i|$  is the number of tokens in  $S_i$ . Please note that “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”). The tokens are pre-processed from raw texts by applying both tokenization and phrasal segmentation (Shang et al., 2018).

Let  $V$  be the vocabulary set of all tokens. For each token  $w \in V$ , we denote its  $d$ -dimensional embedding vector as  $\mathbf{e}_w \in \mathbb{R}^{d \times 1}$ . The embedding representation matrix of text segment  $S_i$  is then defined as  $\mathbf{X}_i = (\mathbf{e}_{w_1}, \dots, \mathbf{e}_{w_{|S_i|}}) \in \mathbb{R}^{d \times |S_i|}$  by concatenating each row vector.

### 3 Pseudo Label Generation

We generate pseudo labels following a multi-head attention mechanism, where each attention head focuses on a specific aspect. It helps our model focus on aspect indicative words and ignore irrel-

evant ones, and derive aspect-oriented representation. The outputs from all attention heads are finally aggregated to derive the prominent aspect of the text segment.

First, we assume that the user-provided seed words can characterize the aspect’s semantics. So we compute  $\mathbf{a}_j$ , the aspect representation of  $A_j$ , by averaging embedding of its seed words.

$$\mathbf{a}_j = \frac{\sum_{w \in V_{A_j}} \mathbf{e}_w}{|V_{A_j}|} \quad (1)$$

A higher embedding similarity between a word and an aspect implies that the word is more closely related to the aspect, and it should be paid greater attention to. Therefore, given a word  $w$ , its attention weight is defined as its maximum similarity over  $K$  aspects.

$$\beta_w = \max_{j=1}^K \{\mathbf{a}_j^T \mathbf{e}_w\} \quad (2)$$

Since text segments are usually short, we use the average of its tokens following the attention weights as its aspect-oriented representation  $\mathbf{z}_i$ .

$$\mathbf{z}_i = \frac{\sum_{w \in S_i} \beta_w \cdot \mathbf{e}_w}{\sum_{w \in S_i} \beta_w} \quad (3)$$

Based on the similarity between text segment representation  $\mathbf{z}_i$  and aspect representation  $\mathbf{a}_j$ , we derive the pseudo label assignments as

$$q_{i,j} \propto \exp(\mathbf{a}_j^T \mathbf{z}_i) \quad (4)$$

We normalize  $q_{i,*}$  into a label distribution over all  $K$  aspects.

### 4 Aspect Classifier Training

Our framework is generally compatible with any text classifiers. In this paper, we choose to use a 1D-CNN model because the multi-head attention mechanism in our pseudo label generation can be viewed as applying a few corresponding convolutional filters. Specifically, every aspect representation  $\mathbf{a}_i$  is equivalent to a convolutional filter of window size one.

As mentioned before, we have  $\mathbf{X}_i$  as the embedding representation matrix of text segment  $S_i$ . We feed  $\mathbf{X}_i$  to our 1D-CNN model, as illustrated in Figure 1. Specifically, we employ various filters of window sizes two, three, and four, corresponding to bi-grams, tri-grams, and four-grams. We

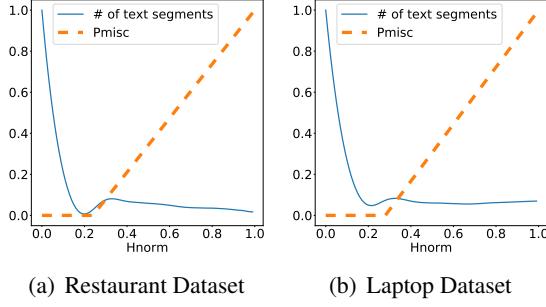


Figure 2:  $H_{norm}$  Distribution and  $P_{misc}$  Visualization.

apply these filters on the input matrix and then add a dropout layer after convolutional layers to alleviate over-fitting. Finally, we use a softmax layer to transform the output to probabilities as  $P_{i,j}$ , denoting the probability of  $S_i$  belonging to aspect  $A_j$ . The pseudo label distribution  $q_i$  generated in the previous step serves as supervision here, using the KL-divergence loss as below.

$$\mathcal{L} = KL(q_i, P_i) = - \sum_{j=1}^K q_{i,j} \log \frac{P_{i,j}}{q_{i,j}} \quad (5)$$

The same classification logic applies to the training of both K-aspect and (K+1)-aspect classifiers.

## 5 Misc Aspect Handling

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.,  $P_{misc}$ . Specifically,

$$H_{norm} = -\frac{1}{\log K} \sum_{j=1}^K P_{i,j} \log(P_{i,j}) \quad (6)$$

As shown in Figure 2, we plot the distribution of  $H_{norm}$  for all text segments on both the Restaurant and Laptop datasets. One can easily observe that a large volume of the text segments have low  $H_{norm}$ , indicating that they belong to some pre-defined aspects. At the same time, those “*misc*” aspect text segments follow a long-tail distribution over large  $H_{norm}$  values. Ideally, we want to (1) classify text segments with low enough  $H_{norm}$  values to

be a non-“*misc*” and (2) assign a higher  $P_{misc}$  if the  $H_{norm}$  is higher. Therefore, we propose to leverage a ReLU-like function to quantify  $P_{misc}$  based on  $H_{norm}$ .

$$P_{i,misc} = \begin{cases} (H_{norm} - \gamma)/(1 - \gamma) & H_{norm} \geq \gamma \\ 0 & H_{norm} < \gamma \end{cases}$$

We choose the value of  $\gamma$  as the 3rd quantile of the  $H_{norm}$  scores of all documents because based on Figure 2, 3rd quantile will give a suitable pivot point. Specifically, in Figure 2, the  $\gamma$  values on the Restaurant and Laptop datasets are 0.24 and 0.28, respectively.

After getting this, we combine  $P_{i,misc}$  and  $P_{i,j}$ :

$$\hat{q}_{i,j} = \begin{cases} (1 - P_{i,misc})P_{i,j} & j \leq K \\ P_{i,misc} & j = K + 1 \end{cases} \quad (7)$$

Finally, we obtain the pseudo labels  $\hat{q}_{i,j}$  for all aspects, including the *misc* aspect. We then train a (K+1)-aspect 1D-CNN classifier.

## 6 Seed Tuning, Expansion, and Filtering

Besides the user-provided seed words, there are usually more strong aspect indicator words embedded in the raw input corpus. It could be helpful to discover and add such words into the seed sets.

**Seed Tuning.** Not every word could be a candidate seed word (e.g., stopwords). Therefore, we build a candidate pool based on the K-aspect classifier. Specifically, we try to replace each word by the special UNK token and compute the KL divergence between the prediction results before and after. Given a word, if there exists one text segment where this word leads to a KL divergence difference more than 0.05, the word becomes a candidate. The intuition here is we want to prepare a candidate pool with high recall and reasonable precision. Also, as further ranking and filtering will be applied, this threshold is fairly easy to decide.

**Seed Expansion.** Then, we expand the seed sets by ranking and adding words from the candidate pool. Given an aspect  $A_j$  and its candidate pool  $C_j$ , we mainly consider two measurements:

- **Indicative.** As the pseudo label generation process can be viewed as a soft version of string matching using embedding, we want to select words whose presence strongly indicate a certain aspect. Mathematically, we want to select the word  $w$ , if it has a high posterior probability  $P(A_j|w)$ .  $P(A_j|w)$  means that given the presence of a word  $w$ , how likely the text segment

Table 1: Dataset Statistics.

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

belongs to the aspect  $A_j$ . Therefore, we define the indicative measure as

$$\text{Indicative}(A_j, w) = \frac{f_{A_j, w}}{f_{A_j}} \quad (8)$$

where  $f_{A_j, w}$  is the frequency of the word  $w$  appeared in text segments of the aspect  $A_j$ , and  $f_{A_j}$  refers to the total text segments of the aspect  $A_j$ . The frequency is calculated based on the prediction results on the training set.

- **Distinctive.** Ideally, a seed word should be only frequent in its own aspect. Therefore, we propose a distinctive measure to capture this. It measures how distinctive this word  $w$  in aspect  $A_j$  is compared with all other aspects.

$$\text{Distinctive}(A_j, w) = \frac{f_{A_j, w}}{\max_{k \neq j} f_{w, A_k}} \quad (9)$$

Since these two scores are of different scales, we aggregate them using the geometric mean, which has been shown effective in other comparative analyses (Tao et al., 2015). Ranking by the aggregated score, we replace the seed words of the aspect  $A_j$  by the top words here.

**Seed Filtering.** It is worth noting that the same ranking heuristic can be applied to the “*misc*” aspect as well. We observe that highly ranked words in the “*misc*” aspect are mostly general words or some noisy words that are related to multiple pre-defined aspects. By checking some examples on the Restaurant dataset, we observe that “restaurant” is ranked high in the “*misc*” aspect, as it can appear in text segments of any aspects; the word “place” is also a top-ranked word for the “*misc*” aspect. Other than location-related text segments, it also appears frequently in text segments like “this restaurant is such a great place.” Intuitively, the user may select this word as a seed word for “*location*” aspect, however, it is in fact very noisy. Therefore, when replacing the seed words, we propose to maintain a new pool of noisy words following the ranking in the “*misc*” aspect and exclude top words in this pool from seed words in pre-defined aspects.

## 7 Experiments

In this section, we empirically evaluate our proposed framework ARYA against many compared

Table 2: User-Provided Seed Words for the Restaurant Dataset. By default, we randomly sample 5 seed words from each aspect and run experiments.

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

methods. We also explore the effects of the number of iterations and the number of seeds. A case study about seed word evolution will be presented too.

### 7.1 Datasets

We have prepared two review datasets in the restaurant and laptop domains for evaluation. Table 1 presents you some statistics. These two datasets can be found in our repo<sup>3</sup>.

- **Restaurant.** There are 5 aspects in our Restaurant dataset: “*food*”, “*service*”, “*ambience*”, “*drinks*”, and “*location*”. For training, we have collected 16,061 unlabeled restaurant reviews from the Yelp Dataset Challenge data<sup>4</sup>.
- **Laptop.** There are 7 aspects in our Laptop dataset: “*support*”, “*display*”, “*battery*”, “*software*”, “*keyboard*”, “*os*”, “*mouse*”. For training, we are using 14,683 unlabeled Amazon reviews on laptop, collected by McAuley et al. (2015); He and McAuley (2016).

**User-Provided Seed Words.** For both datasets, we ask three domain experts to provide 10 seed words for each pre-defined aspect. Table 2 shows the seed word list provided by one expert for the Restaurant dataset. By default, we will randomly choose 5 seed words from them to train all the models, including both ours and baselines. We report the average of these test results. For one tricky aspect, the “*keyboard*” aspect of the Laptop dataset, we have only collected 3 seed words.

**Pre-processing.** We pre-process the corpus using the spaCy<sup>5</sup>. Special characters such as “\*”, “#” and redundant punctuations are removed. We learn word embedding on the unlabeled training corpus.

<sup>3</sup><https://github.com/peiranli/ARYA>

<sup>4</sup><https://www.yelp.com/dataset/challenge>

<sup>5</sup><https://spacy.io/>

Table 3: Evaluation Results on the Restaurant and Laptop Datasets. All precision, recall, and F<sub>1</sub> scores are averaged in the macro-weighted manner. Underlines highlight the best compared models.

Method	Restaurant			Laptop		
	Precision	Recall	F <sub>1</sub>	Precision	Recall	F <sub>1</sub>
CosSim	0.5455	0.4782	0.4985	0.6055	0.5437	0.5083
ABAE	0.5494	0.4904	0.5112	0.6127	0.6168	0.5950
MATE	0.5613	0.5127	0.5177	0.6418	0.6550	0.6474
WeSTClass	0.6153	0.5259	<u>0.5461</u>	0.6688	0.6848	<u>0.6523</u>
Dataless	0.5225	0.4467	0.4265	0.5601	0.5693	0.5569
BERT	0.5955	0.5285	0.5404	0.5949	0.5672	0.5632
Best+OurMisc	0.5864	0.5373	0.5256	0.6724	0.6996	0.6685
<b>ARYA</b>	<b>0.7410</b>	<b>0.6913</b>	<b>0.7067</b>	<b>0.7849</b>	<b>0.7321</b>	<b>0.7447</b>
ARYA-NoIter	0.6934	0.6740	0.6749	0.7508	0.7037	0.7027
ARYA-NoTuning	0.7019	0.6620	0.6729	0.7349	0.6874	0.6822
ARYA-NoFilter	0.7145	0.6706	0.6836	0.7619	0.7158	0.7306

## 7.2 Compared Models

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

- **CosSim** assigns the most similar aspect to each text segment according to the cosine similarity between the average word embedding of the text segment and the average word embedding of all seeds in each aspect.
- **Dataless** (Song and Roth, 2014) 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.
- **ABAE** (He et al., 2017) is an unsupervised neural topic model. We extend the ABAE by utilizing user-provided seed words for each aspect to align its topics to pre-defined aspects.
- **MATE** (Angelidis and Lapata, 2018) is an extended version of ABAE, which accepts seed information for guidance and replaces ABAE’s aspect dictionary with seed matrices.
- **WeSTClass** (Meng et al., 2018) is the state-of-the-art weakly supervised text classification model, which accepts seed words as supervision.
- **BERT** (Devlin et al., 2018) is a powerful contextualized representation learning technique. We use seed words matching and majority voting to generate sentence labels and then fine-tune the BERT for classification.

Most of these models do not take care of the “misc” aspect systematically. Therefore, we fine-tune the best compared method using our proposed “misc”-aspect handling, referred as **Best+OurMisc**.

We denote our model as **ARYA**. In addition, we have a few ablated versions as follows. **ARYA-**

**NoIter** uses our proposed “misc” aspect handling technique to generate the probability of “misc” aspect based on K-aspect classifier, however, without any further steps. **ARYA-NoTuning** refers to the version of our model without the seed tuning technique, i.e., no KL divergence threshold for seed word candidates. **ARYA-NoFilter** is our model without the seed filtering technique, i.e., no noisy seed words removal in pre-defined aspects based on “misc” aspect information.

## 7.3 Experiment Setup

**Default Parameters.** We set the word embedding dimension  $d = 200$ . For the classifier training, we fix the number of epoch as 5 since the training error tends to converge after 5 epochs. The KL divergence threshold for seed tuning is set to 0.05. This value is set based on some human efforts. One can easily observe that words lead to a KL divergence difference less than 0.05 are not very representative for that aspect. Based on the raw corpus sizes, we set the maximum number of seed words per each aspect as 10 on the Restaurant dataset, and 8 on the Laptop dataset.

**Evaluation Metrics.** We use macro-weighted average precision, recall, and F1 scores.

## 7.4 Experiment Results

We present the evaluation results on the Restaurant and Laptop datasets in Table 3. It is clear that our proposed method ARYA outperforms all other methods with significant margins on both datasets because none of these models considers the “misc” aspect systematically. Even compared with the fine-tuned second best models Best+OurMisc, ARYA results in 18% and 8% in absolute improvements over it on the Restaurant and Laptop

Table 4: Seed Word Evolution Examples. The 0-th iteration indicates the user-provided seeds.

Dataset	Aspect	Iter	Seed Words
Restaurant	food	0	spicy, pizza, sushi, food, tasty
		1	pizza, spicy, variety, tasty, tuna, sushi, portion, food, specials, bland
	location	0	avenue, convenient, river, street, block
		1	located, block, view, convenient, river, avenue
		2	located, block, street, view, convenient, park, river, avenue
Laptop	keyboard	0	keyboard, key, space
		1	keyboard, keys, key
	os	2	keys, keyboard, numeric, volume, palm, key, layout, keyboards
		0	system, os, ios, windows, mac
		1	system, os, ios, operating, mac, windows, lion, interface

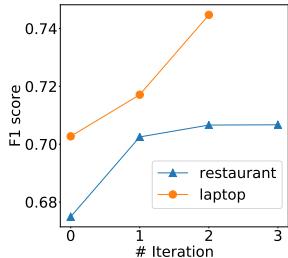


Figure 3: F<sub>1</sub> scores in Different Iterations. ARYA keeps iterating until the seed words converge.

datasets, respectively. It is also worth noting that ARYA-NoIter significantly outperforms all compared methods. All these observations show the importance of properly handling the “*misc*” aspect.

Among all compared methods, MATE is arguably the second-best method. It utilizes the multi-head attention mechanism, which is the same as our pseudo label generation step. This implies that attention mechanism is very important for aspect extraction tasks. ARYA generalizes attentions to more convolutional filters, thus being able to train a more powerful model.

The advantage of ARYA over ARYA-NoIter demonstrates the importance of progressively refine the model by updating seed words at every iteration. Comparing ARYA-NoTuning and ARYA-NoIter, one can see that if we do not carefully limit the scope of seed word candidates, there is a risk of adding noisy seed words that will lead to even worse performance (e.g., on the Laptop dataset). The improvement of ARYA over ARYA-NoFilter reveals the effectiveness of filtering the seed words in pre-defined aspects by the “*misc*” aspect.

## 7.5 Seed Word Evolution

ARYA keeps iterating until the seed words converge. So the number of iterations in ARYA is

decided automatically. Figure 3 shows that the F<sub>1</sub> score increases w.r.t. iterations on both datasets. This suggests that our framework truly enables mutual enhancements between pre-defined aspects and the “*misc*” aspect over iterations.

Table 4 presents the seed words of each aspect w.r.t. different iterations on both datasets. We can observe that the seed words become much better after the seed expansion than the initial seed words.

As mentioned before, even domain experts feel challenging to provide seed words for the “*keyboard*” aspect. Only three seed words, “*keyboard*”, “*key*”, and “*space*” are given at the very beginning. After a few rounds of seed tuning, expansion, and filtering, some interesting words are added to its seed set, such as “*layout*”, “*numeric*”, and “*palm*”, which make sense for the keyboard aspect. For example, “*palm*” describes the how comfortable the palms are when typing on a keyboard or how big the keyboard is compared with palms. It is interesting to see that our model can automatically discover these words beyond typical examples come up by experts.

We also observe that the seed word sets of popular aspects converge faster than infrequent aspects. For example, on the Restaurant dataset, the “*food*”, “*ambience*”, and “*service*” aspects converge after the 1st iteration, and the “*drinks*” and “*location*” aspects requires 2 and 3 iterations, respectively. The first three have significantly more text segments than the latter two.

Another observation is that the tricky aspects converge slower than the other aspects. For example, on the Laptop dataset, the “*keyboard*” aspect converges much slower than the other aspects, because it is very counter-intuitive to come up with the seed words such as “*palm*” and “*numeric*”. On the contrary, the “*os*” aspect is relatively easy com-

pared with other aspects.

## 7.6 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.”, without any specific aspect. ARYA detects that the word “pleasant” as a noisy word, because it can refer to “service” or “ambience”. Therefore, it is filtered for these two aspects. Eventually, ARYA 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. ARYA 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.

## 8 Related Works

Aspect extraction was originated at a document-level task, instead of working on text segments. Rule-based methods (Hu and Liu, 2004; Liu et al., 2005; Zhuang et al., 2006; Scaffidi et al., 2007; Zhang et al., 2010; Qiu et al., 2011) are the pioneers along this direction. A number of unsupervised learning methods based on the LDA topic model and its variants (Titov and McDonald, 2008; Zhao et al., 2010; Brody and Elhadad, 2010; Mukherjee and Liu, 2012; Zhang et al., 2016; Shams and Baraani-Dastjerdi, 2017) treat extracted topics as aspects. More recently, a neural model ExtRA (Luo et al., 2018) is proposed to further improve the aspect extraction at the document level. However, since our problem focuses on text segments, directly applying these document-level methods leads to some unsatisfactory results.

There are several recent unsupervised attempts on aspect extraction for text segments. ABAE (He et al., 2017) employs an attention module to learn embedding for text segments and an auto-encoder framework to build aspect dictionaries. However, it requires users to first set the number of topics as a much larger number than the number of desired aspects, and then manually merge and map the extracted topics back to the aspects. Building upon ABAE, Angelidis and Lapata (2018) further proposed a multi-seed aspect extractor MATE us-

ing 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 seed word tuning and expansion to overcome this issue, thus outperforming MATE significantly in the extensive experiments.

Our problem shares certain similarities with the weakly-supervised text classification problem. Existing methods can build document classifiers by taking either hundreds of labeled training documents (Tang et al., 2015; Miyato et al., 2016; Xu et al., 2017), class/category names (Song and Roth, 2014; Li et al., 2018), or user-provided seed words (Meng et al., 2018) as the source of weak supervision. However, 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.

## 9 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 identify the key challenge lies in how to properly handle the “*misc*” aspect, for which even domain experts cannot easily design seed words. We propose a novel framework, ARYA, which incorporates the “*misc*” aspect systematically. In our framework, we induce supervision for the “*misc*” aspect using seed words of pre-defined aspects. At the same time, we utilize the “*misc*” aspect information to filter out the noisy words from the seed list of pre-defined aspects. Extensive experiments have demonstrated the effectiveness of ARYA and verified the necessity of modeling the “*misc*” aspect.

In the future, we would like to integrate the extracted aspect information with downstream tasks, such as sentiment analysis and opinion summarization. We also want to explore the use of contextualized representation in weakly supervised aspect extraction, further disambiguating words based on contexts. In addition, we are interested in extending our work to document classifications even with multiple labels per document.

## References

- Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 85–94. ACM.
- Stefanos Angelidis and Mirella Lapata. 2018. Summarizing opinions: Aspect extraction meets sentiment prediction and they are both weakly supervised. *arXiv:1808.08858*.
- Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *NAACL*, pages 804–812.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*.
- Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An unsupervised neural attention model for aspect extraction. In *ACL*, pages 388–397.
- Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *SIGKDD*, pages 168–177.
- Benjamin J Kuipers, Patrick Beeson, Joseph Modayil, and Jefferson Provost. 2006. Bootstrap learning of foundational representations. *Connection Science*, 18(2):145–158.
- Keqian Li, Hanwen Zha, Yu Su, and Xifeng Yan. 2018. Unsupervised neural categorization for scientific publications. In *SIAM Data Mining*, pages 37–45. SIAM.
- Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. In *WWW*, pages 342–351.
- Zhiyi Luo, Shanshan Huang, Frank F Xu, Bill Yuchen Lin, Hanyuan Shi, and Kenny Zhu. 2018. Extra: Extracting prominent review aspects from customer feedback. In *EMNLP*, pages 3477–3486.
- Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *SIGIR*.
- Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2018. Weakly-supervised neural text classification. In *CIKM*, pages 983–992.
- Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv:1605.07725*.
- Arjun Mukherjee and Bing Liu. 2012. Aspect extraction through semi-supervised modeling. In *ACL*, pages 339–348.
- Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational linguistics*, 37(1):9–27.
- Ellen Riloff, Janyce Wiebe, and Theresa Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 25–32. Association for Computational Linguistics.
- Christopher Scaffidi, Kevin Bierhoff, Eric Chang, Mikhael Felker, Herman Ng, and Chun Jin. 2007. Red opal: product-feature scoring from reviews. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 182–191.
- Mohammadreza Shams and Ahmad Baraani-Dastjerdi. 2017. Enriched lda (elda): combination of latent dirichlet allocation with word co-occurrence analysis for aspect extraction. *Expert Systems with Applications*, 80:136–146.
- Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *TKDE*, 30(10):1825–1837.
- Yangqiu Song and Dan Roth. 2014. On dataless hierarchical text classification. In *AAAI*.
- Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *SIGKDD*, pages 1165–1174.
- Fangbo Tao, Chao Zhang, Xiusi Chen, Meng Jiang, Tim Hanratty, Lance Kaplan, and Jiawei Han. 2015. Doc2cube: Automated document allocation to text cube via dimension-aware joint embedding. *Dimension*, 2016:2017.
- Ivan Titov and Ryan McDonald. 2008. Modeling online reviews with multi-grain topic models. In *WWW*, pages 111–120. ACM.
- Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. 2017. Variational autoencoder for semi-supervised text classification. In *AAAI*.
- Chen Zhang, Hao Wang, Liangliang Cao, Wei Wang, and Fanjiang Xu. 2016. A hybrid term-term relations analysis approach for topic detection. *Knowledge-Based Systems*, 93:109–120.
- Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O’Brien-Strain. 2010. Extracting and ranking product features in opinion documents. In *Proceedings of the 23rd international conference on computational linguistics: Posters*, pages 1462–1470. Association for Computational Linguistics.
- Wayne Xin Zhao, Jing Jiang, Hongfei Yan, and Xiaoming Li. 2010. Jointly modeling aspects and opinions with a maxent-lda hybrid. In *EMNLP*, pages 56–65.

- Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006. Movie review mining and summarization. In *CIKM*, pages 43–50.