Multi-Label Few-Shot Learning for Aspect Category Detection

Mengting Hu[1] **Shiwan Zhao**[2]* **Honglei Guo**[2] **Chao Xue**[3]†
Hang Gao[1] **Tiegang Gao**[1] **Renhong Cheng**[1] **Zhong Su**[2]
[1] Nankai University [2] IBM Research - China [3] JD Explore Academy
{mthu, knimet}@mail.nankai.edu.cn, {zhaosw, guohl, suzhong}@cn.ibm.com
xuechao19@jd.com, {gaotiegang, chengrh}@nankai.edu.cn

Abstract

Aspect category detection (ACD) in sentiment analysis aims to identify the aspect categories mentioned in a sentence. In this paper, we formulate ACD in the few-shot learning scenario. However, existing few-shot learning approaches mainly focus on single-label predictions. These methods can not work well for the ACD task since a sentence may contain multiple aspect categories. Therefore, we propose a multi-label few-shot learning method based on the prototypical network. To alleviate the noise, we design two effective attention mechanisms. The support-set attention aims to extract better prototypes by removing irrelevant aspects. The query-set attention computes multiple prototype-specific representations for each query instance, which are then used to compute accurate distances with the corresponding prototypes. To achieve multi-label inference, we further learn a dynamic threshold per instance by a policy network. Extensive experimental results on three datasets demonstrate that the proposed method significantly outperforms strong baselines.

1 Introduction

Aspect category detection (ACD) (Pontiki et al., 2014, 2015) is an important task in sentiment analysis. It aims to identify the aspect categories mentioned in a given sentence from a predefined set of aspect categories. For example, in the sentence *"the cheesecake is tasty and the staffs are friendly"*, two aspect categories, i.e. *food* and *service*, are mentioned. The performance of existing approaches for the ACD task (Zhou et al., 2015; Schouten et al., 2018; Hu et al., 2019) relies heavily on the scale of the labeled dataset. They usually suffer from limited data and fail to generalize well to novel aspect categories with only a few labeled instances. On the one hand, it is time-consuming and labor-intensive to annotate large-scale datasets. On the other hand, given a large dataset, many long-tail aspects still suffer from data sparsity.

Support set	
(A) room_cleanliness	1) *Cleanliness was great, and the food was really good.* 2) *People have mentioned, bed bugs on yelp !!*
(B) staff_owner	1) *I think the salon has problems starting with the owner.* 2) *The owner is very nice.*
(C) hotel	1) *Okay, so it is a cute chain hotel.* 2) *I really don't see how people are giving this hotel such high ratings.*
Query set	
(A) and (C)	1) *Hotel is just plain dirty.*
(B)	2) *The owners are extremely smart and worldly*
(B)	3) *Not a typical customer service response, especially from the owner !*

Figure 1: Example meta-task in a 3-way 2-shot scenario. The words in gray background describe the target aspects of interest, while the words marked by the rectangle are irrelevant aspects, which tend to be noise for this meta-task.

Few-shot learning (FSL) provides a solution to address the above challenges. FSL learns like a human, identifying novel classes with limited supervised information by exploiting prior knowledge. Many efforts have been devoted to FSL (Ravi and Larochelle, 2017; Finn et al., 2017; Snell et al., 2017; Wang et al., 2018; Gao et al., 2019). Among these methods, the prototypical network (Snell et al., 2017) is a promising approach, which is simple but effective. It follows the meta-learning paradigm by building a collection of N-way K-shot meta-tasks. A meta-task aims to infer a query set with the help of a small labeled support set. It first learns a prototype for each class in the support set. Then the query instance is predicted by measuring the distance with N prototypes in the embedding space.

In this paper, we formulate ACD in the FSL

* Shiwan Zhao is the corresponding author.
† The work was (partially) done in IBM.

scenario, which aims to detect aspect categories accurately with limited training instances. However, ACD is a multi-label classification problem since a sentence may contain multiple aspect categories. Most FSL works learn a single-label classifier and can not work well to address the ACD task. The reasons are two-fold. Firstly, the sentences of each class (i.e., aspect category) in the support set are diverse and contain noise from irrelevant aspects. As displayed in Figure 1, there are three classes in the support set, and each class has two instances. The aspect categories *food* and *salon* tend to be noise for this meta-task, making it hard to learn a good prototype for each class in the support set. Secondly, the query set is also noisy. Figure 1 demonstrates three different cases. The first sentence mentions two aspects *hotel* and *room_cleanliness* out of the support set. We need to detect both aspects accurately as multi-label classification. When detecting each of them, the other aspect acts as noise and makes the task hard. The second sentence is an easy case with a single aspect *staff_owner*. The third sentence mentions the aspect *staff_owner* out of the support set, while the aspect *service* is noise for this meta-task. In summary, the noise from both the support set and query set makes the few-shot ACD a challenging task.

To this end, we propose a multi-label FSL method based on the prototypical network (Snell et al., 2017). We alleviate the noise in the support set and query set by two effective attention mechanisms. Concretely, the support-set attention tries to extract the common aspect of each class. By removing the noise (i.e., irrelevant aspects), the support-set attention can yield better prototypes. Then for a query instance, the query-set attention utilizes the prototypes to compute multiple prototype-specific query representations, in which the irrelevant aspects are removed. Given the better prototypes and the corresponding prototype-specific query representations, we can compute accurate distances between the query instance and the prototypes in the embedding space. We detect the aspect categories in the query instance by ranking the distances. To select the positive aspects from the ranking, we design a policy network (Williams, 1992) to learn a dynamic threshold for each instance. The threshold is modeled as the action of the policy network with continuous action space.

The main contributions of our work are as follows:

- We formulate ACD as a multi-label FSL problem and design a multi-label FSL method based on the prototypical network to solve the problem. To the best of our knowledge, we are the first to address ACD in the few-shot scenario.
- To alleviate the noise from the support set and query set, we design two effective attention mechanisms, i.e., support-set attention and query-set attention.
- Experimental results on the three datasets demonstrate that our method outperforms strong baselines significantly.

2 Related Work

Aspect Category Detection Previous works for ACD can mainly be divided into two types: unsupervised and supervised methods. Unsupervised approaches extract aspects by mining semantic association (Su et al., 2006) or co-occurrence frequency (Hai et al., 2011; Schouten et al., 2018). These methods require a large corpus to mine aspect knowledge and have limited performance. Supervised methods address this task via hand-crafted features (Kiritchenko et al., 2014), automatically learning useful representations (Zhou et al., 2015), multi-task learning (Xue et al., 2017; Hu et al., 2019), or topic-attention model (Movahedi et al., 2019). The above methods detect aspect categories out of a pre-defined set, which cannot handle the unseen classes. These challenges motivate us to investigate this task in the few-shot scenario.

Few-Shot Learning Few-shot learning (FSL) (Fe-Fei et al., 2003; Fei-Fei et al., 2006) is close to real artificial intelligence, which borrows the learning process from the human. By incorporating the prior knowledge, it obtains new knowledge fast with limited supervised information. Many works have been proposed for FSL, which can be mainly divided into four research directions.

One promising direction is distance-based methods. These methods measure the distance between instances in the feature embedding space. The siamese network (Koch et al., 2015) infers the similarity score between an instance pair. Others compare the cosine similarity (Vinyals et al., 2016) or Euclidean distance (Snell et al., 2017). The relation network (Sung et al., 2018) exploits a neural network to learn the distance metric. Afterward,

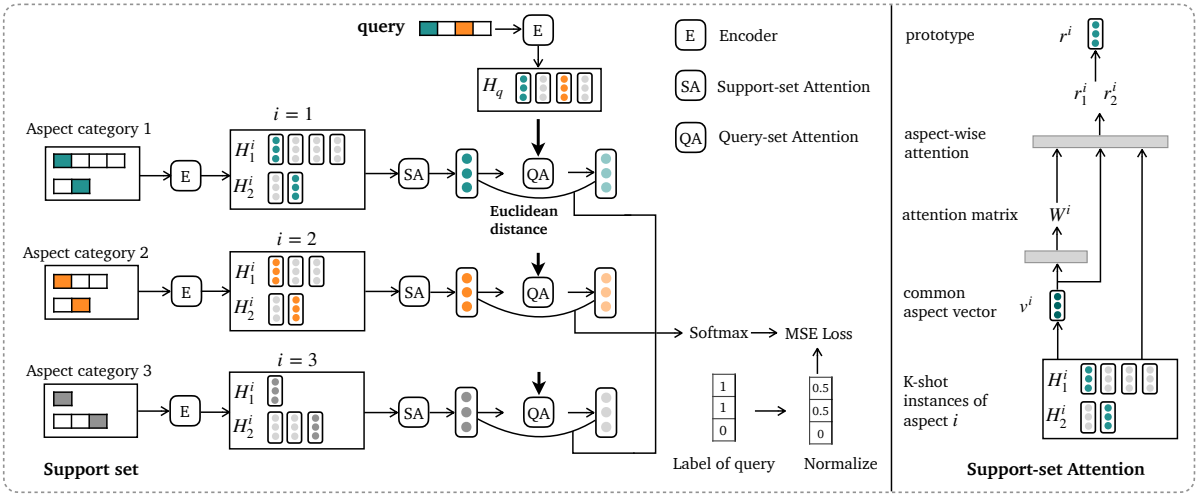


Figure 2: The left part depicts the main network for an example N-way K-shot meta-task with a query instance ($N = 3, K = 2$). Each small cube of the instance symbolizes an aspect category. The colored cubes indicate the target aspects of interest while the white cubes indicate the noisy aspects. The right part shows the details of the support-set attention.

Garcia and Bruna (2018) utilize graph convolution network to extract the structural information of classes. The second direction focuses on the optimization of networks. Model-agnostic meta-learning (MAML) algorithm (Finn et al., 2017) learns a good initialization of the model and updates the model by a few labeled examples. Meta networks (Munkhdalai and Yu, 2017) achieve rapid generalization via fast parameterization. The third type is based on hallucination (Wang et al., 2018; Li et al., 2020). This research line directly deals with data deficiency by "learning to augment", which designs a generator on the base classes and then hallucinates novel class data to augment few-shot samples. The last direction introduces a weight generator to predict classification weight given a few novel class samples, either based on attention mechanism (Gidaris and Komodakis, 2018) or Gaussian distribution (Guo and Cheung, 2020).

A recent work Proto-HATT (Gao et al., 2019) is similar to ours. Proto-HATT is based on the prototypical network (Snell et al., 2017), which deals with the text noise in the relation classification task by employing hybrid attention at both the instance-level and the feature-level. This method is designed for single-label FSL. Compared with it, our method designs two attention mechanisms to alleviate the noise on the support set and query set, respectively. The collaboration of two attentions helps compute accurate distances between the query instance and prototypes, and then improves multi-label FSL.

Multi-Label Few-Shot Learning Compared with single-label FSL, the multi-label FSL has been underexplored. Previous works focus on image synthesis (Alfassy et al., 2019) and signal processing (Cheng et al., 2019). Rios and Kavuluru (2018) develop few-shot and zero-shot methods for multi-label text classification when there is a known structure over the label space. Their approach relies on label descriptors and the hierarchical structure of the label spaces, which limits its application in practice. Hou et al. (2020) propose to address the multi-label intent detection task in the FSL scenario. It calibrates the threshold by kernel regression. Different from this work, we learn a dynamic threshold per instance in a reinforced manner.

3 Methodology

In the few-shot ACD scenario, each meta-task contains a support set S and a query set Q. The meta-task is to assign the query instance to the class(es) of the support set. An instance may be a multi-aspect sentence. Thus a query sentence may describe more than one class out of the support set[1]. Therefore, we define the few-shot ACD as a multi-label few-shot classification problem.

3.1 Overview

Suppose in an N-way K-shot meta-task, the support set is $S = \{(x_1^i, ...x_K^i), y^i\}_{i=1}^N$, where each x^i

[1]We found that the probability of a query instance belonging to more than one class is around 4.5% in the ACD dataset, i.e. FewAsp, by randomly sampling 10,000 5-way 5-shot meta-tasks with 5 query sentences for each class.

is a sentence and $(x_1^i, ..., x_K^i)$ all contain the aspect category y^i. A query instance is $(x_q, \boldsymbol{y_q})$, where $\boldsymbol{y_q}$ is a binary label vector indicating the aspects in x_q out of N classes.

Figure 2 presents the main network by an example 3-way 2-shot meta-task. It is composed of three modules, i.e., encoder, support-set attention (SA) and query-set attention (QA). Each class in the support set contains K instances, which are fed into the encoder to obtain K encoded sequences. Next, SA module extracts a prototype for this class from the encoded sequences. After obtaining N prototypes, we feed a query instance into the QA module to compute multiple prototype-specific query representations, which are then used to compute the Euclidean distances with the corresponding prototypes. Finally, we normalize the negative distances to obtain the ranking of prototypes and then select the positive predictions (i.e., aspect categories) by a dynamic threshold. Next, we will introduce the modules of our method in detail.

3.2 Encoder

Given an input sentence $x = \{w_1, w_2, ..., w_n\}$, we first map it into an embedding sequence $\{\boldsymbol{e_1}, \boldsymbol{e_2}, ..., \boldsymbol{e_n}\}$ by looking up the pre-trained GloVe embeddings (Pennington et al., 2014). Then we encode the embedding sequence by a convolutional neural network (CNN) (Zeng et al., 2014; Gao et al., 2019). The convolution kernel slides with the window size m over the embedding sequence. We gain the contextual sequence $H = \{\boldsymbol{h_1}, \boldsymbol{h_2}, ..., \boldsymbol{h_n}\}$, $H \in \mathbb{R}^{n \times d}$:

$$\boldsymbol{h_i} = \text{CNN}(\boldsymbol{e}_{i-\frac{m-1}{2}}, ..., \boldsymbol{e}_{i+\frac{m-1}{2}}) \quad (1)$$

where CNN$(\cdot)$ is a convolution operation. The advantages of CNN are two-fold: first, the convolution kernel can extract n-gram features on the receptive field. For example, the bi-gram feature of *hot dog* could help detect the aspect category *food*; second, CNN enables parallel computing over inputs, which is more efficient (Xue and Li, 2018).

3.3 Support-set Attention (SA)

In each class of the support set, the K-shot instances describe a common aspect, i.e., the target aspect of interest[2]. As shown in Figure 1, two sentences, *"Cleanliness was great, and the food was really good"* and *"People have mentioned, bed bugs on yelp!!"*, share the common aspect *room_cleanliness*. The former contains two aspect categories *room_cleanliness* and *food*. In this example meta-task, it is an instance of the class *room_cleanliness*. However, when sampling other meta-tasks, the instance may be used to represent the class *food*. This leads to confusion and makes learning a good prototype difficult. To deal with the issue brought by multi-aspect sentences, we first need to identify the common aspect. As depicted in the right part of Figure 2, we compute the *common aspect vector* by the combination of the K-shot instances. We then regard the vector as a condition and inject it into the attention mechanism to make our attention mechanism aspect-wise.

Common Aspect Vector The encoded K-shot instances of a class contain one common aspect and some irrelevant aspects. Among these aspects, the common aspect is the majority. Thus, we simply conduct a word-level average to extract the common aspect vector $\boldsymbol{v^i} \in \mathbb{R}^d$.

$$\boldsymbol{v^i} = \text{avg}(H_1^i, H_2^i, ..., H_K^i) \quad (2)$$

The average operation highlights the common aspect, but cannot completely eliminate noisy aspects. To further reduce the noise of irrelevant aspects in each instance, we use the common aspect as the condition in the attention mechanism.

Aspect-Wise Attention To make the attention mechanism adapt to the condition, we have two designs. First, we directly use the common aspect vector to compute the attention with each instance (see Eq. 4), which filters out the irrelevant aspects of each instance to some extent. Second, we exploit the idea of dynamic conditional network, which has been demonstrated effective in FSL (Zhao et al., 2018). By predicting a dynamic attention matrix with the common aspect vector, our attention mechanism can further adapt to the condition, i.e., the common aspect vector of the class. Specifically, we learn different perspectives of the condition by simply repeating the common aspect vector (Vaswani et al., 2017). Then it is fed into a linear layer to obtain the attention matrix W^i for class i.

$$W^i = W(\boldsymbol{v^i} \otimes e_M) + \boldsymbol{b} \quad (3)$$

where $(\boldsymbol{v^i} \otimes e_M) \in \mathbb{R}^{e_M \times d}$ is the operation repeatedly concatenating $\boldsymbol{v^i}$ for e_M times. The linear layer has parameter matrix $W \in \mathbb{R}^{d \times e_M}$ and bias

[2] In almost all cases, there is only one common aspect in the K instances. We randomly sample 10,000 5-way 5-shot meta-tasks, and found that the probability of containing more than one common aspect in each class is less than 0.086%. The probability will be much lower in the 10-way scenario.

$\boldsymbol{b} \in \mathbb{R}^d$. This layer is shared in the classes of all meta-tasks, which is learned to be class-agnostic. Thus in the testing phase, it can generate aspect-wise attention for a novel class.

Then in class i of the support set, we exploit the common aspect vector and attention matrix to calculate a denoised representation for every instance. The denoised representation $\boldsymbol{r_j^i}$ for the j-th instance is computed as below.

$$\begin{aligned} \boldsymbol{\beta} &= \text{softmax}(\boldsymbol{v^i}\tanh(H_j^i W^i)) \\ \boldsymbol{r_j^i} &= \boldsymbol{\beta} H_j^i \end{aligned} \tag{4}$$

In this way, the support-set attention is adapted to the condition and is also class-specific. Thus it tends to focus on the correct aspect even for a multi-aspect sentence representing different classes.

Finally, the average of denoised representations for K-shot instances is the prototype of this class.

$$\boldsymbol{r^i} = \text{avg}(\boldsymbol{r_1^i}, \boldsymbol{r_2^i}, ..., \boldsymbol{r_K^i}) \tag{5}$$

After processing all classes in the support set, we obtain N prototypes $\{\boldsymbol{r^1}, \boldsymbol{r^2}, ..., \boldsymbol{r^N}\}$.

3.4 Query-set Attention (QA)

A query instance may also contain multiple aspects, making the sentence noisy. To deal with the noise in a query instance, we select the relevant aspects from the query instance by the QA module. Specifically, we first process the query instance by the encoder and obtain the encoded instance H_q. Then we feed H_q into the QA module to obtain multiple prototype-specific query representations $\boldsymbol{r_q^i}$ by the N prototypes.

$$\begin{aligned} \boldsymbol{\rho^i} &= \text{softmax}(\boldsymbol{r^i}\tanh(H_q)) \\ \boldsymbol{r_q^i} &= \boldsymbol{\rho^i} H_q \end{aligned} \tag{6}$$

The QA module tries to focus on the aspect category which is similar to the prototype. In Eq. 6, the attention is non-parametric. It can reduce the dependence on parameters and can accelerate the adaptation to unseen classes.

3.5 Training Objective

For a query instance, we compute the Euclidean distance (ED) between each prototype and its prototype-specific query representation, and we obtain N distances. Next, we normalize the negative distances as the final prediction, which is a ranking of the prototypes.

$$\hat{\boldsymbol{y}} = \text{softmax}(-\text{ED}(\boldsymbol{r^i}, \boldsymbol{r_q^i})), \quad i \in [1, N] \tag{7}$$

Dataset	#cls.	#inst./cls.	#inst.
FewAsp(single)	100	200	20000
FewAsp(multi)	100	400	40000
FewAsp	100	630	63000

Table 1: Statistics of three datasets. #cls. denotes the number of classes. #inst./cls. denotes the number of instances per class. #inst. denotes the total number of instances.

The training objective is the mean square error (MSE) loss:

$$L = \sum (\hat{\boldsymbol{y}} - \boldsymbol{y_q})^2 \tag{8}$$

where $\boldsymbol{y_q}$ is the ground-truth. We also normalize $\boldsymbol{y_q}$ to ensure the consistency between the prediction and the ground-truth.

Learning Dynamic Threshold (DT) To select the positive aspects from the ranking (see Eq. 7) for a query instance, we further learn a dynamic threshold. The threshold is modeled by a policy network (Williams, 1992), which has a continuous action space following Beta distribution (Chou et al., 2017). Given a query instance, we define the *state* as $[(\boldsymbol{r^1} - \boldsymbol{r_q^1})^2; ...; (\boldsymbol{r^N} - \boldsymbol{r_q^N})^2; \hat{\boldsymbol{y}}]$. We feed the state into the policy network and obtain the parameters a and b of a Beta distribution. Then we sample a threshold τ from $Beta(\tau|a, b)$. The *reward score* is the F1 score for this instance based on τ. We also introduce a reference $score^*$, which is the F1 score based on a baseline action, i.e., the mode of $Beta(\tau|a, b)$: $\frac{a-1}{a+b-2}$. The training objective is defined as below to minimize the negative expected reward.

$$L_t = -(score - score^*)\log P(\tau) \tag{9}$$

where $P(\tau)$ is the probability of τ in the Beta distribution. During inference, we select the positive aspects in $\hat{\boldsymbol{y}}$ with the baseline action.

4 Experiments

4.1 Datasets

We construct three few-shot ACD datasets from Yelp_aspect (Bauman et al., 2017), which is a large-scale multi-domain dataset for aspect recommendation. We group all instances by aspects and choose 100 aspect categories. Following Han et al. (2018), we split the 100 aspects without intersection into 64 aspects for training, 16 aspects for validation, and 20 aspects for testing.

Models	5-way 5-shot		5-way 10-shot		10-way 5-shot		10-way 10-shot	
	AUC	F1	AUC	F1	AUC	F1	AUC	F1
Relation Network	0.9331	75.79	0.9086	72.02	0.9181	63.78	0.9054	61.15
Matching Network	0.9705	81.89	0.9749	84.62	0.9630	70.95	0.9672	73.28
Graph Network	0.9654	81.45	0.9746	85.04	0.9545	70.75	0.9697	77.84
Prototypical Network	0.9649	83.30	0.9753	86.29	0.9597	74.23	0.9671	76.83
IMP	0.9665	83.69	0.9747	86.14	0.9600	73.80	0.9691	77.09
Proto-HATT	0.9645	83.33	0.9762	86.71	0.9571	73.42	0.9700	77.65
Proto-AWATT (ours)	**0.9756**†‡	**86.71**†‡	**0.9796**	**88.54**†‡	**0.9701**†‡	**80.28**†‡	**0.9755**†‡	**82.97**†‡

Table 2: Evaluation results in terms of AUC and macro-f1 (%) on FewAsp(single). All results are the average of 5 runs. The marker † refers to p-value$<$0.05 of the T-test when comparing with Prototypical Network. The marker ‡ refers to p-value$<$0.05 when comparing with Proto-HATT.

Models	5-way 5-shot		5-way 10-shot		10-way 5-shot		10-way 10-shot	
	AUC	F1	AUC	F1	AUC	F1	AUC	F1
Relation Network	0.8491	58.38	0.8621	61.37	0.8422	43.71	0.8472	44.85
Matching Network	0.8954	65.70	0.9138	69.02	0.8828	50.86	0.8994	54.42
Graph Network	0.8797	59.25	0.9045	64.63	0.8605	45.42	0.8844	48.49
Prototypical Network	0.8967	67.88	0.9160	72.32	0.8801	52.72	0.9068	58.92
IMP	0.9012	68.86	0.9229	73.51	0.8871	53.96	0.9110	59.86
Proto-HATT	0.9110	69.15	0.9303	73.91	**0.9044**	55.34	**0.9238**	60.21
Proto-AWATT (ours)	**0.9145**†	**71.72**†	**0.9389**†	**77.19**†‡	0.8980†	**58.89**†‡	0.9234†	**66.76**†‡

Table 3: Evaluation results in terms of AUC and macro-f1 (%) on FewAsp(multi).

According to the sentence type, i.e., single-aspect or multi-aspect[3], we sample different types of sentences from each group and construct three datasets: FewAsp(single), FewAsp(multi), and FewAsp, which are composed of single-aspect, multi-aspect, and both types of sentences, respectively. Note that FewAsp is randomly sampled from the original set of each class, which can better reflect the data distribution in real applications. The statistics of the three datasets are shown in Table 1.

4.2 Experimental Settings

Evaluation Metrics Previous single-label FSL (Snell et al., 2017) usually evaluates performance by accuracy. In the multi-label setting, we choose AUC (Area Under Curve) and macro-f1 as the evaluation metrics. AUC is utilized for model selection and macro-f1 is computed with a threshold. In our experiments, we found that for all methods in three datasets, the overall best thresholds are 0.3 in the 5-way setting and 0.2 in the 10-way setting. Thus we choose them for evaluating the baselines.

Training Details We first train the main network with MSE loss L (Eq. 8). Then we initialize the main network with the learned parameters and jointly train the policy network with L_t (Eq. 9). The implementation details are described in the appendix.

[3]A sentence contains a single aspect or multiple aspects.

4.3 Compared Methods

Our approach is named as **Proto-AWATT** (aspect-wise attention). We validate the effectiveness of the proposed method by comparing with the following popular approaches.

- **Matching Network** (Vinyals et al., 2016): It is a metric-based attention method, where distance is measured by cosine similarity.
- **Prototypical Network** (Snell et al., 2017): It computes the average of embedded support examples for each class as the prototype, and then measures the distance between the embedded query instance and each prototype.
- **Relation Network** (Sung et al., 2018): It utilizes a neural network to learn the relation metric.
- **Graph Network** (Garcia and Bruna, 2018): It casts FSL as a supervised message passing task by graph neural network.
- **IMP** (Allen et al., 2019): It proposes infinite mixture prototypes to represent each class by a set of clusters, with the number of clusters determined directly from the data.
- **Proto-HATT** (Gao et al., 2019): It is based on the prototypical network, which deals with the

Models	5-way 5-shot		5-way 10-shot		10-way 5-shot		10-way 10-shot	
	AUC	F1	AUC	F1	AUC	F1	AUC	F1
Relation Network	0.8556	59.52	0.8698	62.78	0.8494	45.62	0.8377	44.70
Matching Network	0.9076	67.14	0.9239	70.09	0.8844	51.27	0.8990	54.61
Graph Network	0.8948	61.49	0.9235	69.89	0.8735	47.91	0.9019	56.06
Prototypical Network	0.8888	66.96	0.9177	73.27	0.8735	52.06	0.9013	59.03
IMP	0.8995	68.96	0.9230	74.13	0.8850	54.14	0.9081	59.84
Proto-HATT	0.9154	70.26	0.9343	75.24	0.9063	57.26	0.9286	61.51
Proto-AWATT (ours)	**0.9335**†‡	**75.37**†‡	**0.9528**†‡	**80.16**†‡	**0.9206**†	**65.65**†‡	**0.9342**†	**69.70**†‡

Table 4: Evaluation results in terms of AUC and macro-f1 (%) on FewAsp.

Models	FewAsp	
	AUC	F1
Proto-AWATT (ours)	**0.9206**	**65.65**
w/o SA	0.8890	54.34
w/o attention matrix W^i	0.9128	61.68
w/o QA	0.8886	51.19
w/o DT	0.9161	64.48
w/o DT w/ KR	0.9159	64.06
w/o DT w/ MS	0.9163	64.00

Table 5: Ablation study of the 10-way 5-shot scenario on FewAsp.

Models	Proto-HATT		Proto-AWATT	
	AUC	F1	AUC	F1
GloVe + CNN	0.9063	57.26	0.9206	65.65
GloVe + LSTM	0.9137	59.46	0.9357	66.86
BERT	0.8971	57.33	0.9459	70.09
DistilBERT	0.9067	59.57	0.9451	70.23

Table 6: Ablation study of using different encoders in the 10-way 5-shot scenario on FewAsp.

noise with hybrid instance-level and feature-level attention mechanisms.

4.4 Experimental Analysis

We report the experimental results of various methods in Table 2, Table 3, Table 4 and Table 5. The best scores on each metric are marked in bold. The experimental results demonstrate the effectiveness of our method.

Overall Performance AUC and macro-f1 scores of all the methods are shown in Table 2, Table 3 and Table 4. Firstly, we observe that our method Proto-AWATT achieves the best results on almost all evaluation metrics of the three datasets. This reveals the effectiveness of the proposed method. Secondly, compared to Proto-HATT, Proto-AWATT achieves significant improvement. It is worth noting that the average improvement of macro-f1 on three datasets is 4.99%. This exhibits that the SA and QA modules successfully reduce noise for few-shot ACD. Meanwhile, accurate distance measurement between prototypes and the prototype-specific query representations can facilitate the detection of multiple aspects in the query instance.

Then we found that all methods on FewAsp(multi) perform consistently worse than the counterparts on FewAsp(single) and FewAsp. This is because more aspects increase the complexity of the dataset. On FewAsp(multi), Proto-AWATT still outperforms other methods in most settings, which demonstrates the robustness of our model on various data distributions.

In general, the 10-way scenario contains much more noise than the 5-way. We observe that compared to Proto-HATT, Proto-AWATT achieves more significant improvements in the 10-way scenario than the 5-way. The results further indicate that Proto-AWATT can really alleviate the noise.

Ablation Study Table 5 depicts the results of ablation study. Firstly, without the SA module, the performances of Proto-AWATT drop a lot. In particular, AUC drops by 3.43%, and macro-f1 drops by 17.23% relatively. This verifies that the SA module helps reduce noise and extract better prototypes. We can also see that without attention matrix W^i in SA causes consistent decreases on all metrics. This suggests that predicting dynamic attention matrix for each class is effective, which makes the SA module extract better prototypes. Then we found that without the QA module, Proto-AWATT significantly performs worse. This validates that for a query instance, computing multiple prototype-specific query representations helps obtain accurate distances for ranking, which facilitates the multi-label predictions.

Finally, when removing DT and using a static threshold ($\tau = 0.2$ in the 10-way setting), it causes a slight decrease. This shows that learning dynamic threshold is effective. We further compare DT with two alternative dynamic threshold methods: (1) MS (mean $\pm$ standard deviation of the threshold by cross-validation); (2) a kernel regression (KR)

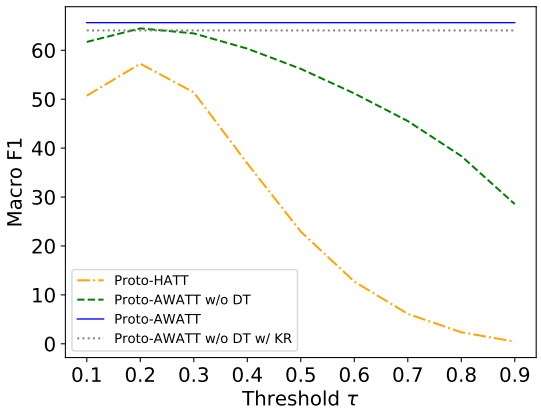


Figure 3: Macro-f1 scores for different thresholds on 10-way 5-shot setting of FewAsp.

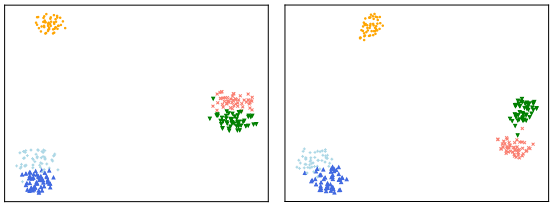
approach which is proposed by Hou et al. (2020) to calibrate the threshold. Comparing with MS and KR, our method slightly outperforms them. This is because DT benefits from reinforcement learning and directly optimizes the evaluation metrics.

Different Encoders We also compare the performances of our method with a strong baseline Proto-HATT when using different encoders to obtain the contextual sequence H. The results are reported in Table 6. The output of pre-trained encoders, i.e., BERT (Devlin et al., 2019) or DistilBERT (Sanh et al., 2019), are directly used as the contextual sequence. We observe that Proto-AWATT significantly outperforms the strong baseline Proto-HATT on all encoders.

Effects of Thresholds As depicted in Figure 3, we analyze the impact of different thresholds on the macro-f1 score during inference. We can see that Proto-AWATT without DT consistently outperforms Proto-HATT in various thresholds. Macro-f1 scores of the two methods are getting worse as τ grows. However, the declines in Proto-HATT are more significant. At $\tau = 0.9$, the macro-f1 of Proto-HATT drops nearly to 0. Proto-AWATT without DT still achieves much higher macro-f1. This indicates that the proposed two attention mechanisms help extract an accurate ranking of prototypes. The ranking is less sensitive to the threshold, which makes our method robust and stable. We also found that learning threshold by DT benefits from a reinforced way, which slightly outperforms KR and the best static threshold.

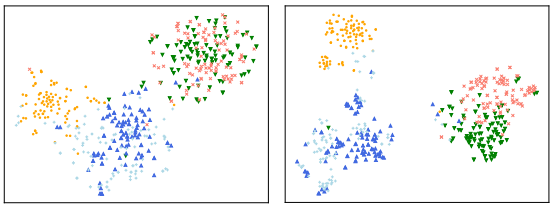
4.5 Visualizations

We further analyze Proto-AWATT by visualizing the extracted representations from the support set and query set, respectively. The representations are visualized by t-SNE (Maaten and Hinton, 2008). To observe the performance in a challenging situation, we choose the testing set from **FewAsp(multi)** as an example.



(a) Proto-HATT (b) Proto-AWATT

Figure 4: Visualization of extracted prototypes for the support set.



(a) Proto-HATT (b) Proto-AWATT

Figure 5: Visualization of extracted representations for the query set.

Support Set Figure 4 presents the visualization of extracted prototypes from two methods. We randomly sample 5 classes and then sample 50 times of 5-way 5-shot meta-tasks for the five classes. Then for each class, we have 50 prototype vectors. We observe that prototype vectors from our approach are more separable than those from Proto-HATT. This further indicates that the SA module can alleviate noise and thus yield better prototypes.

Query Set We randomly sample 5 classes and then sample 20 times of 5-way 5-shot meta-tasks for these classes. Each meta-task has 5 query instances per class. Thus we have $25 \times 20 = 500$ query instances. It is worth noting that our model learns N prototype-specific query representations for each query instance. We choose the representations according to the ground-truth label. However, Proto-HATT only outputs a single representation for a query instance. As depicted in Figure 5, we can see that the representations learned by our method are obviously more separable than those by Proto-HATT. This further reveals that Proto-AWATT can obtain accurate prototype-specific query representations, which contributes to com-

puting accurate distances.

5 Conclusion

In this paper, we formulate the aspect category detection (ACD) task in the few-shot learning (FSL) scenario. Existing FSL methods mainly focus on single-label predictions. They can not work well for the ACD task since a sentence may contain multiple aspect categories. Therefore, we propose a multi-label FSL method based on the prototypical network. Specifically, we design two effective attention mechanisms for the support set and query set to alleviate the noise from both sets. To achieve multi-label inference, we further learn a dynamic threshold per instance by a policy network with continuous action space. Extensive experimental results in three datasets demonstrate that our method outperforms strong baselines significantly.

Acknowledgements

We sincerely thank all the anonymous reviewers for providing valuable feedback. This work is supported by the National Science and Technology Major Project, China (Grant No. 2018YFB0204304).

References

Amit Alfassy, Leonid Karlinsky, Amit Aides, Joseph Shtok, Sivan Harary, Rogerio Feris, Raja Giryes, and Alex M Bronstein. 2019. Laso: Label-set operations networks for multi-label few-shot learning. In *(CVPR)*, pages 6548–6557.

Kelsey Allen, Evan Shelhamer, Hanul Shin, and Joshua Tenenbaum. 2019. Infinite mixture prototypes for few-shot learning. In *(ICML)*, pages 232–241.

Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. 2017. Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In *(KDD)*, page 717–725.

Kai-Hsiang Cheng, Szu-Yu Chou, and Yi-Hsuan Yang. 2019. Multi-label few-shot learning for sound event recognition. In *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–5. IEEE.

Po-Wei Chou, Daniel Maturana, and Sebastian Scherer. 2017. Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution. In *(ICML)*, pages 834–843.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *(NAACL-HLT)*, pages 4171–4186.

Li Fe-Fei et al. 2003. A bayesian approach to unsupervised one-shot learning of object categories. In *(ICCV)*, pages 1134–1141.

Li Fei-Fei, Rob Fergus, and Pietro Perona. 2006. One-shot learning of object categories. *(TPAMI)*, 28(4):594–611.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *(ICML)*, pages 1126–1135.

Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019. Hybrid attention-based prototypical networks for noisy few-shot relation classification. In *(AAAI)*, pages 6407–6414.

Victor Garcia and Joan Bruna. 2018. Few-shot learning with graph neural networks. In *(ICLR)*.

Spyros Gidaris and Nikos Komodakis. 2018. Dynamic few-shot visual learning without forgetting. In *(CVPR)*, pages 4367–4375.

Yiluan Guo and Ngai-Man Cheung. 2020. Attentive weights generation for few shot learning via information maximization. In *(CVPR)*, pages 13499–13508.

Zhen Hai, Kuiyu Chang, and Jung-jae Kim. 2011. Implicit feature identification via co-occurrence association rule mining. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 393–404.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *(EMNLP)*, pages 4803–4809.

Yutai Hou, Yongkui Lai, Yushan Wu, Wanxiang Che, and Ting Liu. 2020. Few-shot learning for multi-label intent detection. *arXiv preprint arXiv:2010.05256*.

Mengting Hu, Shiwan Zhao, Li Zhang, Keke Cai, Zhong Su, Renhong Cheng, and Xiaowei Shen. 2019. CAN: Constrained attention networks for multi-aspect sentiment analysis. In *(EMNLP-IJCNLP)*, pages 4601–4610.

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 437–442.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*.

Kai Li, Yulun Zhang, Kunpeng Li, and Yun Fu. 2020. Adversarial feature hallucination networks for few-shot learning. In *(CVPR)*, pages 13470–13479.

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

Sajad Movahedi, Erfan Ghadery, Heshaam Faili, and Azadeh Shakery. 2019. Aspect category detection via topic-attention network. *arXiv preprint arXiv:1901.01183*.

Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *(ICML)*, pages 2554–2563.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *(EMNLP)*, pages 1532–1543.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *(SemEval 2015)*, pages 486–495.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *(SemEval 2014)*, pages 27–35.

Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. In *(ICLR)*, pages 1–11.

Anthony Rios and Ramakanth Kavuluru. 2018. Few-shot and zero-shot multi-label learning for structured label spaces. In *(EMNLP)*, pages 3132–3142.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Kim Schouten, Onne Van Der Weijde, Flavius Frasincar, and Rommert Dekker. 2018. Supervised and unsupervised aspect category detection for sentiment analysis with co-occurrence data. *IEEE Transactions on Cybernetics*, 48(4):1263–1275.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *(NeurIPS)*, pages 4077–4087.

Qi Su, Kun Xiang, Houfeng Wang, Bin Sun, and Shiwen Yu. 2006. Using pointwise mutual information to identify implicit features in customer reviews. In *International Conference on Computer Processing of Oriental Languages*, pages 22–30.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *(CVPR)*, pages 1199–1208.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *(NeurIPS)*, pages 5998–6008.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *(NeurIPS)*, pages 3630–3638.

Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. 2018. Low-shot learning from imaginary data. In *(CVPR)*, pages 7278–7286.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Wei Xue and Tao Li. 2018. Aspect based sentiment analysis with gated convolutional networks. In *(ACL)*, pages 2514–2523.

Wei Xue, Wubai Zhou, Tao Li, and Qing Wang. 2017. MTNA: A neural multi-task model for aspect category classification and aspect term extraction on restaurant reviews. In *(IJCNLP)*, pages 151–156.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *(COLING)*, pages 2335–2344.

Fang Zhao, Jian Zhao, Shuicheng Yan, and Jiashi Feng. 2018. Dynamic conditional networks for few-shot learning. In *(ECCV)*, pages 19–35.

Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. 2015. Representation learning for aspect category detection in online reviews. In *(AAAI)*, page 417–423.

Models	FewAsp(single)		FewAsp(multi)		FewAsp	
	AUC	F1	AUC	F1	AUC	F1
Proto-AWATT (ours)	0.9701	**80.28**	0.8980	58.89	**0.9206**	**65.65**
w/o SA	0.9304	63.63	0.8854	53.02	0.8890	54.34
w/o attention matrix W^i	**0.9703**	78.08	0.8959	57.42	0.9128	61.68
w/o QA	0.9541	69.61	0.8920	51.51	0.8886	51.19
w/o DT	0.9689	79.46	0.8970	**59.40**	0.9161	64.48
w/o DT w/ KR	0.9695	79.41	**0.9006**	59.13	0.9159	64.06
w/o DT w/ MS	0.9681	78.73	0.8976	59.01	0.9163	64.00

Table 7: Ablation study of the 10-way 5-shot scenario on three datasets.

A Implementation Details

Hyperparameters All baselines and our model are implemented by Pytorch. We initialize word embeddings with 50-dimension GloVe vectors and fine-tune them during the training. All other parameters are initialized by sampling from a normal distribution $\mathcal{N}(0, 0.1)$. The dimension of the hidden state d is 50. The convolutional window size m is set as 3. The optimizer is Adam with a learning rate 10^{-3}. When jointly training the policy network, the learning rate is set to 10^{-4}. In each dataset, we construct four FSL tasks, where $N = 5, 10$ and $K = 5, 10$. And the number of query instances per class is 5. For example, in a 5-way 10-shot meta-task, there are $5 \times 10 = 50$ instances in the support set and $5 \times 5 = 25$ instances in the query set.

Dynamic Threshold (DT) In this module, we first map the state into a vector representation through linear layers. Then the vector is mapped into two separate linear layers with softplus as the activation function. We obtain the parameters of Beta distribution, i.e. a and b, respectively. When training the policy network, a reward is computed based on the $\mathrm{softmax}$ output (i.e. ranking of prototypes). However, the $\mathrm{softmax}$ output is narrow and highly confident, resulting in sparse rewards. Therefore, we exploit a temperature $T = 2$ to make the $\mathrm{softmax}$ output more smooth. In addition, two-stage training is also designed to deal with the sparse rewards. We first train the main network to obtain accurate rankings. Then when learning the policy network, we can gain more meaningful rewards.

Training Details In every epoch, we randomly sample 800 meta-tasks for training. The number of meta-tasks during validation and testing are both set as 600. The average score of meta-tasks are used for evaluation. We employ an early stop strategy if the AUC score of the validation set is not improved in 3 epochs, and the best model is chosen for testing. For all baselines and our model, we report the average testing results from 5 runs, where the seeds are set to [5, 10, 15, 20, 25]. All models are trained on one Tesla P100 GPU with 16GB of RAM.

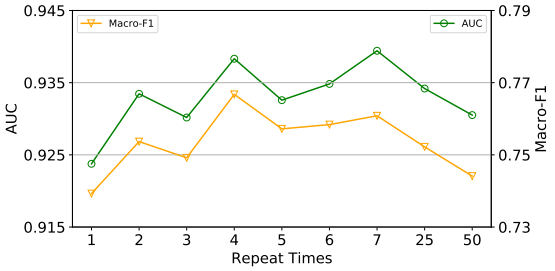


Figure 6: Effects of attention matrix on 5-way 5-shot setting of FewAsp.

B Experimental Results

Ablation Study We display the results of ablation study on three datasets in Table 7.

Effects of Attention Matrix To explore the effects of the condition on the attention matrix, we compare the performances of Proto-AWATT by setting different repeat times e_M in Eq. 3. The results are displayed in Figure 6. We can see that by repeating more times of the common aspect vector, the AUC and macro-f1 score both outperform the results of setting $e_M = 1$. As e_M grows, the performances are improved. However, when setting e_M as 25 or even 50, the performances decline. A possible reason is that the model tends to overfit the training classes.