Exploiting Careful Design of SVM Solution for Aspect-term Sentiment Analysis

Hanfeng Liu[†], **Minping Chen**[†], **Zhenya Zheng**[§,†], **Zeyi Wen**[†‡*]
[†]The Hong Kong University of Science and Technology (Guangzhou)
[‡]The Hong Kong University of Science and Technology
[§]South China University of Technology
[†]{hliu174, mchen779}@connect.hkust-gz.edu.cn
[§,†]202220145182@mail.scut.edu.cn
[†‡]wenzeyi@ust.hk

Abstract

Aspect-term sentiment analysis (ATSA) identifies fine-grained sentiments towards specific aspects of the text. While pre-trained language models (PLMs) have set the state-of-the-art (SOTA) for ATSA, they are resource-intensive due to their large model sizes, restricting their wide applications to resource-constrained scenarios. Conversely, conventional machine learning methods, such as Support Vector Machines (SVMs), offer the benefit of less resource requirement but have lower predictive accuracy. This paper introduces an innovative pipeline, termed SVM-ATSA, which bridges the gap between the accuracy of SVM-based methods and the efficiency of PLM-based methods. To improve the feature expression of SVMs and better adapt to the ATSA task, SVM-ATSA decomposes the learning problem into multiple view subproblems, and dynamically selects as well as constructs features with reinforcement learning. The experimental results demonstrate that SVM-ATSA surpasses SOTA PLM-based methods in predictive accuracy while maintaining a faster inference speed and significantly reducing the number of model parameters.

1 Introduction

Aspect-term sentiment analysis (ATSA) is one of the subtasks in aspect-based sentiment analysis (ABSA) (Pontiki et al., 2014), which aims to predict fine-grained sentiment polarities given different aspect terms in a sentence. Specifically, given a sentence $S = w_1, w_2, \ldots, w_n$, and let $A = a_1, a_2, \ldots, a_k$ be the set of aspect terms extracted from, the goal of this task is to determine the polarity of each aspect term. For instance, consider the sentence "*The battery life of this laptop is amazing, but the screen is too dim.*" The input of the pipeline should be the original sentence and the identified aspect terms "*battery life*" and "*screen*". Thus, the output would be $(battery\ life, positive)$ and $(screen, negative)$. It is important to note that different data samples can share one original sentence.

With the emergence of pre-trained language models (PLMs), researchers have proposed numerous PLM-based methods, which are the SOTA methods for the ATSA task (Tian et al., 2021; Wang et al., 2021; Chen et al., 2022; Yang and Li, 2024). However, their large model sizes make them resource-intensive, restricting their wide applications to resource-constrained scenarios.

A recent work of Tarzanagh et al. (2023) demonstrates a formal equivalence between the optimization geometry of self-attention and a hard-margin SVM problem that uses linear constraints on the outer-products of token pairs to separate the optimal input tokens from non-optimal tokens. Inspired by this work, this paper revisits SVMs for the ATSA task and proposes an SVM-ATSA pipeline to address the poor accuracy problem in SVM-based methods and the resources-intensive issue in PLM-based methods.

Unlike neural networks can automatically learn the expressive feature representation, the feature expression ability of a single SVM is limited, thus tending to result in poor accuracy. To tackle this challenge, SVM-ATSA exploits a multiple view subproblem construction technique. Specifically, the learning problem is first decomposed into multiple subproblems according to the aspect terms (i.e., using clustering to construct each subproblem with similar aspect terms) which may be iteratively divided when further decomposition is needed (e.g., due to the complexity of the problems). Moreover, SVM-ATSA automatically considers data balancing when constructing the subproblems, since each subproblem is handled by a single SVM to better fit the subproblem and the data imbalance problem can lead to poor model quality. Then, feature selection and construction are performed for each

*Corresponding author

Findings of the Association for Computational Linguistics: EACL 2024, pages 5897–5906

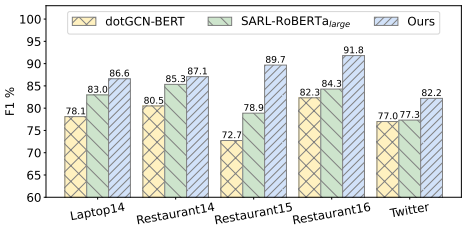


Figure 1: F_1 comparison on the ATSA task

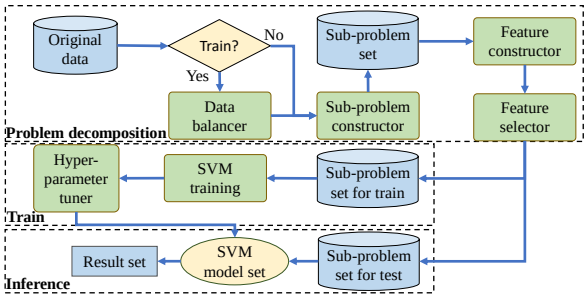


Figure 2: SVM-ATSA training and inference pipeline

base SVM model in SVM-ATSA, with reinforcement learning techniques to dynamically customize feature sets for each subproblem. After that, SVM-ATSA automatically sets the hyper-parameter configuration of the SVM model for each subproblem. Finally, the overall model quality is evaluated and may get further improved with zeroth order optimization. We formulate the learning problem of SVM-ATSA, and theoretically demonstrate that SVM-ATSA has the same training error bound to the single SVM-based approach. Our experimental results on the aspect-term sentiment analysis (ATSA) show that SVM-ATSA outperforms the existing SOTA approaches based on RoBERTa$_{large}$ and DeBERTa (Wang et al., 2021; Yang and Li, 2021), which are summarized in Figure 1.

To summarize, our major contributions in the work are as follows.

- We propose SVM-ATSA for the ATSA task which can automatically tune the whole pipeline including feature construction and feature selection. In SVM-ATSA, an effective technique based on reinforcement learning is proposed for dynamic feature selection.
- We support SVM-ATSA with techniques including multiple view subproblem construction while considering the complexity of the ATSA task in SVM training.
- Our results show that SVM-ATSA can achieve higher accuracy than the SOTA models for ATSA, and enjoys a lower training time complexity. The model trained has 10 times fewer parameters than the SOTA models.

2 Our SVM-ATSA Solution

In this section, we elaborate on the details of SVM-ATSA. The overview of SVM-ATSA is shown in Figure 2, which summarizes the key components of SVM-ATSA. It decomposes the learning problem of ATSA into multiple subproblems based on clustering so that the feature expression of SVM can be boosted. Besides, SVM-ATSA ensures the data in each subproblem is balanced to guarantee the model quality. Through this subproblem construction, SVM-ATSA trains a number of SVM models in the learning pipeline, instead of restricting to using only one SVM model (Kiritchenko et al., 2014; Wen et al., 2018) or simply decomposing the learning problem (Graf et al., 2004; You et al., 2015). To further enhance the model quality of SVM-ATSA, we propose techniques including feature construction on the raw features and feature selection to represent each subproblem, and automatically setting hyper-parameter configurations. We power the training process for SVM-ATSA with zeroth order optimization to calibrate potential flaws for different components in the pipeline. At last, all the SVM models are used collectively to make predictions. The whole pipeline of SVM-ATSA is elaborated in greater detail next.

2.1 Problem Definition

Before introducing the technical details of SVM-ATSA, we formulate its learning problem first. Although our SVM-ATSA solution consists of multiple components, we can formulate it as an optimization problem. Formally, let $\mathcal{D}_T$ and $\mathcal{D}_V$ represent the training and validation datasets, respectively. The datasets are further divided into k subsets $\mathcal{D}_T^1, \ldots, \mathcal{D}_T^k$ and $\mathcal{D}_V^1, \ldots, \mathcal{D}_V^k$, respectively. In the learning pipeline, the hyper-parameters need to be configured including k, $\mathcal{F}$ which is a set of features, and Λ for the configuration of SVMs. Thus, the search space can be written as $\Theta = k \times \mathcal{F} \times \Lambda$. Thus, the goal is to minimize the objective below:

$$\arg\min_{k \in \mathbb{N}^+, \boldsymbol{\theta}^i \in \Theta} \mathcal{L}(k, \boldsymbol{\theta}^i, \mathcal{D}_T^i, \mathcal{D}_V^i) = \sum_{i=1}^{k} \frac{|\mathcal{D}_V^i|}{|\mathcal{D}_V|} \mathcal{L}_0(\boldsymbol{\theta}^i, \mathcal{D}_T^i, \mathcal{D}_V^i) \quad (1)$$

where $\mathcal{L}_0(\boldsymbol{\theta}^i, \mathcal{D}_T^i, \mathcal{D}_V^i)$ is the empirical loss of the i-th SVM on $\mathcal{V}^i$, $\boldsymbol{\theta}^i = \{f^i, \lambda^i\}$, f^i denotes the features for the i-th SVM, λ^i denotes the hyperparameter configuration for the SVM, $\frac{|\mathcal{D}_V^i|}{|\mathcal{D}_V|}$ represents the importance of the i-th SVM, and $\mathbb{N}^+$ is positive integers.

The gradient-based methods can resolve the above learning problem. The derivative of the learning problem over $\boldsymbol{\theta}^i$ is $\sum_{i=1}^{k} \frac{|\mathcal{D}_V^i|}{|\mathcal{D}_V|} \cdot \frac{\partial \mathcal{L}_0(\boldsymbol{\theta}^i, \mathcal{D}_T^i, \mathcal{D}_V^i)}{\partial \boldsymbol{\theta}^i}$. Since $\boldsymbol{\theta}^i = \{f^i, \lambda^i\}$, the derivative can be written as $\sum_{i=1}^{k} \frac{|\mathcal{D}_V^i|}{|\mathcal{D}_V|} \cdot (\frac{\partial \mathcal{L}_0(\boldsymbol{\theta}^i, \mathcal{D}_T^i, \mathcal{D}_V^i)}{\partial f^i \cdot \lambda^i} + \frac{\partial \mathcal{L}_0(\boldsymbol{\theta}^i, \mathcal{D}_T^i, \mathcal{D}_V^i)}{f^i \cdot \lambda^i} + \frac{\partial \mathcal{L}_0(\boldsymbol{\theta}^i, \mathcal{D}_T^i, \mathcal{D}_V^i)}{\partial \lambda^i \cdot f^i})$. As $\Theta = \mathcal{F} \times \Lambda$ contains conditional variables (e.g., γ is used only for the RBF kernel), no closed form can be used to compute the gradients. In SVM-ATSA, we apply Hyperband (Li et al., 2018) to find the solution for the optimization problem, which can work on the non-continuous search space with conditional variables.

Next, we introduce each component in the SVM-ATSA pipeline, including subproblem construction, feature construction, and feature selection. We also introduce the training and inference process of SVM-ATSA and provide some analysis of it.

2.2 Subproblem Construction

Here, we elaborate on the subproblem construction component of SVM-ATSA, which aims to improve the feature presentation of the SVM model. Before presenting the technical details, we provide the motivation of the need for subproblem construction.

2.2.1 Motivation of Subproblem Construction

A single SVM model may suffer from poor predictive accuracy for complex problems due to the its limited parameters and the unpowerful feature representation. In SVM-ATSA, we construct multiple view subproblems from the original problem, where each subproblem is much smaller than the original one, and meanwhile, consider the balancing of the data distribution. Hence, our SVM-ATSA solution can utilize more SVMs to handle complex problems in the ATSA task and improve the feature expression of single SVM. We introduce the complexity of the ATSA task and provide examples to support our motivation for the subproblem construction in the following.

Complexity of the ATSA task: Problem decomposition is critical for complex problems like the ATSA task due to fine-grained information in the analysis. In AutoSVM, each subproblem is constructed for several similar aspect terms such as "service" and its similar aspect terms. This is because the adjectives used to describe similar aspect terms tend to be similar. For example, "tasty" and "delicious" can be used for the food aspect, while "costly" and "cheap" describe the cost aspect of the restaurant. By dedicating a subproblem to similar aspect terms, SVM-ATSA can learn knowledge of similar aspect terms and consider adjectives of similar meaning, and meanwhile, more SVMs (and hence more learnable parameters) can be used to fit the problem. Next, we explain the subproblem construction process summarized in Algorithm 1.

Algorithm 1: Subproblem Construction

Input: training data $\mathcal{D}_T$, # of subproblems k, balancing blc
Output: a set of subproblems $\{\mathcal{D}_T^1, \ldots, \mathcal{D}_T^k\}$

1 $\mathbb{L}_T \leftarrow$ PerformClustering($\mathcal{D}_T{}'$, k);
2 initialize $\{\mathcal{D}_T^1, \ldots, \mathcal{D}_T^k\}$ as empty sets;
// Allocate the instances to the subproblems.
3 **for** *the i-th instance in $\mathcal{D}_T$* **do**
4 $t \leftarrow \mathbb{L}_T[i]$;
5 add the i-th instance to subset $\mathcal{D}_T^t$;
// Need data balancing.
6 **if** $blc = true$ **then**
7 BalanceSubproblems($\mathcal{D}_T^1, \ldots, \mathcal{D}_T^k$)/*cf. Operation 1*/;
8 **return** $\{\mathcal{D}_T^1, \ldots, \mathcal{D}_T^k\}$

2.2.2 Subproblem Construction Process

Given the complex ATSA task, how to divide it into several subproblems while ensuring model quality is challenging. In SVM-ATSA, we use the clustering algorithm as a good heuristic. Let $\{a_1, a_2, a_3, \ldots, a_n\}$ denote the aspect terms in the training dataset $\mathcal{D}_T$, where a_j corresponds to the aspect term of the j-th training instance. We use k-means to cluster the training instances based on the word embedding of the aspect terms. After the clustering, the training dataset $\mathcal{D}_T$ is divided into k subproblems denoted by $\mathcal{D}_T^1, \mathcal{D}_T^2, \mathcal{D}_T^3, \ldots, \mathcal{D}_T^k$. There are more technical details on the decomposition process. For example, when the data among a subproblem is unbalanced, we need to perform data balancing to ensure the quality of the model (cf. Line 7 of Algorithm 1). As the subproblems are

independent, SVM-ATSA can train a customized SVM model for each subproblem to achieve high predictive accuracy. When a test instance is fed in, we can easily assign it to a subproblem with the cluster centers of the subproblems.

2.2.3 Data Balancing

A significant obstacle encountered in clustering the training data to subproblems is the potential for data imbalance within each subproblem. For instance, it is common to find a subproblem with a disproportionate number of positive reviews compared to neutral ones. Such imbalance can detrimentally affect the performance of the SVM models derived from this data. To address this issue and ensure that each subproblem is characterized by a balanced dataset, we implement a data augmentation strategy that enriches the minority classes. The pseudo-code for this balancing technique is detailed in Operation 1. Initially, we identify the largest class, which serves as a benchmark (eg., the class of positive). Our goal is to augment the size of the smaller classes (e.g., neutral and negative) to match the benchmark class. To achieve this, we randomly select and replicate instances from the underrepresented class (e.g., the negative class) within the training dataset. This resampling process continues until the proportion of classes in the subproblem is the same as in the training set. Within SVM-ATSA, the decision to employ this resampling method is determined by a learnable parameter for each subproblem. This parameter is adjustable via the *blc* variable, as specified in Line 6 of Algorithm 1.

Operation 1: BalanceSubproblems($\mathcal{D}^1, \ldots, \mathcal{D}^k$)

Input: Datasets $\mathcal{D}^1, \ldots, \mathcal{D}^k$

1 $\mathcal{D} = \mathcal{D}^1 \bigcup \cdots \bigcup \mathcal{D}^k$;
2 **for** *each* j *in* $\{1, ..., k\}$ **do**
3 $\quad y \leftarrow$ GetMajorityClass($\mathcal{D}^j$);
4 $\quad \mathcal{D}^j_y, \mathcal{D}_y \leftarrow$ GetDataOfClassY($\mathcal{D}^j$, $\mathcal{D}$, y);
5 $\quad$ **for** *an instance* t *of class* y' *in* $\mathcal{D}$ **do**
6 $\quad\quad \mathcal{D}^j_{y'}, \mathcal{D}_{y'} \leftarrow$ GetDataOfClassY($\mathcal{D}^j$, $\mathcal{D}$, y');
7 $\quad\quad \mathcal{P}_{label} \leftarrow \frac{|\mathcal{D}^j_{y'}|}{|\mathcal{D}^j_y|}, \mathcal{P}'_{label} \leftarrow \frac{|\mathcal{D}_{y'}|}{|\mathcal{D}_y|}$;
// add t to the subproblem.
8 $\quad\quad$ **if** $\mathcal{P}_{label} < \mathcal{P}'_{label}$ **then**
9 $\quad\quad\quad \mathcal{D}^j \leftarrow \mathcal{D}^j \bigcup \{t\}$;

We would like to clarify that data augmentation is performed after clustering, ensuring it does not affect the computation of cluster centers in Algorithm 1. Additionally, data augmentation does not duplicate instances, as examples in different clusters do not overlap. Even when instances share the same original sentence, the features extracted for them differ significantly due to the syntactic parser. Consequently, data augmentation enhances the overall performance of the model. There might be an illusion of inconsistency between the subproblem construction and data balancing. The inconsistency in cluster divisions between the original training data and augmented data arises from our dual objectives of reducing noise and enhancing generalization. Firstly, the primary goal of using word embeddings of aspect terms for clustering is to ensure that different categories of aspects are grouped separately. This method helps in reducing noise in the model input. For instance, terms related to food aspects such as "*poor taste*" or "*unpleasant smell*" are clustered together, separate from service-related aspects. This prevents irrelevant terms from introducing noise into the training process of specific aspect models. Secondly, to address the issue of having too few samples in certain subproblem, we incorporate data balancing by using samples from other subproblems. This helps ensure that each subproblem has enough data to represent sentiment terms adequately. Subproblems with fewer samples might not provide enough data for the model to learn effectively. By augmenting these subproblems with samples from other subproblems, we introduce additional sentiment terms that are generally applicable, such as "*bad*" or "*excellent*".

2.3 Feature Construction

After obtaining the subproblems, SVM-ATSA represents each subproblem with a customized set of features, to ensure the model quality. Although we can customize the features by feature selection for each subproblem. However, due to the complexity of some problems, SVM-ATSA enhances original features with automatic feature construction, which can construct new features from the original data to better mine the deeper correlation information within the data. After that, we utilize a feature selection method to choose new features that can better express the subproblem. In what follows, we describe the automatic feature construction and feature selection processes in detail.

In SVM-ATSA, feature construction uses the in-

teraction information of the original features of the dataset to generate new features. We employ various methodologies for feature construction, specifically focusing on transformation and aggregation methodologies. SVM-ATSA employs the following six construction methods. (i) Standardization: transforms the numerical type feature into a standard normal distribution. (ii) Discretization: divides the numerical feature into several bins or intervals. (iii) Addition: adds two numeric features to obtain the sum as a feature. (iv) Subtraction: subtracts two numeric features to obtain the difference as a feature. (v) Multiplication: multiplies two numeric features to obtain their product as a feature. (vi) Division: computes the division between two numeric features and uses it as a feature. These methods help generate features of various aspects. We also provide flexibility in feature construction, where the construction methods can be customized or extended based on the specific requirements of the practitioners or the properties of the datasets and machine learning models.

2.4 Feature Selection

In SVM-ATSA, feature selection is employed in two stages: before and after automatic feature construction. For the first stage of feature selection, we filter non-informative features before automatic feature construction (cf. Section 2.3), which can reduce the number of features to be generated, thus mitigating time costs. For this part of feature selection, we sort the features using their relevance score to the corresponding subproblem. The relevance score of a feature f in the i-th subproblem is measured by chi-squared: $chi(f) = \frac{M(M_f^i M_{\bar{f}}^{\bar{i}} - M_{\bar{f}}^i M_f^{\bar{i}})^2}{(M_f^i+M_{\bar{f}}^i)(M_f^i+M_f^{\bar{i}})(M_{\bar{f}}^i+M_{\bar{f}}^{\bar{i}})(M_{\bar{f}}^i+M_f^{\bar{i}})}$, where M denotes the size of the whole learning problem; f denotes a feature; M_f^i (resp. $M_{\bar{f}}^i$) represent the number of training instances having nonzero (resp. zero) values at the feature f in the i-th subproblem; $M_f^{\bar{i}}$ represents the number of training instances with nonzero values at the feature but not in the i-th subproblem; $M_{\bar{f}}^{\bar{i}}$ is the number of training instances neither having nonzero values in the feature f nor belonging to the i-th subproblem.

2.4.1 QBSO-FS for Feature Selection

For the second stage of feature selection, we select the valuable features after automatic feature construction to enhance the model quality. In this stage of feature selection, we exploit the QBSO-FS (Sadeg et al., 2019) algorithm, which uses a hybrid version of BSO (Sadeg et al., 2015) with Q-learning (Watkins, 1989) for generating feature subsets and a classifier to evaluate them. Here we adopt the SVM model as the evaluator to instruct the value of state and action for the Q-learning. Specifically, QBSO-FS replaces the local search executed by bees in the BSO algorithm with the Q-learning algorithm. In this context, each bee is considered an intelligent agent that collects experience during its search process and benefits from the experiences of other bees. The states constituting the environment are feature subsets of all the possible solutions within the bee's neighborhood, where a solution is represented as a Boolean vector indicating which features belong to the feature subset. During the intelligent agent decision-making process, an action involves adding or removing a feature from the current subset, essentially flipping the state of a feature in the current solution. The primary performance metric is classification accuracy, with the size of the feature subset being a secondary criterion. The reward r_t associated with a pair (s_t, a_t) is computed based on the classification accuracy. Formally, let $A_t = \{a_{t_1}, a_{t_2}, \ldots, a_{t_m}\}$ represent the set of possible actions in the current state s_t, and s_{t+1} denotes the next state reached after selecting an action from A_t. If we use $\mathcal{A}(s)$ to denote the classification accuracy obtained with the feature subset represented by s, then the reward r_t is given by:

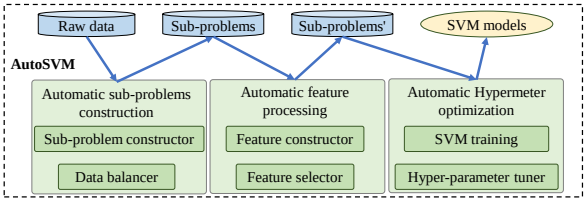


Figure 3: The overview of SVM-ATSA

$$r_t = \begin{cases} \mathcal{A}(s_{t+1}), & if\ \mathcal{A}(s_t) < \mathcal{A}(s_{t+1}) \\ \frac{1}{2} \times \mathcal{A}(s_{t+1}), & if\ \mathcal{A}(s_t) = \mathcal{A}(s_{t+1}) \text{ and } |s| > |s_{t+1}| \\ -\frac{1}{2} \times \mathcal{A}(s_{t+1}), & if\ \mathcal{A}(s_t) = \mathcal{A}(s_{t+1}) \text{ and } |s| \leq |s_{t+1}| \\ \mathcal{A}(s_{t+1}) - \mathcal{A}(s_t)), & if\ \mathcal{A}(s_t) > \mathcal{A}(s_{t+1}) \end{cases} \quad (2)$$

where $|s|$ denotes the size of the set s. Intuitively, the reward function aims to incentivize actions that lead to improvements in classification accuracy while considering the size of the feature subset.

2.5 Training and Inference Process

SVM-ATSA can automatically adjust the feature selector, hyper-parameter configuration tuner, and SVM trainer in the learning pipeline with the zeroth order optimization, using the current overall model quality information. The overview of the automatic parts of SVM-ATSA is shown in Figure 3. Thus, the SVM model trained in SVM-ATSA is the result of a combination of well-tuned subproblem construction, carefully selected features, and properly configured hyper-parameters. Next, we present the details of automatically selecting features, setting hyper-parameters and training SVMs.

Feature Selection in the Training: A learning problem can have many relevant features, but some work better for one subproblem than another. For instance, the SVM sentiment analysis classifier may only use surface and word similarity features for the first aspect, while it may use all the features for the second aspect. Therefore, different subproblems require different sets of features, which means we need to customize the features. To do this, we need to find the best feature combination for each subproblem. This paper proposes a solution that ranks all the features, constructs new features based on the good ones, and selects the best features for each subproblem.

Automatic Hyper-parameter Configuration: The quality of SVM models is significantly impacted by their hyper-parameters, which determine the SVM's hyper-plane space. In SVM-ATSA, we automatically configure the hyper-parameters instead of manually setting them. We use Hyperband (Li et al., 2018) to select the kernel type and hyper-parameters for each SVM model. This is done by training a machine learning model using the history of hyper-parameters to guide the search for the best kernel and corresponding hyper-parameters. Additionally, SVM-ATSA learns whether to balance the training instances for the SVMs. Once the training instances are represented as feature vectors and hyper-parameters are set, an SVM model is trained for each subproblem.

Inference of SVM-ATSA: When presented with a test instance $\boldsymbol{x}_t$ with the target value y_t, SVM-ATSA assigns $\boldsymbol{x}_t$ to the SVM model whose subproblem cluster center is most similar to it. Relevant features are then selected using the techniques outlined in Section 2.3, and the SVM model predicts a target value (e.g., positive class).

2.6 Analysis of SVM-ATSA

Here we analyze the training time complexity and the training error bound of SVM-ATSA.

2.6.1 Training Time Complexity

SVMs generally have lower training time complexity than the BERT-based solutions. Next, we provide the time complexity analysis for SVM-ATSA on the ATSA task, in comparison with RoBERTa$_{large}$ (Wang et al., 2021) which builds one of the existing SOTA solutions for ATSA. We let α be the sentence length on average, m be the training dataset size, d be the dimensionality of training data, t be the number of epochs, L be the number of hidden layers, H as the number of heads of self-attention and R_d as the representation dimension (768 in BERT specifically). For the BERT-based model (i.e., RoBERTa), the training time complexity can be calculated as $\mathcal{O}(\alpha^2 \cdot R_d \cdot H \cdot L \cdot t \cdot m)$. However, SVMs have the time complexity of $\mathcal{O}(t \cdot m \cdot d)$ for the SVM training (Keerthi et al., 2001), which indicates SVM-ATSA has a much lower training time complexity than the existing SOTA solutions.

2.6.2 Training Error Bound of SVM-ATSA

SVM-ATSA employs multiple SVMs to fit the learning problem, while the single SVM-based solution tends to handle complex tasks poorly (Pontiki et al., 2014). Here we demonstrate SVM-ATSA has the same training error bound.

Theorem 2.1 (Training Error Bounds of SVM-ATSA)**.** *Let h_S be the hypothesis returned by SVMs for a sample S, and let $N_{SV}(S)$ be the number of support vectors that define h_S. Then,*

$$\mathbb{E}_{S\sim\mathcal{D}^m}\left[R(h_S)\right] \leq \mathbb{E}_{S\sim\mathcal{D}^m}\left[\frac{N_{SV}(S)}{m+1}\right].$$

As we can see from the theorem, the training error bound of SVM-ATSA depends on the number of support vectors and the training set size, which is similar to the original SVMs. In other words, the training error bound of SVM-ATSA is theoretically the same as that of the original SVMs.

3 Experimental Study

Here, we present our experimental setup, the results of different methods on the ATSA task, and some analysis experiments.

Table 1: Details of the datasets for the ATSA task

datasets		#positive	#negative	#netural	#total
Laptop14	Train	987	866	460	2313
	Test	341	128	169	638
Restaurant14	Train	2164	807	637	3608
	Test	728	196	196	1120
Restaurant15	Train	907	256	36	1199
	Test	326	182	34	542
Restaurant16	Train	1235	439	69	1743
	Test	467	116	30	613
Twitter	Train	1561	1560	3127	6248
	Test	173	173	346	692
MAMS	Train	3380	2764	5042	11186
	Test	400	329	607	1336

3.1 Experimental Setup

SVM-ATSA was implemented in Python and the code is available at [URL omitted due to anonymous reviews]. We used k-means for the subproblem construction, although other methods for subproblem construction can be easily plugged into SVM-ATSA. The experiments were conducted on a workstation running Linux with an Intel(R) Xeon(R) Platinum 8336C CPU @ 2.30GHz 128 core CPU and 128GB main memory, and a NVIDIA GeForce RTX 4080 GPU with 16GB memory. The code of our paper can be found in: `https://github.com/Kurt-Liuhf/absa-svm`

Datasets. We used six public ATSA datasets for evaluation: Table 1 provides the details of the six datasets for the ATSA task (Kiritchenko et al., 2014), covering *Laptop*, *Restaurant*, *Twitter*, and *MAMS*. The polarity including positive, negative, and neutral of each aspect is available in the ATSA task, rather than the polarity of the whole review.

SOTA Models: PLM-based models have become predominant in the ATSA task. As a result, we have selected four of the most recent PLM-based models as our baseline models. In these baseline models, some of them combine Graph Convolutional Network (GCN) and BERT, such as dotGCN-BERT (Chen et al., 2022) and TGCN-BERT (Tian et al., 2021). Some of them improve the model performance by employing the Span-based Anti-bias aspect Representation Learning (SARL) framework, such as SARL-RoBERTa (Wang et al., 2021) and SARL-DeBERTa, while LSA-X-DeBERTa (Yang and Li, 2021, 2024) attempts to address the task via fine-grained utilization of the sentiment information. We do not compare with DPL-BERT (Zhang et al., 2022) and TF-BERT (Zhang et al., 2023) as their data statistics are different from other baselines and their predictive accuracy is worse than other models such as SARL and LSA-X-DeBERTa. In SVM-ATSA, the features were extracted from surface, parse, word similarity, and sentiment lexicon following the existing conventional machine learning approach for this task (Kiritchenko et al., 2014). The baseline results are obtained from the original papers of the baselines. We believe the model parameters yielding the results in the original papers have been carefully tuned, ensuring a fair comparison.

Hyper-parameters and dictionaries: The number of subproblems was selected from 1 to 35; C was chosen from 1 to 2^{20}. The kernel functions include RBF, polynomial, Sigmoid, and linear kernels. The polynomial kernel's degree was in 1 to 100. The RBF kernel's γ was in 10^{-3} to 10 divided by the training dataset size. The features were selected automatically for each subproblem, with the automatic feature construction and feature selection described in Section 2.3. In the experiments, we used the following common lexicon dictionaries (Kiritchenko et al., 2014): *hashtag sentiment*, *tweet sentiment*, *sentiment140*, *NRC emotion*, *MPQA subjectivity*, *Amazon laptop word-aspect association* and *Yelp restaurant word–aspect association*. We extract sentiment lexicon features for SVM-ATSA based on the above eight dictionaries, following similar settings as the existing SVM-based approach (Pontiki et al., 2014).

3.2 Predictive Accuracy Comparison

Table 2 and Table 3 shows the results of different methods on the ATSA task. SVM-ATSA outperforms the existing SOTA models, even those using DeBERTa, and achieves SOTA results among all these six datasets. This result is encouraging, because SVM-ATSA is a smaller model that does not need pre-trained models (e.g., BERT) or extra labeled data (e.g., DPL-BERT), meanwhile SVM-ATSA uses much fewer computation resources. We also report the Macro-F_1 scores among these models, and the findings are similar. Overall, SVM-ATSA consistently outperforms all the existing solutions across these six tasks, achieving an accuracy or F_1 score improvement up to around 6% over the SOTA solution.

3.3 Analysis Experiments

In this set of experiments, we investigate the inference efficiency and analyze the model size of SOTA solutions and our SVM-ATSA solution for

Table 2: Accuracy and Macro-F_1 comparison. Due to different preprocess ways, data statistics of the *Restaurant15*, *Restaurant16* and *MAMS* datasets used in LSA-X-DeBERTa are not the same as in other baselines. We run SVM-ATSA for both versions of datasets for fair comparison (see Table 3).

Models	Laptop14		Restaurant14		Restaurant15		Restaurant16		Twitter		MAMS	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
dotGCN-BERT (Chen et al., 2022)	81.03	78.10	86.16	80.49	85.24	72.74	93.18	82.32	78.11	77.00	84.95	84.44
TGCN-BERT (Tian et al., 2021)	81.97	78.71	/	/	/	/	/	/	78.03	77.31	83.68	83.07
SARL-RoBERTa (Wang et al., 2021)	85.42	82.97	88.21	82.44	88.19	73.83	94.62	81.92	78.03	77.32	/	/
SARL-RoBERTa$_{large}$ (*Wanget al.*, 2021)	85.74	82.97	90.45	85.34	91.88	78.88	95.76	84.29	78.32	77.32	/	/
SARL-DeBERTa (Wang et al., 2021)	83.32	79.95	86.69	78.91	86.53	69.73	93.31	80.13	/	/	82.03	81.84
LSA-X-DeBERTa (Yang and Li, 2021, 2024)	86.46	84.41	90.98	87.02	/	/	/	/	77.17	76.45	/	/
SVM-ATSA (ours)	**88.24**	**86.60**	**91.16**	**87.07**	**93.91**	**89.67**	**95.76**	**91.80**	**83.38**	**82.20**	**87.20**	**86.77**

Table 3: Accuracy and Macro-F_1 comparison.

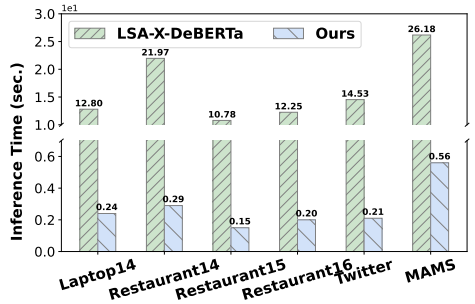
Models	Restaurant15		Restaurant16		MAMS	
	Acc	F1	Acc	F1	Acc	F1
LSA-X-DeBERTa	91.85	81.29	95.61	84.87	86.38	85.97
SVM-ATSA (ours)	**92.41**	**84.07**	**95.72**	**92.40**	**86.98**	**86.47**

the ATSA task. Besides, we investigate the impact of feature processing and data balancing.

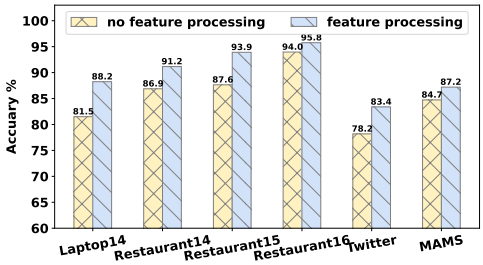
Inference efficiency: Figure 4a shows the inference efficiency of SVM-ATSA and the SOTA PLM-based method *LSA-X-DeBERTa*. The efficiency of SVM-ATSA is 40 to over 70 times faster than the LSA-X-DeBERTa approach. In the *Twitter* dataset, SVM-ATSA only took 0.21 seconds while LSA-X-DeBERTa needed about 14.53 seconds. This is an important property of SVM-ATSA which produces better predictive accuracy to the SOTA PLM-based solutions, while faster in inference.

Model size for the ATSA task: Here, we measure the model size by the number of parameters, which shows that SVM-ATSA has a significantly smaller number of parameters than the existing SOTA models for the ATSA task. Specifically, our solution only requires fewer than 10M parameters on the *Laptop14* and *Restaurant14* datasets, while the BERT-based models have over 110M parameters. For larger BERT-based models, even more learnable parameters need to be introduced. For example, the RoBERTa$_{base}$ requires around 125M and the RoBERTa$_{large}$ requires 355M parameters. SVM-ATSA can outperform the SOTA solutions both in accuracy and F_1, while reducing the number of parameters by over 10 times.

The impact of feature processing: To study the importance of feature processing which includes feature construction and feature selection introduced in Section 2.3 and Section 2.4 respectively, we conducted the ablation experiment of feature processing. Figure 4b shows the results which demonstrate the feature processing is crucial in boosting the predictive accuracy.



(a) Inference efficiency



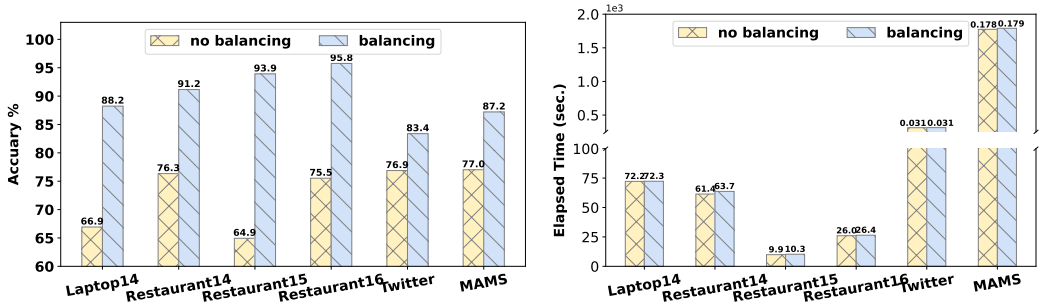
(b) Feature processing

Figure 4: Inference efficiency comparison and the effect of feature processing

Balancing subproblems: We also conducted experiments to investigate the effect of data balancing in subproblem construction. The results are shown in Figure 5. As we can see from the results, the balancing process took only a very small amount of time compared to the whole subproblem construction process in the SVM-ATSA training. This small amount of time brings a significant boost in the model quality for all ATSA datasets. For example, in the *Laptop14* dataset, the accuracy is increased significantly by balancing the subproblems.

4 Related Work

In this section, we review the work of SVM-based methods and PLM-based methods for the aspect-term sentiment analysis (ATSA) task.



(a) *Accuracy comparison* (b) *Efficiency comparison*

Figure 5: The effect of data balancing in subproblem construction

SVM-based ATSA methods: SVMs have been used to tackle the aspect term sentiment analysis (ATSA) task (Pontiki et al., 2014; Kiritchenko et al., 2014; Wen et al., 2021), by utilizing lexical, surface, semantic, and sentiment lexicon features. However, their predictive accuracies are much lower than the PLM-based models.

PLM-based ATSA methods: A series of studies based on pre-trained language models such as BERT, RoBERTa$_{large}$ or DeBERTa have been dedicated to solving the ATSA task (Chen et al., 2022; Wang et al., 2021; Yang and Li, 2021). For example, dotGCN-BERT (Chen et al., 2022) and TGCN-BERT (Tian et al., 2021) combine Graph Convolutional Network (GCN) with BERT. SARL (Wang et al., 2021) proposes a span-based anti-bias aspect representation learning framework which eliminates the sentiment bias in the aspect embedding through adversarial learning against the prior sentiment of the aspects. TF-BERT (Zhang et al., 2023) considers the consistency of multi-word opinion expressions at the span-level. DPL-BERT (Zhang et al., 2022) enhances a pseudo-label framework to leverage the coarser-grained SA labels to assist the ATSA task. LSA-X-DeBERTa (Yang and Li, 2021, 2024) proposes a local sentiment aggregation (LSA) paradigm by constructing a differential-weighted sentiment aggregation window to model aspect sentiment coherency.

Although these methods achieve good accuracy, they have a large model size which restricting their wide applications to resource-constrained scenarios. Our proposed SVM-ATSA solution outperforms all the existing SOTA models, while having a smaller model size and faster inference efficiency.

5 Conclusion

We have proposed SVM-ATSA for the aspect term sentiment analysis (ATSA) task. By effectively decomposing the ATSA task into multiple subproblems and employing dynamic feature selection through reinforcement learning, SVM-ATSA successfully reconciles the need for both high predictive accuracy and smaller model size. SVM-ATSA outperforms the existing SOTA models based on RoBERTa$_{large}$ and DeBERTa and enjoys substantially fewer parameters. Our results show that an SVM-based solution can outperform SOTA language models on the ATSA task, when the SVM-based learning pipeline is properly designed.

6 Limitations

PLM-based methods do not need feature construction and feature selection. Instead, they only need to tokenize the input text and convert the tokens into token ids. Furthermore, PLM-based methods can automatically learn the feature representation for the task. Our method is a pipeline containing sub-problem construction, feature construction, and feature selection. Each step in the pipeline requires to be carefully designed to achieve good predictive accuracy.

7 Acknowledgements

This work is supported by the Guangzhou Industrial Information and Intelligent Key Laboratory Project (No. 2024A03J0628). This work is also funded by the Guangzhou Science and Technology Development Project (No. 2023A03J0143 and No. 2024A04J4458) and the NSFC Project (No. 62306256).

References

Chenhua Chen, Zhiyang Teng, Zhongqing Wang, and Yue Zhang. 2022. Discrete opinion tree induction for aspect-based sentiment analysis. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2051–2064.

Hans Graf, Eric Cosatto, Leon Bottou, Igor Dourdanovic, and Vladimir Vapnik. 2004. Parallel support vector machines: The cascade svm. *Advances in neural information processing systems*, 17.

S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and Karuturi Radha Krishna Murthy. 2001. Improvements to platt's smo algorithm for svm classifier design. *Neural Computation*, 13(3):637–649.

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.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27—35.

Souhila Sadeg, Leila Hamdad, Karima Benatchba, and Zineb Habbas. 2015. Bso-fs: bee swarm optimization for feature selection in classification. In *Advances in Computational Intelligence: 13th International Work-Conference on Artificial Neural Networks, IWANN 2015, Palma de Mallorca, Spain, June 10-12, 2015. Proceedings, Part I 13*, pages 387–399. Springer.

Souhila Sadeg, Leila Hamdad, Amine Riad Remache, Mehdi Nedjmeddine Karech, Karima Benatchba, and Zineb Habbas. 2019. Qbso-fs: A reinforcement learning based bee swarm optimization metaheuristic for feature selection. In *Advances in Computational Intelligence: 15th International Work-Conference on Artificial Neural Networks, IWANN 2019, Gran Canaria, Spain, June 12-14, 2019, Proceedings, Part II 15*, pages 785–796. Springer.

Davoud Ataee Tarzanagh, Yingcong Li, Christos Thrampoulidis, and Samet Oymak. 2023. Transformers as support vector machines. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*.

Yuanhe Tian, Guimin Chen, and Yan Song. 2021. Aspect-based sentiment analysis with type-aware graph convolutional networks and layer ensemble. In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 2910–2922.

Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, and Yi Chang. 2021. Eliminating sentiment bias for aspect-level sentiment classification with unsupervised opinion extraction. *arXiv preprint arXiv:2109.02403*.

Christopher John Cornish Hellaby Watkins. 1989. Learning from delayed rewards.

Zeyi Wen, Jiashuai Shi, Qinbin Li, Bingsheng He, and Jian Chen. 2018. Thundersvm: A fast svm library on gpus and cpus. *The Journal of Machine Learning Research*, 19(1):797–801.

Zeyi Wen, Zhishang Zhou, Hanfeng Liu, Bingsheng He, Xia Li, and Jian Chen. 2021. Enhancing svms with problem context aware pipeline. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 1821–1829, New York, NY, USA. Association for Computing Machinery.

Heng Yang and Ke Li. 2021. Improving implicit sentiment learning via local sentiment aggregation. *arXiv preprint arXiv:2110.08604*.

Heng Yang and Ke Li. 2024. Modeling aspect sentiment coherency via local sentiment aggregation. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 182–195.

Yang You, James Demmel, Kenneth Czechowski, Le Song, and Richard Vuduc. 2015. Ca-svm: Communication-avoiding support vector machines on distributed systems. In *2015 IEEE International Parallel and Distributed Processing Symposium*, pages 847–859. IEEE.

Mao Zhang, Yongxin Zhu, Zhen Liu, Zhimin Bao, Yunfei Wu, Xing Sun, and Linli Xu. 2023. Span-level aspect-based sentiment analysis via table filling. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9273–9284.

Yiming Zhang, Min Zhang, Sai Wu, and Junbo Zhao. 2022. Towards unifying the label space for aspect- and sentence-based sentiment analysis. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 20–30.