Sentiment Interpretable Logic Tensor Network for Aspect-Term Sentiment Analysis

Bowen Zhang[1], Xu Huang[2], Zhichao Huang[3], Hu Huang[4], Baoquan Zhang[2], Xianghua Fu[1]* and Liwen Jing[1,5] *

[1]College of Big Data and Internet, Shenzhen Technology University, Shenzhen, China

[2] School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China

[3] JD Intelligent Cities Research, Beijing, China

[4] School of Cyber Science and Technology, University of Science and Technology of China, Hefei, China

[5] Shenzhen innovation Institute (Xinstitute), Shenzhen, China

Abstract

Aspect-term sentiment analysis (ATSA) is an important task that aims to infer the sentiment towards the given aspect-terms. It is often required in the industry that ATSA should be performed with interpretability, computational efficiency and high accuracy. However, such an ATSA method has not yet been developed. This study aims to develop an ATSA method that fulfills all these requirements. To achieve the goal, we propose a novel Sentiment Interpretable Logic Tensor Network (SILTN). SILTN is interpretable because it is a neurosymbolic formalism and a computational model that supports learning and reasoning about data with a differentiable first-order logic language (FOL). To realize SILTN with high inferring accuracy, we propose a novel learning strategy called the two-stage syntax knowledge distillation (TSynKD). Using widely used datasets, we experimentally demonstrate that the proposed TSynKD is effective for improving the accuracy of SILTN, and the SILTN has both high interpretability and computational efficiency.

1 Introduction

1.1 Motivation

Aspect-term sentiment analysis (ATSA) is a fine-grained task in sentiment analysis, which aims is to recognize the sentiment polarity of the given aspect-term in a sentence (Zafra et al., 2019). Early research for ATSA was developed based on manually extracted features. For example, Poria et al. (2014) designed hand-crafted dependency rules to obtain aspect-related words, which are then fed into the machine learning methods to infer the sentiment polarity. With the development of deep learning, deep neural networks (DNNs) have dominated the study. Generally, the performance of DNNs is superior to traditional machine learning methods when the labeled training data is sufficient. More recently, there are two classes of methods that have received attention: (i) syntax-aware neural network (SaNN), which integrates syntax knowledge into attention-based neural network that increases the predictor's performance and interpretability (Zhang et al., 2019; Wang et al., 2020; Li et al., 2020; Nguyen and Shirai, 2015); (ii) large pre-trained language model (PLM) for ATSA (e.g., Bert (Devlin et al., 2019; Song et al., 2019; Zeng et al., 2019; Dai et al., 2021)), which learns knowledge from large-scale corpus and stably exceeds the other baseline by a significant margin.

Despite the effectiveness of prior work, ATSA in real-world remains a challenge for several reasons. First, DNNs usually perform as a "black box", because they cannot explicitly explain the process of the analysis; therefore, they cannot be applied in cases where explanations are required. Second, the performance of SaNNs relies on the intricate knowledge integration mechanism, which introduces more trainable parameters and brings extra computational costs. Furthermore, based on our empirical observation, the SaNN achieves merely limited performance improvements on most attention-based models (see methods 9-12 in Table 1). Third, for the pre-trained model, the enormous parameters lead to high storage and computational costs, making them a burden to be deployed in resource-constrained application scenarios such as real-time inference on mobile or edge devices. Besides, similar to DNNs, the pre-trained methods also lack interpretability.

1.2 Purpose

In response, this study aims to develop an ATSA method that can achieve interpretability, computational efficiency and high accuracy simultaneously.

Interpretability. To satisfy the interpretability requirement, we aim to develop an understandable ATSA method that can extract aspect-related words

* Corresponding authors: xianghuafu@sztu.edu.cn, ljing@connect.ust.hk

Proceedings of the 29th International Conference on Computational Linguistics, pages 6705–6714
October 12–17, 2022.

with explicit semantic knowledge, and build the sentiment inferring neural network that can be explained by first-order logic language (FOL). The FOL type of explanation should be agreeable for humans, and valuable in a situation where explanations are required.

Computational efficiency. To achieve high computational efficiency, we aim to develop an efficient ATSA method that utilizes fewer parameters for prediction while still achieving comparable results as conventional state-of-the-art models.

High accuracy. To achieve high accuracy, we aim to develop an ATSA method that achieves outstanding performance while maintaining interpretability and computational efficiency at the same time. Although it is very challenging, inspired by knowledge distillation (Hinton et al., 2015), we aim to develop a knowledge distillation strategy that transfers the knowledge of large and high-performance networks into an interpretable and computational efficient network.

1.3 Approach

To achieve the goal, we propose a novel **S**entiment **I**nterpretable **L**ogic **T**ensor **N**etwork (SILTN) for ATSA. Further, to realize this SILTN with high inferring accuracy, we propose a two-stage syntax knowledge distillation (TSynKD) strategy.

SILTN. SILTN is a neurosymbolic formalism and a computational neural network that supports learning and reasoning about data with a differentiable first-order logic language. Logic rules provide a flexible declarative language for communicating high-level cognition and expressing structured knowledge.

TSynKD. Although SILTN is well interpretable, its predictive performance is unsatisfactory because of the shallow network structure. Therefore, we propose the TSynKD to improve the inferring accuracy of SILTN, which is motivated by the observation that knowledge distillation can compress the large and high-performance networks (teacher) into a small student model while preserving the knowledge of the teacher model. TSynKD consists of two distillation stages with three networks: a large network (first teacher), a big network (second teacher) and a small network (student). The first distillation stage is the *output distillation* stage, which makes a large network output logits as a big network training objective. In this paper, we use the pre-trained Bert as the large network. As for the big network, we propose an aspect-specific dynamic graph convolutional network (AsDGCN) to model the dependency knowledge. The second stage is *feature distillation* stage, which allows a student to learn from a teacher's intermediate features. Here, SILTN is the student for distilling dependency knowledge from teacher AsDGCN.

1.4 Contribution

We summarize our contributions as follows: (1) To the best of our knowledge, this is the first work to integrate an interpretable logic tensor network in a principled framework for ATSA. SILTN is constructed followed by FOL, which provides a flexible declarative language for communicating high-level cognition and expressing structured knowledge. (2) We propose a two-stage syntax knowledge distill strategy (TSynKD) in ATSA, which significantly improves the performance of SILTN. (3) Extensive experiments have been conducted to evaluate the effectiveness of our model for ATSA.

2 Our Methodology

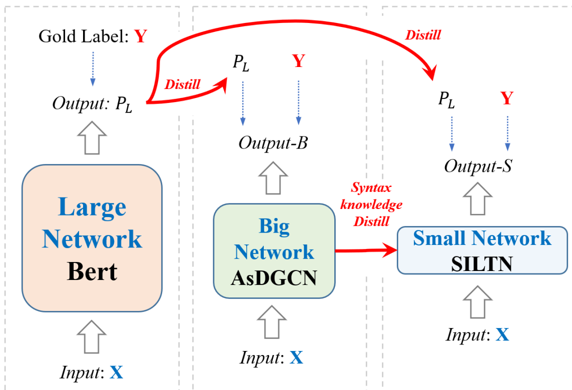


Figure 1: Overall framework.

2.1 Problem Definition

The ATSA task can be formulated as follows. Given a sentence $x = \{w_1^c, ..., w_a^a, ..., w_{a+m}^a, ..., w_n^c\}$ contains the corresponding aspect-term words $\{w_a^a, ..., w_{a+m}^a\}$, where w denotes each word in the sentence and m denotes the aspect-term length. Each sentence has a sentiment label y. ATSA aims to predict a sentiment label for the input sentence x towards the given aspect-term. In this paper, we use superscripts “c”, “a” to indicate a context word and aspect-term word, respectively.

2.2 Framework Overview

Figure 1 shows the overall framework of our proposed method. The blue block denotes the SILTN, which serves as a basic prediction model. SILTN is developed following FOL, where each *Grounding* $\mathcal{G}$ is constructed by a simple trainable neural network structure. To realize SILTN with high inferring accuracy, SILTN is trained by TSynKD strategy. TSynKD consists of two distillation stages with three networks: a large network (Bert), a big network (AsDGCN), and a small network (SILTN). The first stage is the *output distillation*, which utilizes Bert's output logits as the AsDGCN's training objective. The second stage is the feature distillation, where SILTN learns dependency knowledge from AsDGCN and utilizes Bert's output logits as the SILTN's training objective for further improving the inferring accuracy.

2.3 SILTN

2.3.1 Preliminary: Logic Tensor Network

Logic Tensor Network (LTN) is a neuro-symbolic formalism and computational model that supports learning and reasoning about data with rich knowledge. The semantics of logic in LTN (called *Real Logic*) depart from the standard abstract semantics of FOL. In *Real Logic*, every object denoted by constants, variables and terms is interpreted as a tensor of real values. *Predicates* are interpreted as functions or tensor operations projecting onto a value in the interval [0, 1]. Here, functions are usually implemented by neural networks. *Grounding* in *Real Logic*, denoted by $\mathcal{G}$, associates a tensor of real numbers, where a real number in the interval [0, 1].

2.3.2 SILTN Notation and Definition

The notation and definition of SILTN are as follows:

Domains: *texts*, denoting the examples from dataset. *labels*, denoting the class labels.

Variables: x_+, $x_\circ$ and x_- denoting the text of "positive", "neutral" and "negative", respectively. x for all examples. D(x_+) = D($x_\circ$) = D(x_-) = D(x) = texts.

Constants: l_+, $l_\circ$ and l_-, the labels of classes for "positive", "neutral" and "negative", respectively. D(l_+) = D($l_\circ$) = D(l_-) = labels.

Predicates: $A(x)$ denoting the dependency relation. $K(k,q)$ denoting the knowledge rule $k \rightarrow q$. $R(x,a)$ denoting the fact that the aspect-related words toward the aspect-term a. $P(R(x,a),l)$ denoting the fact that text x is classified as l when targeting to aspect-term a.

Axioms:

$$\forall x\ K(A(x), R(x,a)) \tag{1}$$

$$\forall x_+\ P(R(x_+,a), l_+) \tag{2}$$

$$\forall x_\circ\ P(R(x_\circ,a), l_\circ) \tag{3}$$

$$\forall x_-\ P(R(x_-,a), l_-) \tag{4}$$

Notice that rules about exclusiveness such as $\forall x(P(x,l_+) \rightarrow (\neg P(x,l_\circ) \wedge \neg P(x,l_-)))$ are not included since such constraints are already imposed by the grounding of P below, more specifically the softmax function.

Grounding: $\mathcal{G}(l)$ is the one-hot vector where $\mathcal{G}(l_+) = [1,0,0]$, $\mathcal{G}(l_\circ) = [0,1,0]$ and $\mathcal{G}(l_-) = [0,0,1]$. $\mathcal{G}(x)$ is a word matrix of x. $\mathcal{G}(a)$ and $\mathcal{G}(c)$ are word matrix of aspect-term a and content words c, respectively. $\mathcal{G}(A(x))$ is a dependency relation vector sequence of the given text x. $\mathcal{G}(R(x,a))$ is a vector sequence that computed by $\mathcal{G}(x)$ and $\mathcal{G}(a)$. $\mathcal{G}(K(A(x), R(x,a)))$ is a vector sequence that computed by $\mathcal{G}(A(x))$ and $\mathcal{G}(R(x,a))$. $\mathcal{G}(P|\theta)$: (x, a), $l \mapsto$ softmax(SILTN$_\theta$(x, a)), where the LTN has three output neurons corresponding to the sentiment polarity "positive, neural or negative", and each neurons gives the probability corresponding to the class l.

2.3.3 Network structure of SILTN

Follow the SILTN definitions, we aim to achieve $\mathcal{G}(P|\theta)$: (x, a), $l \mapsto$ softmax(SILTN$_\theta$(x, a)), where each grounding is constructed by neural network structure. Figure 2 is the framework of SILTN.

Specifically, following FOL, $\mathcal{G}(P)$ can be decomposed by each groundings. According to Figure 2, the first layer *Grounding* is $\mathcal{G}(x)$, where the *Grounding* of input sentence x is constructed by the text representation model such as LSTM or Bert, etc. Formally, given input sequence x, we convert the i-th word into a low-dimensional vector representation e_i by embedding layer, where d denotes the dimension of embedded vectors. The $\mathcal{G}(x)$ represents the hidden states of LSTM, where the input is the combination of e_i.

The second layer *Grounding* is *Predicates* $A(x)$, which aims to model the dependency relation of each word toward the aspect-term. To achieve this goal, the distilled dependency knowledge is utilized in this layer. Formally, $\mathcal{G}(A(x))$ can be calculated as:

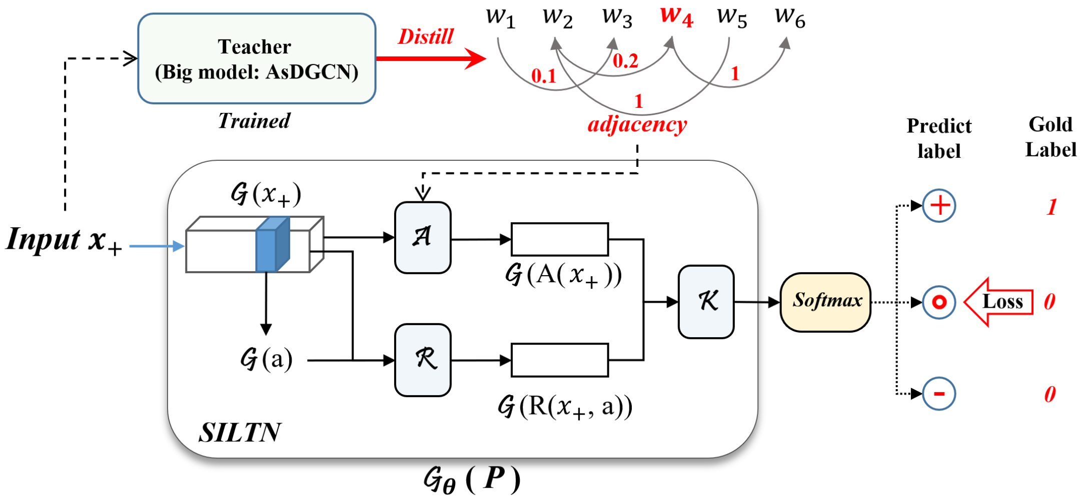


Figure 2: Framework of the proposed SILTN.

$$\mathcal{G}(A(x)) = \sigma(\sum_{i=0}^{m} \hat{A}_i \mathcal{G}(x)_i W_i + b), \tag{5}$$

where $\hat{A}$ is the distilled dependency knowledge (see Eq. (9)); σ denotes the active function. Note that W, b are distilled from AsDGCN (share parameters).

Next, $\mathcal{G}(R(x, a))$ is computed by :

$$c = \mathcal{G}(x)\mathcal{G}(a)^{\mathrm{T}},$$
$$\mathcal{G}(R(x, a)) = \mathcal{G}(x)_t softmax(\sum_t c_t), \tag{6}$$

where $softmax(f_i) = \frac{e^{f_i}}{\sum_j e^{f_j}}$, t denodes the t-th word. Specifically, $\mathcal{G}(a)$ is obtained by selecting the corresponding vector through the index $\{w_a^a, ..., w_{a+m}^a\}$ from $\mathcal{G}(x)$.

After acquiring the representation $\mathcal{G}((R(x, a))$, it is fed into the feed-forward layer to compute $\mathcal{G}(K(A(x), R(x, a)))$. Here, $K(A(x), R(x, a))$ represents the soft logic $A(x) \rightarrow R(x, a)$. Follow soft logic operation in Badreddine et al. (2022), $K(A(x), R(x, a))$ is computed by:

$$\mathcal{G}(K(A(x), R(x, a))) =$$
$$1 - \mathcal{G}(A(x)) + \mathcal{G}(A(x)) \cdot \mathcal{G}(R(x, a)) \tag{7}$$

Then the softmax layer to obtain the sentiment probability distribution:

$$P_S = softmax(W_1 \mathcal{G}(K(A(x), R(x, a))) + b_1), \tag{8}$$

where W_1 and b_1 are trainable parameters.

2.4 TSynKD

2.4.1 Key Idea of TSynKD

The "deepth" of the network structure is shallow, which poses a big challenge to improve the network's performance on ATSA while maintaining an interpretable and simple network structure. An effective solution is to adopt knowledge distillation, which can transfer the knowledge from the large network (teacher) into the small network (student), and improve the student's performance significantly.

2.4.2 Large Network (Bert)

In this paper, we deploy the pre-trained Bert model as the first teacher that produces features learned from large-scale corpus. Specifically, we first fine-tune the Bert model, and then generate the outputs of all training samples. The outputs are then denoted as a big model training objective in *output distillation*. Bert model takes "[CLS] sentence [SEP] aspect-terms [SEP]" as input, which computes the deep representations of sentences and aspects. We denote the output of Bert as P_L.

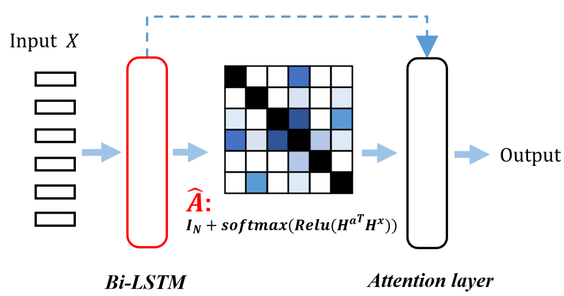


Figure 3: Framework of AsDGCN.

2.4.3 Big Network (AsDGCN)

AsDGCN is the extention of Zhang et al. (2019), which aims to distill dependency knowledge from Bert by *output distillation*, and then transfer the knowledge into the student SILTN. We only briefly discuss sections overlapping with contents in Zhang et al. (2019) so that we can put more emphasis on the new contributions.

Formally, the first layer of AsDGCN is Bi-LSTM network, which can be denoted as $H^x = \{h_1^c, ..., h_2^a, ..., h_n^c\}$, H^c and H^a are the combination of the hidden states of content words and aspect-terms, respectively. The second layer is constructed for learning the dependency relation (denoted as $\hat{A}$) of each word. In particular, we propose a novel attention mechanism, where the attention distributions simulate the adjacency relations. Specifically, we first use the hidden states of aspect-term (H^a) as the attention query to calculate the attention distribution with the hidden state H^x. Then, $\hat{A}$ can be computed as:

$$\hat{A} = I_N + softmax(Relu({H^a}^{\mathrm{T}} H^x)), \quad (9)$$

where I_N is the identity matrix. After obtaining the adjacency $\hat{A}$, The final graph representation S can be calculated by GCN:

$$S = \sigma(\hat{A}HW + b), \quad (10)$$

where σ represents a non-linear function, W+b are trainable parameters.

Finally, S is then fed into the attention-based layer and followed by the softmax layer to compute a sentiment probability distribution P_B.

2.4.4 Learning

- **Output distillation:**

  The output logits serve as a soft target providing richer supervision than the hard target of the one-hot gold label for the training (Hinton et al., 2015). Given an input sentence x with the gold label y (one-hot), the output logits of the large network (Bert) P_L and the output logits of the big network (AsDGCN) P_B, the loss function of output distillation denotes as:

$$\begin{aligned} L_{od} &= \alpha_1 \cdot CE(P_B, Y) \\ &+ (1 - \alpha_1) \cdot MSE(P_L, P_B) \end{aligned} \quad (11)$$

- **Feature distillation:**

  To capture rich syntactic tree features, we first consider allowing the student SILTN to directly utilize dependency relations $\hat{A}$ from AsDGCN.

  Second, SILTN shares the parameters of syntax layer with AsDGCN (See from Eq. (5) and Eq. (10)).

- **Loss function for SILTN:**

  The loss function of SILTN is similar to *output distillation*, which can be computed as:

$$\begin{aligned} \mathcal{L} &= \alpha_2 \cdot CE(P_S, Y) \\ &+ (1 - \alpha_2) \cdot MSE(P_L, P_S) \end{aligned} \quad (12)$$

Training strategy: From Figure 1, the training of the overall framework has three steps: (1) fine-tune Bert and then predict the sentiment label P_L; (2) train AsDGCN by output distillation (Eq. 11); (3) train SILTN by leveraging $\hat{A}$ (Eq. 9) from the trained AsDGCN and then optimize through the loss function $\mathcal{L}$ (Eq. 12). During inference, the well-trained SILTN can make predictions on its own for the given input.

3 Experiments

Datasets. To evaluate the effectiveness of our method, we conduct extensive experiments on five datasets. **Twitter** dataset is obtained from Dong et al. (2014). There are 1561 positive, 3127 neutral and 1560 negative tweets for training and 692 for the test. **Lap14** and **Rest14** datasets are taken from SemEval-14 Task 4 in Pontiki et al. (2014). Lap14 denotes the laptop reviews and it contains 2328 training texts and 638 test samples. Rest14 composes of the restaurant reviews, it contains 2164 positive, 637 neutral and 807 negative texts for training and 1120 samples for test. **Rest15** is collected from SemEval 2015 task 12 in Pontiki et al. (2015), it contains in total 1204 training samples with three sentiment classes and 542 samples for test. **SPD** is collected from Zhang et al. (2020), the sentences in SPD all contain unique structures, such as conditional statements and subjunctive. It contains 4726 training samples and 1182 test samples.

Baselines We adopt several sentiment classification methods as baselines. **SVM** (Kiritchenko et al., 2014) is an effective traditional mechine learning based method. **LSTM** (Tang et al., 2016a) utilizes the standard LSTM to model the sentiment representation. **IAN** (Ma et al., 2017), **MemNet** (Tang

Model	Twitter		Lap14		Rest14		Rest15		SPD	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
1.SVM¶	63.40	63.30	70.49	-	80.16	-	-	-	-	-
2.LSTM†	69.56	66.42	69.28	63.21	78.13	67.42	77.37	52.97	61.16	59.17
3.Memnet¶	71.48	68.14	70.64	65.19	79.61	68.14	77.31	56.17	61.42	57.56
4.AOA†	72.30	68.20	72.62	66.97	79.97	69.59	78.17	57.21	63.76	64.41
5.IAN†	72.50	68.14	72.05	67.38	79.26	70.12	78.54	52.21	63.76	63.96
6.CapsNet¶	-	-	-	-	69.63	69.63	78.14	61.57	-	-
7.TNet-LF†	72.98	71.43	74.61	70.14	80.40	70.57	78.47	59.12	63.76	64.96
8.MIMLLN¶	-	- -	-	-	81.06	71.25	78.27	60.59	-	-
Syntax-aware methods										
9.PRNN¶	-	-	-	-	66.20	59.32	-	-	-	-
10.SAttn¶	-	-	72.57	69.13	80.45	71.26	-	-	-	-
11.ASGCN†	72.15	71.00	71.05	70.72	80.86	72.73	79.89	59.47	66.24	65.24
12.R-GAT†	71.56	71.07	72.49	71.01	73.83	72.14	78.92	61.24	66.87	65.14
Ours (SILTN)										
-sp	70.95	68.73	72.57	68.14	81.16	71.87	79.97	57.76	63.74	60.12
-(dep)	**73.12**	72.25	**76.96**	72.95‡	**83.13**	75.12‡	81.01	64.11	67.60	67.38‡
-(dis-dep)	73.01	**73.07**‡	76.77	**73.03**‡	83.02	**75.86**‡	**81.37**	**64.26**‡	**67.94**	**68.01**‡

Table 1: Evaluation results (%) on none-pretrained based methods. The best result on each task is in bold. The mark ¶ refers to the results reported in the original papers, while † mark refers to the open implementation, ‡ mark refers to p-value < 0.05 when comparing with the best competitor.

	Twitter	Rest14	Lap14
Bert	74.41	76.21	75.10
Bert-PT	-	76.48	75.08
LCF-BERT	73.34	75.03	76.26
AEN-Bert	73.13	73.76	76.31
RoBerta-ASC	-	75.12	70.52
Bert-ASGCN	74.67	76.29	75.96
Bert-RGAT	74.88	74.88	74.07
SILTN-Bert	**75.52**	**77.04**	**76.34**

Table 2: Evaluation results (F1 %) compared with pre-trained models.

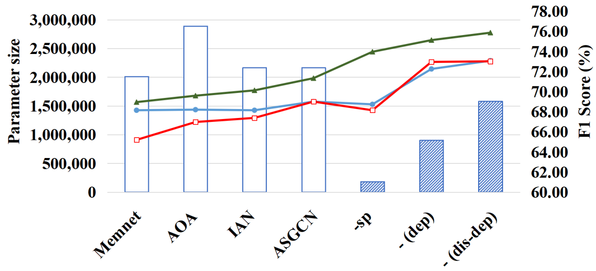
et al., 2016b), **AOA** (Huang et al., 2018)and **TNet-LF** (Li et al., 2018) are attention-based methods. **ASGCN** (Zhang et al., 2019) and **R-GAT** (Wang et al., 2020) use GCN to model the dependency tree graph for ATSA. **MIMLLN** (Li et al., 2020) treats the aspect category as the key instances. **PRNN** (Nguyen and Shirai, 2015) takes both dependency and constituent trees into LSTM. **SAttn** further integrates attention mechanism with PhraseRNN. **Bert** (Devlin et al., 2019) is a pre-trained BERT model to perform ATSA. We convert the given context and target to "[CLS] + aspect-term + [SEP] + context" structure. Further we select several variants Bert-based model **Bert-PT** (Xu et al., 2019),**AEN-Bert** (Song et al., 2019), **LCF-Bert** (Zeng et al., 2019), **RoBerta-ASC**, **Bert-ASGCN** and **Bert-RGAT**(Dai et al., 2021).

Variant of SILTN. To get a trade-off between interpretability, computational efficiency and high accuracy, we increase or simplify trainable parameters for SILTN, and compare their prediction accuracy and prediction time. **SILTN-sp**: This method maximally simplifies the training parameters. We expect that each grounding of SILTN is a simple trainable tensor production, and for the first layer of grounding $\mathcal{G}(x)$, we directly initialize it with glove vector without adding any text representation structure. **SILTN-LSTM**: To sacrifice the interpretability and improve inferring performance, we use one LSTM layer to construct $\mathcal{G}(x)$. Note that, the **SILTN-(dep)** denotes the dependency knowledge acquired from the external tools, and the dependency knowledge of **SILTN-(dis-dep)** is distilled from Big network AsDGCN. **SILTN-Bert**: To further improve the inferring performance, we use pretrained Bert to model $\mathcal{G}(x)$.

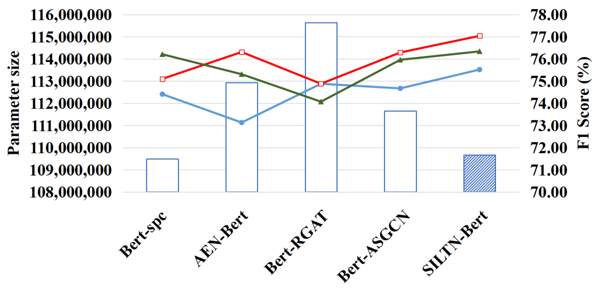
Experimental Setting. In this paper, we utilize 300-dimensional pre-trained GloVe vectors to initialize the word embeddings. The dimensions of the hidden state of Bi-LSTM is 128. The scale weight $\alpha_{\{1,2\}}$ is 0.01. The model is optimized with the Adam optimization algorithm with the batch size of 32 and the learning rate is 0.001. As in Zhang et al. (2019), We use accuracy and macro-averaged F1 score as the evaluation metrics, which are widely adopted in sentiment classification. We compute the metrics independently for each class and then take the average (hence treating all classes equally), as the final performance.

3.1 Task Setup and Quantitative Results

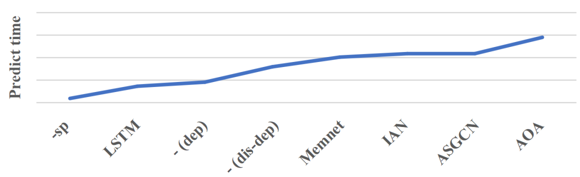
To evaluate the stability of the model, following Zhang et al. (2019), we run the method three times and summarize the best results in Tables 1. Besides, we adopt the *t*-test to confirm the significance of differences between the other methods with a p-value of 0.05.



(a) None-Bert model.



(b) Bert-based model.



(c) Predict time (sort by time increase).

Figure 4: Comparison results of the size of training parameters vs. F1 score.

According to the results, we can observe that our model consistently outperforms the compared baseline methods. **First**, SILTN outperforms all none-syntax baselines (No. 1-8) on all datasets, which indicates that SILTN has better ability to infer the sentiment polarities by utilizing dependency knowledge. For example, compared with the best competitor of methods 1-8, SILTN(dis-dep) improves 1.64% on Twitter, 4.61% on Rest14 and 2.69% on Rest15 for F1 score, respectively. This is because SILTN utilizes the dependency knowledge distilled from the big model, which enriches the learning ability between the word and aspect-terms. **Second**, compared with the syntax-aware models (No. 9-12), SILTN improves 2.02% on Lap14, 3.13% on Rest14 and 3.02% on Rest15 for F1 score, respectively. The reason is that the syntax dependency tree from the SpaCy tool utilized by the conventional method may introduce additional errors especially for the text are short and informal. **Third**, to further imporve the infering accuracy, we compare our method with Bert-based methods and give the results in Table 2. The result shows SILTN-Bert achieves the best F1 score over the competitors. Note that, SILTN-(dis-dep) reduces parameters by 99.17% compared to the Bert model; however, SILTN still achieves compatible results.

In sum, the advantage of SILTN comes from its two characteristics: (i) TSynKD provides adequate syntax knowledge, making the SILTN able to combine the prior knowledge effectively. (ii) SILTN uses fewer parameters while interpretable, which leads to high efficiency and high accuracy.

3.1.1 Cost Efficiency vs. Accuracy

Figure 4 summarized the number of parameters and the F1 score results. First, compare SILTN with the baselines, SILTN contains fewer parameters but achieves state-of-the-art results. For example, *-dep* has 58.32% less parameters than ASGCN, but the performance improved by 4.46% on Rest15 and 2.23% on Lap14. Moreover, the predict time of *-dep* is comparable to simple LSTM model. Second, compared with SILTN-sp, which has the fewest parameters (even fewer 74.91% than standard LSTM), our method can still obtain competitive results to the conventional best attention mechanism model (ASGCN). In sum, the results show the effectiveness of the TSynKD framework, and it proves that our model can strike a balance between computational efficiency and high accuracy.

3.2 Ablation Study

To study the impact of each component of the proposed method, we implement the ablation test to remove the proposed component denoted as -w/o.

Specifically, **-w/o TSynKD**: SILTN without the TSynKD framework, and utilize the standard dependency tree[1] as the syntax knowledge. **-w/o TSynKD$_{OT}$**: SILTN trained by utilizing only *output distillation strategy* and utilize the standard dependency tree as the syntax knowledge. **-w/o TSynKD$_{FT}$**: SILTN trained by utilizing only *feature distillation strategy* and utilize the standard dependency tree as the syntax knowledge.

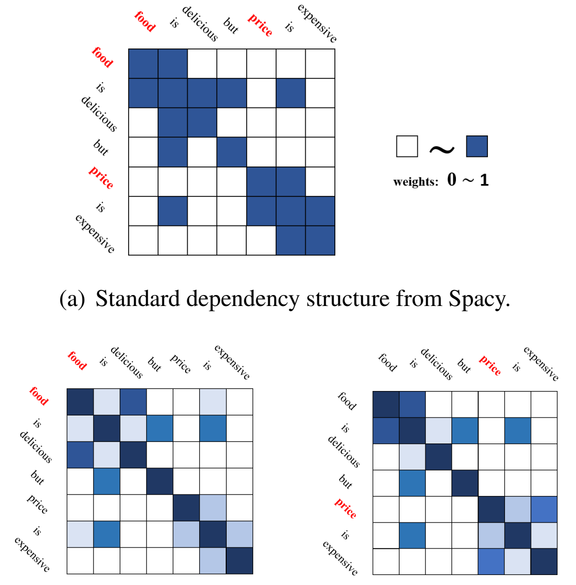
Methods	Twitter	Lap14	Rest14
SILTN-sp	68.73	68.14	71.87
-w/o TSynKD	65.84	65.44	68.53
-w/o TSynKD$_{OT}$	66.73	66.22	70.20
-w/o TSynKD$_{FT}$	67.42	66.74	69.66

Table 3: Ablation study results (F1%).

The ablation results are summarized in Table 3. According to the results, we can summarize that all the proposed components contribute a great improvement to SILTN. In particular, the accuracy score decreases sharply when discarding the TSynKD framework. This is within our expectation since the TSynKD injects knowledge from the large network and the dependency knowledge from the big model into SILTN. For example, the F1 score drops 2.89%, 2.70% and 3.34% for Twitter, Lap14 and Rest14, respectively. In addition, the proposed two-stage distillation framework also contributes to the effectiveness of SILTN. For example, the F1 score decreases 1.92% for Twitter when discarding *feature distillation*, and decreases 2.21% for Rest14 when discarding *output distillation*. Not surprisingly, combining all factors achieves the best performance for all the experiments.

3.3 Visualized of Distilled Dependency Relation

The dependency knowledge enables SILTN to utilize tree structures for capturing the corrected aspect-related words. To understand how the distilled model promotes the mutual learning of dependency structures, we empirically visualize the adjacency relation based on a test example: "food is delicious but price is expensive". This sample contains "food" and "price" two aspect-terms. The visualization is summarized in Figure 5. Figure 5. (a) is the standard dependency structure from SpaCy tools, which the conventional methods utilize. Both "food" and "price" share the same adjacency matrix. Figure 5. (b) and (c) are acquired from Eq. 9 for our AsDGCN. They generate adjacency matrices for each aspect-term, respectively. For example, in Figure 5. (b) "food" increases the dependency weight associated with "delicious"; and in Figure 5. (c), the "price" directly connects with "expensive".



(a) Standard dependency structure from Spacy.

(b) From AsDGCN with aspect-term "food".

(c) From AsDGCN with aspect-term "price".

Figure 5: Visualized of distilled dependency relation.

4 Conclusion

In this paper, we propose a novel Sentiment Interpretable Logic Tensor Network (SILTN). SILTN is interpretable because it is constructed followed by first-order logic language (FOL). The sentiment inferring process can be decomposed into different *grounding*, which is constructed by the simple neural network; therefore, SILTN is computationally efficient. To achieve high inferring accuracy, we propose a two-stage syntax knowledge distillation (TSynKD) strategy. TSynKD consists of two distillation stages with three networks: Bert, AsDGCN and SILTN. The first distill stage refers to the output distillation, which makes the Bert output logits as the training objective of AsDGCN. In AsDGCN, we develop a novel attention structure to learn the aspect-specific dependency knowledge through output distillation. The second stage is *feature distillation*, which allows a SILTN to learn from AsDGCN's intermediate feature representations. Extensive experiments have been conducted on 5 real-world datasets. The experimental results show that the proposed SILTN with

[1]Spacy tools: https://spacy.io/

the TSynKD strategy significantly outperforms the conventional attention-based methods, and achieve compatible results compared with the state-of-the-art Bert-based methods for aspect-term sentiment analysis.

5 Acknowledgements

This research is supported by the Stable Support Project for Shenzhen Higher Education Institutions (SZWD2021011).

References

Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. 2022. Logic tensor networks. *Artif. Intell.*, 303:103649.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? A strong baseline for aspect-based sentiment analysis with roberta. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 1816–1829. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 49–54.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Binxuan Huang, Yanglan Ou, and Kathleen M Carley. 2018. Aspect level sentiment classification with attention-over-attention neural networks. In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, pages 197–206. Springer.

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.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018. Transformation networks for target-oriented sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 946–956.

Yuncong Li, Cunxiang Yin, Sheng-hua Zhong, and Xu Pan. 2020. Multi-instance multi-label learning networks for aspect-category sentiment analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 3550–3560. Association for Computational Linguistics.

Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4068–4074. AAAI Press.

Thien Hai Nguyen and Kiyoaki Shirai. 2015. Phrasernn: Phrase recursive neural network for aspect-based sentiment analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2509–2514.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Haris 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, Dublin, Ireland. Association for Computational Linguistics.

Soujanya Poria, Nir Ofek, Alexander F. Gelbukh, Amir Hussain, and Lior Rokach. 2014. Dependency tree-based rules for concept-level aspect-based sentiment analysis. volume 475, pages 41–47.

Youwei Song, Jiahai Wang, Tao Jiang, Zhiyue Liu, and Yanghui Rao. 2019. Attentional encoder network for targeted sentiment classification. *arXiv preprint arXiv:1902.09314*.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016a. Effective lstms for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307.

Duyu Tang, Bing Qin, and Ting Liu. 2016b. Aspect level sentiment classification with deep memory network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 214–224.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238. Association for Computational Linguistics.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2019. Bert post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335.

Salud M. Jiménez Zafra, María Teresa Martín Valdivia, Eugenio Martínez-Cámara, and Luis Alfonso Ureña López. 2019. Studying the scope of negation for spanish sentiment analysis on twitter. *IEEE Trans. Affect. Comput.*, 10(1):129–141.

Biqing Zeng, Heng Yang, Ruyang Xu, Wu Zhou, and Xuli Han. 2019. Lcf: A local context focus mechanism for aspect-based sentiment classification. *Applied Sciences*, 9(16):3389.

Bowen Zhang, Xutao Li, Xiaofei Xu, Ka-Cheong Leung, Zhiyao Chen, and Yunming Ye. 2020. Knowledge guided capsule attention network for aspect-based sentiment analysis. *IEEE ACM Trans. Audio Speech Lang. Process.*, 28:2538–2551.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4560–4570.