Joint Inference for Aspect-level Sentiment Analysis by Deep Neural Networks and Linguistic Hints

Yanyan Wang, Qun Chen, Murtadha Ahmed, Zhanhuai Li, Wei Pan, Hailong Liu

Abstract—The state-of-the-art techniques for aspect-level sentiment analysis focused on feature modeling using a variety of deep neural networks (DNN). Unfortunately, their performance may still fall short of expectation in real scenarios due to the semantic complexity of natural languages. Motivated by the observation that many linguistic hints (e.g. sentiment words and shift words) are reliable polarity indicators, we propose a joint framework, SenHint, which can seamlessly integrate the output of deep neural networks and the implications of linguistic hints in a unified model based on Markov logic network (MLN). SenHint leverages the linguistic hints for multiple purposes: (1) to identify the easy instances, whose polarities can be automatically determined by the machine with high accuracy; (2) to capture the influence of sentiment words on aspect polarities; (2) to capture the implicit relations between aspect polarities. We present the required techniques for extracting linguistic hints, encoding their implications as well as the output of DNN into the unified model, and joint inference. Finally, we have empirically evaluated the performance of SenHint on both English and Chinese benchmark datasets. Our extensive experiments have shown that compared to the state-of-the-art DNN techniques, SenHint can effectively improve polarity detection accuracy by considerable margins.

Index Terms—Deep neural networks, Linguistic hints, Aspect-level sentiment analysis

1 Introduction

Aspect-level sentiment analysis (ALSA) [1], a fine-grained classification task, has recently become an active research area in NLP. Its goal is to extract the opinions expressed towards different aspects of a product. ALSA can provide important insights into products to both consumers and businesses [2]. In the literature [3], two finer subtasks of ALSA have been studied: aspect-category sentiment analysis (ACSA) and aspect-term sentiment analysis (ATSA). ACSA aims to predict the sentiment polarity towards a few predefined aspect categories, which may not explicitly appear in the text. ATSA instead deals with explicit aspects involving a single word or a multi-word phrase. In this paper, we target both ACSA and ATSA. Consider the running example shown in Table 1, in which R_i and S_{ij} denote the review and sentence identifiers respectively. It can be observed that in R_2, the aspect term "battery" explicitly appears in the sentence S_{21}, while the sentence S_{22} does not explicitly contain its target aspect term ("laptop#performance"). ACSA has to detect the polarities of the aspects in both S_{21} and S_{22}. In contrast, ATSA only needs to detect the aspect polarity in S_{21}.

The state-of-the-art solutions for aspect-level sentiment analysis [4], [5] are mainly built on a variety of deep neural networks (DNN), which can automatically learn multiple levels of feature representation. Even though the DNN techniques can achieve empirically better performance than the previous alternatives (e.g. the techniques based on lexicon [6], [7] and SVM [8], [9]), their practical performance may still fall short of expectation due to the semantic complexity of natural languages. For instance, on most ACSA tasks of the popular SemEval benchmark, the reported top accuracy levels are only around 80% [1], [10].

TABLE 1: A Running Example from Laptop Reviews

R_i	S_{ij}	**Text**
R_1	S_{11}	I **like** the battery that can last **long** time.
	S_{12}	However, the keyboard sits a little **far** back for me.
R_2	S_{21}	The laptop has a **long** battery life.
	S_{22}	It also can run my games **smoothly**.

It can be observed that natural languages provide rich linguistic hints potentially useful for polarity reasoning. A sentence may contain strong sentiment words that explicitly express sentiment. In the running example, the presence of the strong sentiment word "like", together with the absence of any negative word, suggests that the sentiment of the sentence S_{11} is positive. A sentence may also contain shift words (e.g. *but* and *however*), which do not directly indicate polarity but explicitly specify the relationship between two neighboring aspect polarities. Again in the running example, the word "However" at the beginning of the sentence S_{12} indicates that its polarity is opposite to the polarity of the sentence S_{11}. In contrast, the absence of any shift word between two neighboring sentences usually means that their polarities are similar (e.g. S_{21} and S_{22}).

Unfortunately, the existing DNN techniques have limited capability in modeling varied linguistic hints. In this paper, we propose a novel framework, SenHint, which enables joint inference based on both DNN and linguistic hints. It

• *1. School of Computer Science, Northwestern Polytechnical University.*
2. Key Laboratory of Big Data Storage and Management, Northwestern Polytechnical University, Ministry of Industry and Information Technology
127 West Youyi Road, Xi'an Shaanxi, China
E-mail: {wangyanyan@mail., chenbenben@, murtadha@mail., lizhh@, panwei1002@, liuhailong@}nwpu.edu.cn

first extracts explicit linguistic hints and then encodes their implications as well as the output of DNN in a unified model based on Markov logic network (MLN) [11]. We note that it is not new to leverage linguistic hints for sentiment analysis. The traditional lexicon-based approaches [12] used the hints of sentiment words to directly predict polarity by summing up all the sentiment scores; the hints of context-sensitive sentiment words have been integrated into deep neural networks for sentiment analysis [13]; the hints of shift words have also been used to tune the performance of deep neural networks for sentence-level sentiment analysis [14]. However, *SenHint* is novel in that it models both the output of deep neural networks and the implications of linguistic hints as first-class citizens in a unified MLN. Compared with previous work, SenHint also leverages linguistic hints for new purposes. For instance, it uses the hints of shift words to capture the implicit relations between aspect polarities for MLN reasoning.

The major contributions of this paper can be summarized as follows:

1) We propose SenHint, a joint inference framework for aspect-level sentiment analysis based on MLN. SenHint can seamlessly integrate the output of DNN and the implications of linguistic hints in a unified model;
2) We present the required techniques for linguistic hint extraction, MLN model construction, and joint MLN inference;
3) We empirically evaluate the performance of SenHint on both English and Chinese benchmark datasets. Our extensive experiments show that compared to the state-of-the-art DNN techniques, SenHint can effectively improve polarity detection accuracy by considerable margins .

Note that a prototype of SenHint has been demonstrated in [15]. We summarize the new contributions of this technical paper as follows:

1) It proposes an improved MLN model. Besides the implicit polarity relations indicated by the presence/absence of shift words, the new MLN model also encodes the influence of sentiment words on aspect polarities;
2) It presents the improved techniques for linguistic hint extraction, MLN model construction, and joint inference. Unlike the demo paper, it provides with the technical details of each proposed technique;
3) In empirical evaluation, while the demo paper only applied SenHint to ACSA tasks, it extends SenHint to handle both ACSA and ATSA tasks. Besides the DNN models used in the demo paper, it also compares SenHint to the more recently proposed DNN techniques for both ACSA and ATSA. It also separately evaluates the effect of various linguistic hints on the performance of SenHint. Finally, it empirically compares the new SenHint with the original version proposed in the demo paper. The experiments have shown that the new SenHint performs evidently better.

The rest of this paper is organized as follows: Section 2 reviews more related work. Section 3 defines the task and introduces Markov logic network, the reasoning model underlying SenHint. Section 4 gives the overview of the proposed framework. Section 5 presents the techniques of extracting linguistic hints. Section 6 describes how to encode the implications of linguistic hints as well as the output of DNN in a MLN. Section 7 presents the technique of joint inference. Section 8 presents the empirical evaluation results. Finally, we conclude this paper with some thoughts on future work in Section 9.

2 Related work

In general, sentiment analysis involves various tasks, such as polarity classification, subjectivity or objectivity identification, and multimodal fusion [16]. In this paper, we focus on the essential task of polarity classification. Sentiment analysis at different granularity levels, including document, sentence, and aspect levels, has been extensively studied in the literature [17]. In this section, we first review the work on document and sentence level sentiment analysis, then the work on aspect-level sentiment analysis, and finally other relevant work on sentiment analysis.

2.1 Document and Sentence Level Sentiment Analysis

At the document (resp. sentence) level, its goal is to detect the polarity of the entire document (resp. sentence) without regard to the mentioned aspects. The state-of-the-art approaches were built on deep neural networks (e.g. CNN and RNN), which include Character-level Convolutional Networks [18], Deep Pyramid Convolutional Neural Networks [19] and Linguistically Regularized LSTM [20]. Some work proposed to combine an attention mechanism with neural networks, for instance Hierarchical Attention Network [21], Hierarchical Query-driven Attention Network [22], Linguistic-aware Attention Network [23] and Cognition Based Attention Model [24]. Moreover, Self-Attention Network [25] (inspired by the Transformer architecture), a flexible and interpretable architecture, has been proposed for text classification. Unfortunately, all these proposals can not be directly applied to aspect-level sentiment analysis because a sentence may hold different opinions on different aspects.

2.2 Aspect-level Sentiment Analysis

Aspect-level sentiment analysis needs to first extract the target aspects from a given sentence, and then determine their sentiment polarities. The dominant models for aspect extraction, which include Attention Based Aspect Extraction [26] and Aspect Extraction with Sememe Attentions [27], employed unsupervised framework analogous to an autoencoder to learn the aspects with varied attention mechanisms. There also exist some work aiming to jointly detect the aspects and identify their sentiment polarity [28], [29].

In this paper, we instead focus on how to determine the polarities of the given aspects in a sentence. Since deep neural networks can automatically learn high-quality features or representations, the state-of-the-art approaches attempted to adapt such models for aspect-level sentiment analysis. The existing work can be divided into two categories based on the two finer subtasks of ATSA and ACSA.

For ATSA task, Dong [30] initially proposed an Adaptive Recursive Neural Network (AdaRNN) that can employ a novel multi-compositionality layer to propagate the sentiments of words towards the target. Noticing that the models based on recursive neural network heavily rely on external syntactic parser, which may result in inferior performance, the following work [31] focused on recurrent neural networks. The alternative solutions include memory networks [32] and convolutional neural networks [33]. Due to the great success of attention mechanism in machine translation [34] and question answering [35], many models based on LSTM and attention mechanism have also been proposed. These models, including Hierarchical Attention Network [36], Segmentation Attention Network [37], Interactive Attention Networks [38], Recurrent Attention Network [39], Attention-over-Attention Neural Networks [40], Effective Attention Modeling [41], Content Attention Model [42], Multi-grained Attention Network [43], employed different attention mechanisms to output the aspect-specific sentiment features. More recently, the capsule networks [44], a type of artificial neural network that can better model hierarchical relationships, have also been leveraged for ATSA task. Chen [45] proposed a Transfer Capsule Network for transferring document-level knowledge to aspect-level sentiment analysis.

In comparison, there exist fewer works for ACSA because the implicit aspects make the task more challenging. Ruder [46] proposed a hierarchical bidirectional LSTM for the ACSA task by modeling the inter-dependencies of sentences in a review, which does not fully employ the given aspect. Wang [47] presented an attention-based LSTM that employs an aspect-to-sentence attention mechanism to concentrate on the key part of a sentence given an aspect. Xue [3] introduced a model based on convolutional neural networks and gating mechanisms, which is more accurate and efficient. Wang [48] presented an AS-Capsule model that can fully employ the correlation between aspect and sentiment through shared components. Note that the models proposed for ACSA can also be used for ATSA, but the ones for ATSA usually solely benefit themselves because they usually employ specific components to model explicit aspect-term together with its relative context.

2.3 Other Relevant Work

There also exist some work that studied how to integrate linguistic hints into DNN models. Ruder [46] presented a hierarchical LSTM (H-LSTM) to model the inter-dependencies of sentences in a review. Teng [13] employed a recurrent neural network to learn the sentiments strength, intensification and negation of lexicon sentiments that can capture the context of sentiment words. Qian [20] introduced linguistically regularized LSTM (LR-LSTM) to model the effect of sentiment, negation and intensity words. However, it modeled sentences independently and could not capture contextual relationship between sentences. Hu [14] employed the hints of shift words to capture the contrastive sense and then used this information to regulate the learning process of DNN. It is worthy to point out that although the aforementioned DNN models provide mechanisms for leveraging linguistic hints, they have very limited capability in the modeling of complex linguistic semantics (e.g., the relations between aspect polarities). Therefore, instead of tuning DNN models by linguistic hints, this paper proposes a novel approach for leveraging linguistic hints. It treats both DNN output and linguistic hints as first-class citizens and integrates their influence in a coherent model based on MLN.

Word representation, which is used as input by all the DNN models, plays an important role in sentiment analysis. Traditional word representations [49], [50] are effective at capturing semantic and syntactic information, but they usually perform poorly in capturing sentiment polarity. Therefore, there exist some recent work on sentiment-specific work representation. For instance, for twitter sentiment classification, Tang [51], [52] proposed C&W based models [53] to learn sentiment-specific word embedding by distant supervision. Fu [54] employed local context information as well as global sentiment representation to learn sentiment-specific word embeddings.

Markov logic network, as an expressive template language, enables joint inference based on both feature and relational information. It has been widely applied to many applications, such as entity resolution [55], [56], information extraction [57], [58] and sentiment analysis [59], [60]. However, the existing approaches based on MLN generally require human-designed features. In this paper, we integrate the DNN output and linguistic hints into a unified model based on MLN, which can retain the relational reasoning ability of MLN while avoiding complicated feature engineering.

3 Preliminaries

In this section, we first define the task and then introduce Markov logic network (MLN), the inference model underlying SenHint.

3.1 Task statement

For presentation simplicity, we have summarized the frequently used notations in Table 2. We formulate the task of aspect-level sentiment analysis as follows:

Definition 1. ***[Aspect-level Sentiment Analysis]*** Let $t_i = (r_j, s_k, a_l)$ be an aspect unit, where r_j is a review, s_k is a sentence in the review r_j, and a_l is an aspect associated with the sentence s_k. Note that the aspect a_l can be a aspect category or aspect term, and a sentence may express opinions towards multiple aspects. Given a corpus of reviews, R, the goal of the task is to predict the sentiment polarity of each aspect unit t_i in R.

3.2 Markov logic network

Markov logic network combines first-order logic and probabilistic graphical model in a single representation. In first-order logic, a set of formulas represent the hard constraints over a set of instances, and the rules can not be violated. The basic idea of MLN is to generalize first-order logic by softening the hard constraints, assigning a real number (a weight) to each formula to indicate how strong a constraint

TABLE 2: Frequently used notations.

Notation	Description
$t_i = (r_j, s_k, a_l)$	an aspect unit
r_j	a review
s_k	a sentence
a_l	an aspect category or aspect term
$T = \{t_i\}$	a set of aspect units
$v(t_i)$	a boolean variable indicating whether the sentiment polarity of t_i is positive
$V = \{v(t_i)\}$	a set of aspect polarity variables

is. In MLN, the instances can violate the formulas but need to pay a penalty: the higher the weight, the greater the penalty to be paid. MLN has been widely used to infer uncertain knowledge (e.g. Deepdive [61] and ProbKB [62]). Formally, a MLN is defined as follows [11]:

Definition 2. [Markov Logic Network] A MLN consists of a collection of weighted first-order logic formulas $\{(F_i, w_i)\}$, where F_i is a formula in first-order logic and w_i is a real number indicating the level of confidence on this formula.

Table 3 shows an example of MLN. In this example, the first formula states that if a person smokes, there is an implication that he/she would have cancer; the second one states that if two persons are friends and one of them smokes, the other one would also smoke. However, these formulas are not absolutely true. Therefore, the weights 2.0 and 3.0 are used to indicate how strong the constraints are. Note that in MLN, a weight can take the extreme value of ∞, which dictates that its corresponding rule can not be violated.

Grounding. A MLN provides a template for constructing factor graph. A factor graph consists of variable vertices $X = \{x_1, \cdots, x_n\}$ and factor vertices $\Phi = \{\phi_1, \cdots, \phi_n\}$, where each factor ϕ_i is a function $\phi_i(X_i)$ over the variables X_i ($X_i \subset X$). The factors together define a joint probability distribution over all the variables X. Table 4 shows an example of factor graph. In the figure, the variables and factors are represented by round and box nodes respectively.

Provided with a MLN and a set of constants, the process of constructing factor graph is called *grounding* [62]. In the grounding process, for each predicate and formula in MLN, we will create a set of *ground atoms* and *ground formulas* respectively, which are represented by the variables and factors in the factor graph. For instance, in Table 4, the ground atoms $smoke(Anna)$ and $cancer(Anna)$ are represented by the variables x_1 and x_2, and the ground formula "$smoke(Anna) \rightarrow cancer(Anna)$" is represented by the factor f_1 associated with x_1 and x_2. In the factor graph, each variable takes a value of 0 or 1 indicating its truth assignment, and each factor has a value of e^w if the ground formula is true, or 1 otherwise, where w denotes the weight of the formula.

Marginal Inference. A factor graph defines a probability distribution over its variables X:

$$P(X = x) = \frac{1}{Z}\prod_i \phi_i(X_i) = \frac{1}{Z}exp(\sum_i w_i n_i(x)) \quad (1)$$

where n_i denotes the number of true groundings of the formula F_i in x, w_i denotes the weight of F_i, and Z is the partition function, i.e. normalization constant. The process of computing the probability of each variable is referred to as *marginal inference*. Unfortunately, exact inference in MLN is intractable. Therefore, the sampling algorithms based on MCMC (e.g. Gibbs sampling [63]) are usually employed for approximate inference.

4 Framework Overview

As shown in Figure 1, the framework of SenHint consists of the following three modules:

- **Linguistic Hint Extraction:** This module retrieves relevant linguistic hints from reviews. It identifies easy instances of aspect polarity, extracts common sentiment features shared by aspect polarities and mines their polarity relations.
- **Knowledge Encoding:** This module employs weighted first-order logic rules to encode the implications of linguistic hints as well as the outputs of DNN into a MLN. The outputs of DNN capture the implicit influence resulting from multiple levels of automatically learned features, while the implications of linguistic hints enable explicit polarity inference.
- **Joint Inference:** This module constructs a ground factor graph based on the generated weighted first-order logic rules, and then performs joint inference on the factor graph.

The example factor graph constructed for the running example has been shown in Figure 1. In the factor graph, aspect polarities are represented by variables (round nodes in the figure), and the influence of DNN output and linguistic implications are represented by factors (box nodes in the figure). The value of a variable indicates its polarity. There are two types of variables: *evidence variable* and *inference variable*. The evidence variables represent the easy instances, whose sentiment polarities can be directly determined by explicit linguistic hints with high accuracy. They participate in the process of MLN inference, but their values are specified beforehand and remain unchanged throughout the whole process. The inference variables represent the more challenging instances. Their values should be instead inferred based on the constructed factor graph.

Additionally, there are four types of factors: *DNN factor*, *sentiment factor*, *similar factor* and *opposite factor*. The DNN factor simulates the effect of DNN output on polarity. The sentiment factor captures the influence of sentiment features. The similar factor and opposite factor encode the relations between aspect polarities. Intuitively, a similar factor between two variables indicates that their polarities are similar, while an opposite factor indicates their opposite relation.

5 Linguistic Hint Extraction

In this section, we describe how to identify easy instances, extract sentiment features and mine polarity relations by linguistic hints.

TABLE 3: An example of MLN and its corresponding predicates and constants.

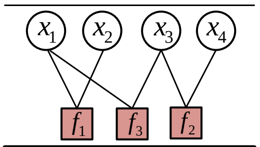
Weight	First-order logic
2.0	$smoke(x) \rightarrow cancer(x)$
3.0	$smoke(x) \wedge friend(x,y) \rightarrow smoke(y)$

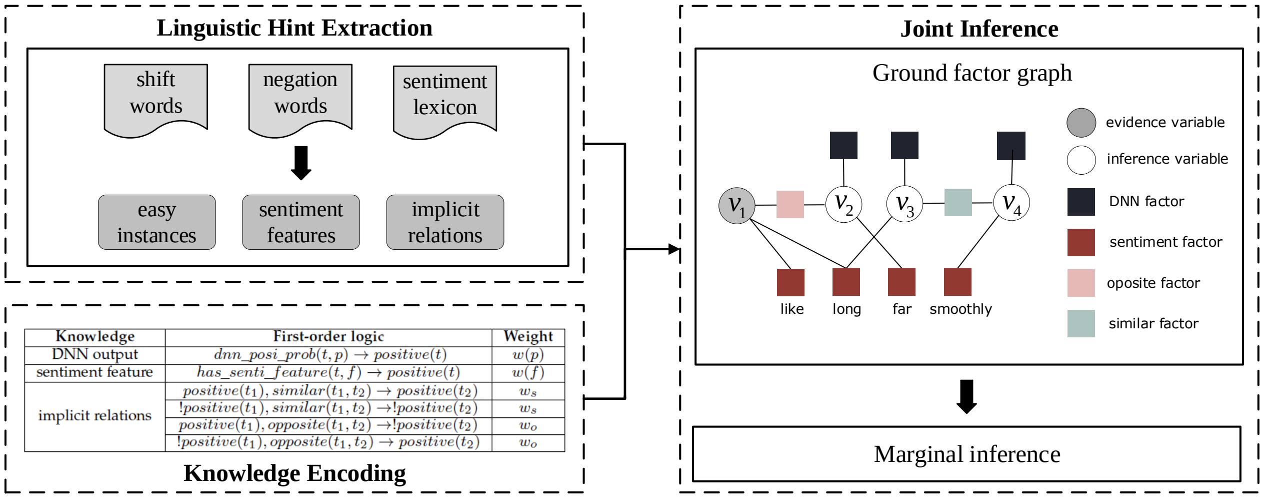
Predicate	Person(P)	Fact
$smoke(x)(x \in P)$	Anna	friend(Anna, Bob)
$cancer(x)(x \in P)$	Bob	
$friend(x,y)(x, y \in P)$		

TABLE 4: Grounding of the example MLN (V_{id} and F_{id} represent variable and factor respectively).

V_{id}	Ground atoms
x_1	smoke(Anna)
x_2	cancer(Anna)
x_3	smoke(Bob)
x_4	cancer(Bob)

F_{id}	Ground formulas
f_1	$smoke(Anna) \rightarrow cancer(Anna)$
f_2	$smoke(Bob) \rightarrow cancer(Bob)$
f_3	$smoke(Anna) \wedge friend(Anna, Bob) \rightarrow smoke(Bob)$

Ground factor graph




Knowledge	First-order logic	Weight
DNN output	$dnn\_posi\_prob(t,p) \rightarrow positive(t)$	$w(p)$
sentiment feature	$has\_senti\_feature(t,f) \rightarrow positive(t)$	$w(f)$
implicit relations	$positive(t_1), similar(t_1,t_2) \rightarrow positive(t_2)$	w_s
	$!positive(t_1), similar(t_1,t_2) \rightarrow !positive(t_2)$	w_s
	$positive(t_1), opposite(t_1,t_2) \rightarrow !positive(t_2)$	w_o
	$!positive(t_1), opposite(t_1,t_2) \rightarrow positive(t_2)$	w_o

Fig. 1: The framework overview of SenHint.

5.1 Identifying easy instances

The existing lexicon-based approaches essentially reason about polarity by summing up the polarity scores of the sentiment words in a sentence. The score of a sentiment word indicates its intensity of sentiment, which increases with the absolute value of score. Since negation words can effectively reverse polarity, they usually perform negation detection for each sentiment word by examining whether there is any negation in its neighboring words [12].

Unfortunately, the lexicon-based approaches are prone to error under some ambiguous circumstances. Firstly, the presence of contrast (e.g. *but* and *although*), hypothetical (e.g. *if*) or condition (e.g. *unless*) connectives could significantly complicate polarity detection. For instance, the sentence "would be a very nice laptop if the mousepad worked properly" contains only the positive sentiment words "nice" and "properly", but it holds negative attitude due to the presence of the hypothetical connective "if". Secondly, the presence of negation words involving long-distance dependency could also make the task challenging. For instance, in the sentence "I don't really think the laptop has a good battery life", the negation word "don't" reverses the polarity, but it is far away from the sentiment word "good". Unfortunately, the existing approaches for negation detection based on local neighborhood can not work properly in the circumstance of long-distance dependency. Finally, a sentence may not contain strong sentiment words, or even if it does, multiple sentiment words may hold conflicting polarities. For instance, consider the sentence "To be honest, i am a little disappointed and considering returning it". Since it contains both the positive word "honest" and the negative word "disappointed", its true polarity is not easily detectable based on sentiment word scoring.

Therefore, for easy instance identification, SenHint chooses to exclude the instances with the aforementioned ambiguous patterns. Specifically,

Definition 3. *[Easy Instances]* SenHint identifies an aspect polarity as an easy instance if and only if the sentence expressing opinions about the aspect satisfies the following three conditions:

- It contains at least one strong sentiment word, but does not simultaneously contain any sentiment word holding the conflicting polarity;
- It does not contain any contrast, hypothetical or condition connective;
- It does not contain any negation word involving long-distance dependency;

In SenHint, the polarity of an easy instance is simply determined by the polarity of its strong sentiment word.

SenHint considers a sentiment word as *strong* if and only if the absolute value of its score exceeds a pre-specified threshold (e.g. 1.0 in our experiment, where the scores of sentiment words are normalized into the interval of [-4,4]). Moreover, a negation word is supposed to involve long-distance dependency if and only if it is not in the neighboring 3-grams preceding any sentiment word. We illustrate the difference between the easy and challenging instances by Example 1.

Example 1. [Easy Instances] In a phone review, the sentence "the screen is not good for carrying around in your bare hands ", which expresses the opinion about "screen", is an easy instance, because the sentiment word "good" associated with the local negation cue "not" strongly indicates the negative sentiment. In contrast, the sentence "I don't know why anyone would want to write a great review about this battery", which expresses the opinion about "battery", is not an easy instance. Even though it contains the strong sentiment word "great", it includes the negation word "don't" involving long-distance dependency. Similarly, the sentence "I like this laptop, the only problem is that it can not last long time" is not an easy instance, because it contains both the positive and negative words (e.g. "like" and "problem").

5.2 Extracting sentiment features

Sentiment words usually play an important role in determining the aspect polarities in a sentence. Accordingly, two sentences sharing a sentiment word usually have the same sentiment polarity. Hence, SenHint extracts the common sentiment words from sentences and model their influence by feature factors in the unified MLN model. Sentiment features include both the generic sentiment words in an open-source lexicon developed by Liu [2], or the domain-specific sentiment words [1] that can be automatically mined from the unlabeled review corpora. Since negation words can effectively reverse polarity [64], [65], we also perform negation detection for each sentiment word by examining whether there is any negation in its neighboring words.

To enable more accurate influence modeling, we also propose to filter sentiment features based on the syntactic structure of sentence. Firstly, SenHint uses the constituency based parse tree [66] to identify sentence structure (e.g. compound or complex) and then determines the important part of a sentence based on the structure. Specifically, if a sentence describes only one aspect and has a compound structure with the coordinating conjunction "but", we only retain the sentiment features appearing in the "but" clause. Secondly, in the case that multiple aspects are opined in a sentence, SenHint uses the dependency based parse tree [67] to extract the opinion phrases, each of which is a pair of opinion target and word, for the mapping between the sentiment features and their target aspects. Specifically, it associates an opinion word (corresponding to a sentiment feature) with an aspect if and only if either its opinion target or the opinion word itself is close to the aspect term in vector space. We illustrate sentiment feature extraction by Example 2.

1. http://www.wowbigdata.cn/SenHint/SenHint.html

Example 2. [Sentiment Feature Extraction] Consider the sentence, "I thought learning the Mac OS would be hard, but it is easily picked up", which expresses the opinion about the aspect "os#usability". SenHint extracts "easily" as sentiment feature but not "hard", because the word "hard" does not appear in the "but" clause. Consider another example, "The screen is gorgeous, and the performance is excellent.", which comments on both aspects of "display#quality" and "laptop#performance". SenHint extracts two opinion phrases $\langle screen, gorgeous\rangle$ and $\langle performance, excellent\rangle$, and then reasons that 1) "gorgeous" is a feature of the aspect "display#quality" because its opinion target "screen" is very close to the aspect in vector space; 2) "excellent" is a feature of the aspect "laptop#performance" because the aspect term explicitly appears in the opinion phrase.

5.3 Mining polarity relations

Modeling sentences independently, the existing DNNs for aspect-level sentiment analysis have very limited capability in capturing contextual information at the sentence level. However, sentences build upon each other. There often exist some discourse relations between clauses or sentences that can provide valuable hints for sentiment prediction [68]. The most influential discourse relation is the contrast relation, which is often marked by shift words (e.g. *but* and *however*). Specifically, two sentences connected with a shift word usually have opposite polarities. In contrast, two neighboring sentences without any shift word between them usually have similar polarities.

Based on these observations, SenHint employs rules to extract the similar and opposite relations between aspect polarities based on sentence context. Given two aspect units $t_i = \{r_i, s_i, a_i\}$ and $t_j = \{r_j, s_j, a_j\}$ that occur in the same review (namely $r_i = r_j$), the rules for extracting polarity relations are defined as follows:

1) If the sentences s_i and s_j are identical (s_i=s_j) or adjacent and neither of them contains any shift word, t_i and t_j are supposed to hold similar polarities;
2) If two adjacent sentences s_i and s_j are connected by a shift word and neither of them contains any inner-sentence shift word, t_i and t_j are supposed to hold opposite polarities;
3) If the sentences s_i and s_j are identical and the opinion clauses associated with them are connected by a inner-sentence shift word, t_i and t_j are supposed to hold opposite polarities.

Note that the 3rd rule can be easily checked in the scenario of ATSA, in which opinion clauses can be easily identified because the aspect term explicitly appears in the text. The scenario of ACSA is instead more challenging. SenHint first uses the dependency-based parse tree to extract all the pairs of opinion phrases, and associates an opinion clause with a specific aspect if either its opinion target or opinion word is close to the aspect in the vector space. We illustrate polarity relation mining by Example 3.

Example 3. [Polarity Relation Mining] In the running example shown in Table 1, the aspect polarities in the sentences S_{21} and S_{22} are supposed to be similar based on the 1st rule. Since the sentences S_{11} and S_{12} in R_1 are connected by the shift word of "However", their aspect polarities are reasoned to be opposite based on the 2nd rule. Additionally, consider the sentence "The screen is bright but the processing power is not very good", which expresses the opinions about both "screen" and "processing power". It can be observed that the two opinion clauses are connected by the shift word of "but" within the sentence. Therefore, their polarities are supposed to be opposite based on the 3rd rule.

6 Knowledge Encoding in MLN

Note that SenHint models the easy instances of aspect polarity as evidence variables in MLN. In this section, we describe how to encode the output of DNN, sentiment features and polarity relations in MLN.

6.1 Encoding DNN output

Deep neural networks can automatically learn different levels of representations and thus avoid feature engineering. In this paper, we use the recently proposed gated convolutional networks [3] (GCAE) as an illustrative example. It is worthy to point out that the outputs of other DNNs can be encoded in SenHint in the same way. GCAE, whose model architecture is shown in Figure 2, uses convolutional neural networks and gating mechanisms to selectively output the sentiment features associated with a given aspect. Its output can indicate the influence resulting from multiple levels of features that correspond to different levels of abstraction.

To encode the output of DNN into a MLN, SenHint uses the weighted first-order logic rule expressed by

$$w(p):\ dnn\_posi\_prob(t,p) \rightarrow positive(t), \tag{2}$$

in which the left-hand side (LHS), $dnn\_posi\_prob(t,p)$, predicates that the probability of an aspect unit t having the positive polarity is equal to the value of p, and the right-hand side (RHS), $positive(t)$, is a boolean variable indicating whether the polarity of t is positive. The weight function $w(p)$ denotes the level of confidence on the rule. Observing that the relationship between the weight w and the probability p (for a boolean variable x being true) can be expressed by $p(x=1)=e^w/(1+e^w)$, we define the rule weight as

$$w(p) = ln(\frac{p}{1-p}). \tag{3}$$

According to Eq. 3, $w(p) > 0$ if $p > 0.5$; otherwise, if $p < 0.5$, then $w(p) < 0$. In the case of $w(p) > 0$, a zero value of $positive(t)$ would invoke a cost penalty as desired. In the case of $w(p) < 0$, a positive value for $positive(t)$ would instead invoke a cost penalty.

6.2 Encoding sentiment features

SenHint encodes the influence of sentiment features using the following rule:

$$w(f): has\_senti\_feature(t,f) \rightarrow positive(t), \tag{4}$$

where $has\_senti\_feature(t,f)$ predicates that the aspect unit t has the sentiment feature f, and $w(f)$ denotes the feature weight. Note that the weight of a sentiment feature can be positive or negative. In our implementation, the weight of a sentiment feature is initially set to 1 if it is a positive word in the lexicon, or -1 if it is a negative word. Based on the labeled instances, SenHint learns the weights of sentiment features in joint inference, and their learned values are supposed to reflect their sentiment strengths. For instance, in the factor graph constructed for the running example as shown in Figure 1, the variable v_1, which represents the instance of aspect polarity in the sentence S_{11}, contains two sentiment features "like" and "long", and the sentiment feature of "long" is also shared by v_3, which represents the instance in the sentence S_{21}. Both sentiment features have positive weights, and the weight of "like" holds a higher value than the weight of "long" due to its stronger sentiment intensity.

6.3 Encoding polarity relations

SenHint specifies the influence of similar relation between two aspect polarities by the following two rules:

$$w_s:\ positive(t_1), similar(t_1,t_2) \rightarrow positive(t_2), \tag{5}$$

and

$$w_s: !positive(t_1), similar(t_1,t_2) \rightarrow !positive(t_2), \tag{6}$$

in which w_s denotes a positive rule weight, t_1 and t_2 denote two aspect units and $!positive(t_i)$ denotes the negation of a boolean variable. In MLN, a positive rule weight means that if LHS is true, RHS also tends to be true; otherwise, the rule is violated with a cost penalty. For instance, in the factor graph constructed for the running example as shown in Figure 1, there exists a similar relation between v_3 and v_4, which represent the instances in S_{21} and S_{22} respectively. The encoding rules of Eq. 5 and 6 would force them to hold similar polarity, otherwise a cost penalty would be invoked.

Similarly, SenHint encodes the influence of opposite relation between two aspect polarities by

$$w_o:\ positive(t_1), opposite(t_1,t_2) \rightarrow !positive(t_2), \tag{7}$$

and

$$w_o: !positive(t_1), opposite(t_1,t_2) \rightarrow positive(t_2), \tag{8}$$

in which w_o denotes a positive rule weight.

SenHint interprets rule weight or confidence on rule as the accuracy of mined relations. For presentation simplicity, we use $v(t_i)$ to denote whether the polarity of the aspect unit t_i is positive, and it takes the value of 1 if the polarity is positive, or 0 otherwise. With the polarity of t_1 being positive, the probability of the polarity of t_2 being positive can be computed by

$$p(v(t_2)=1) = e^{w_s}/(1+e^{w_s}). \tag{9}$$

Approximating $p(v(t_2)=1)$ with the accuracy r_{acc}, we can establish the relationship between the rule weight and the relation accuracy by

$$w_s = ln(\frac{r_{acc}}{1-r_{acc}}). \tag{10}$$

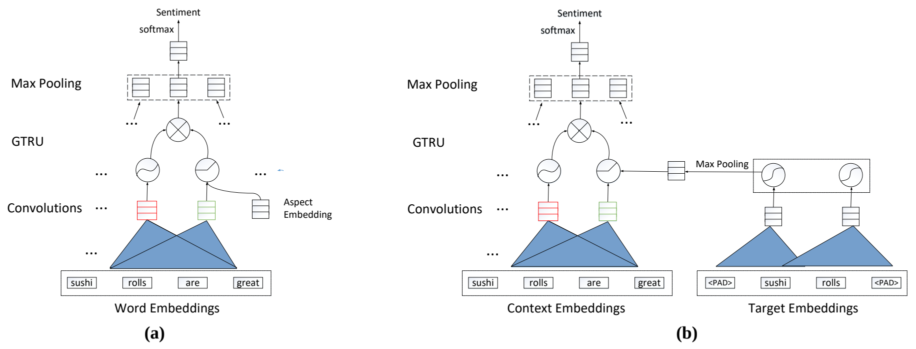


Fig. 2: The GCAE neural networks for ACSA (left) and ATSA (right).

SenHint sets the rule weight w_o specified in 7 and 8 in a similar way. Note that the the higher the estimated accuracy r_{acc}, the higher the rule weights w_s and w_o. For accuracy estimation of mined relations, SenHint first applies the mining rules to the labeled data used for DNN training, and then approximates the accuracy on the test data with the result observed on the training data. Our empirical evaluation in Subsection 8.4 has shown that the accuracies achieved on the test data are generally high, and very similar to the results observed on the training data in most cases.

7 Joint Inference

The MLN model of SenHint is comprised of the formulas specified in Eq. 2, 4, 5, 6, 7 and 8. Based on the model, SenHint first constructs a factor graph, and then estimates the marginal probabilities of inference variables.

For each specified rule, SenHint creates a type of factor in the factor graph. For the formula of Eq. 2, SenHint considers the DNN output as a feature of an inference variable and transforms their relationship into a unary factor that is referred to as *DNN factor*. For the formula of Eq. 4, SenHint encodes the influence of a sentiment feature into a unary factor that is referred to as *sentiment factor*. Unlike the DNN factor, a sentiment factor may be shared by multiple aspect polarities. For the formulas of Eq. 5 and 6, SenHint transforms the similar polarity relations into the binary factors between variables. Since the types of factors corresponding to Eq. 5 and 6 are similar, we refer to both of them as *similar factor* for the sake of presentation simplicity. Similarly, SenHint models the opposite polarity relations as specified in Eq. 7 and 8 by the binary factors between variables, which are referred to as *opposite factor*.

Denoting the *DNN, sentiment, similar, opposite factors* by $\phi_p^{dnn}(\cdot)$, $\phi_f^{sent}(\cdot)$, $\phi^{sim}(\cdot,\cdot)$, $\phi^{opp}(\cdot,\cdot)$ respectively, SenHint defines them as follows:

$$\phi_p^{dnn}(v(t)) = \begin{cases} 1 & v(t)=0, \\ e^{w(p)} & v(t)=1. \end{cases} \tag{11}$$

$$\phi_f^{sent}(v(t)) = \begin{cases} 1 & v(t)=0, \\ e^{w(f)} & v(t)=1. \end{cases} \tag{12}$$

$$\phi^{sim}(v(t_1),v(t_2)) = \begin{cases} 1 & v(t_1) \mathrel{!}= v(t_2), \\ e^{w_s} & v(t_1) \ = v(t_2). \end{cases} \tag{13}$$

$$\phi^{opp}(v(t_1),v(t_2)) = \begin{cases} 1 & v(t_1) \mathrel{!}= v(t_2), \\ e^{-w_0} & v(t_1) \ = v(t_2). \end{cases} \tag{14}$$

where $v(t)$ denotes a boolean variable indicating the polarity of t, $w(p)$, $w(f)$, w_s and w_o denote the rule weights.

Based on the above factors, the factor graph defines a joint probability distribution over its variables V by

$$\begin{aligned} P_w(V) = & \frac{1}{Z}\prod_{v\in V}\phi_p^{dnn}(v(t))\prod_{v\in V}\prod_{f\in F_v}\phi_f^{sent}(v(t)) \\ & \prod_{(t_1,t_2)\in R}\phi^{rel\_type}(v(t_1),v(t_2)) \end{aligned} \tag{15}$$

where F_v denotes the set of sentiment features associated with the variable v, R denotes the sets of polarity relations between aspect units, rel_type denotes the relation type of aspect units t_1 and t_2 (namely sim or opp) and Z denotes a partition function, i.e. normalization constant.

Given a factor graph with some labeled evidence variables, SenHint reasons about the factor weights by minimizing the negative log marginal likelihood as follows

$$\hat{w} = arg\min_{w} -log\sum_{V_I} P_w(\Lambda, V_I), \tag{16}$$

where Λ denotes the observed labels of evidence variables and V_I denotes the set of inference variables. The objective function effectively learns the factor weights most consistent with the label observations of the evidence variables. SenHint optimizes the objective function by leveraging the Snorkel engine, which interleaves stochastic gradient descent steps with Gibbs sampling ones. It has been shown in [69], [70] that similar to contrastive divergence [71], the optimization process can guarantee convergence. Note that in our implementation, the weights $w(p), w_s, s_o$ are automatically set to be fixed values based on the formulas of Eq. 3 and 10, while the weight $w(f)$ is learned by optimizing the objective function. Once the weights are learned, SenHint performs the marginal inference over the factor graph to compute the probability distribution for each inference variable $v(t) \in V$. SenHint uses the Numbskull library [2] for marginal inference.

8 Empirical Evaluation

In this section, we empirically evaluate the performance of SenHint on the benchmark datasets by a comparative study. We compare SenHint with the state-of-the-art DNN models

2. https://github.com/HazyResearch/numbskull

proposed for ACSA and ATSA. For the ACSA tasks, the compared models include:

- **H-LSTM [46].** The hierarchical bidirectional LSTM can model the inter-dependencies of sentences in a review;
- **AT-LSTM [47].** The Attention-based LSTM (AT-LSTM) employs an attention mechanism to concentrate on the key parts of a sentence given an aspect, where the aspect embeddings are used to determine the attention weight;
- **ATAE-LSTM [47].** The Attention-based LSTM with Aspect Embedding (ATAE-LSTM) extends AT-LSTM by appending the input aspect embedding into each word input vector;
- **GCAE [3].** The gated convolutional network employs CNN and gating mechanisms to selectively output the sentiment features according to a given aspect.

For the ATSA, the compared models inlcude:

- **IAN [38].** The interactive attention network interactively learns the attentions in the contexts and targets, and generates the representations for targets and contexts separately;
- **RAM [39].** The multiple-attention network can effectively capture sentiment features separated by a long distance, and is usually more robust against irrelevant information;
- **AOA [40].** The attention-over-attention network models aspects and sentences in a joint way, and can explicitly capture the interaction between aspects and context sentences;
- **TNet [33].** Compared with previous alternatives, the target-specific transformation network can better integrate target information into the word representations.

The rest of this section is organized as follows: Subsection 8.1 describes the experimental setup. Subsection 8.2 presents the comparative evaluation results. Subsection 8.3, Subsection 8.4 and Subsection 8.5 separately evaluate the effect of easy instances, aspect polarity relations and sentiment features on the performance of SenHint. Finally, Subsection 8.6 presents the results of error analysis on SenHint for its future improvement.

8.1 Experimental setup

We used the benchmark datasets in four domains (phone, camera, laptop and restaurant) and two languages (Chinese and English) from the SemEval 2015 task 12 [10] and 2016 task 5 [1]. Our experiments performed 2-class classification to label an aspect polarity as *positive* or *negative*, and thus ignored the neutral instances in our experiments. The statistics of the test datasets are presented in Table 5, in which {#R, #S, #T(ACSA), #T(ATSA)} denote the numbers of {reviews, sentences, aspect category units, aspect term units} respectively. Since there are no labeled aspect terms in the Chinese datasets, we compare SenHint to its alternatives only on the English datasets for ATSA. Note that given a test dataset, the number of instances in its factor graph is equal to the number of aspect category units or aspect term units it contains.

On all the datasets, we used the default split for train and test data. We used Glove embeddings [3] for English data, and word embeddings from Baidu [4] for Chinese data. We employed jieba [5] to tokenize Chinese sentences. For identifying easy instances, we used the Opinion Lexicon [6] and EmotionOntology [7] lexicons for English and Chinese data respectively. For the Chinese lexicon, the scores for sentiment words are normalized into the range of $[-4, 4]$, and we consider a sentiment word *strong* if its absolute sentiment score is at least 1. Due to their limited numbers, we manually specified the negation and shift words, which are summarized in Table 6.

In our experiments, we used the GCAE model to predict the DNN output, because it has been empirically shown to outperform other DNN alternatives. For GCAE training, we used the default parameters [3]. However, SenHint can easily integrate any other DNN model into its MLN. In the implementation of SenHint joint inference, the number of learning and inference epochs is set at 1000, the step size for learning is set at 0.01, the decay for updating step size is set at 0.95, and the regularization penalty is set at $1e-6$. Our implementation codes have also been made open-source [8].

8.2 Comparative Evaluation

We have compared performance on both metrics of accuracy and macro-F1. Note that the metric of macro-F1 is the unweighted average of the F1-score for each label. The detailed comparative results on the ACSA and ATSA tasks are presented in Table 7 and 8 respectively, in which *SenHint(demo)* denotes the original approach presented in our demo paper [15] and *SenHint* denotes the improved approach proposed in this paper. We have highlighted the best performance on each test task by **bold** in the tables. It can be observed that compared with the DNN approaches, the new SenHint achieves better performance on all the test datasets. For ACSA, it outperforms the best DNN model by around 2%-6% on all the test datasets in terms of both accuracy and macro-F1. It achieves the improvement of more than 4 % on 5 out of totally 6 tasks (i.e. PHO16, CAM16, LAP16, LAP15 and RES15). For ATSA, the experimental results are similar. The new SenHint outperforms the best DNN model by around 7% on LAP15 and LAP16, and by around 4% on RES15. Due to the widely recognized challenges of sentiment analysis, the achieved improvements can be considered to be very considerable. These experimental results clearly demonstrate the effectiveness of SenHint.

It is also worthy to point out that *SenHint* consistently performs better than *SenHint(demo)*. The achieved improvements on most tasks are between 1% and 3%. The maximal improvement of around 3.5% is achieved on the LAP16 workload of ATSA. The only exception is PHO16, on which *SenHint* performs slightly worse than *SenHint(demo)* by less than 0.1% if measured by macro-F1. Our experimental results have evidently validated the efficacy of the improved MLN model proposed in this paper.

3. https://nlp.stanford.edu/projects/glove/
4. http://pan.baidu.com/s/1jIb3yr8
5. https://github.com/fxsjy/jieba
6. https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html
7. http://ir.dlut.edu.cn/EmotionOntologyDownload
8. http://www.wowbigdata.cn/SenHint/SenHint.html

TABLE 5: Details of benchmark datasets for ACSA and ATSA task (*PHO*, *CAM*, *LAP* and *RES* refer to the domain phone, camera, laptop and restaurant respectively).

Data	Language	Train				Test			
		#R	#S	#T(ACSA)	#T(ATSA)	#R	#S	#T(ACSA)	#T(ATSA)
PHO16	chinese	140	6330	1333	—	60	3191	529	—
CAM16	chinese	140	5784	1259	—	60	2256	481	—
LAP16	english	450	2500	2715	1478	80	808	751	435
RES16	english	350	2000	2134	1662	90	676	693	578
LAP15	english	277	1739	1864	1049	173	761	868	410
RES15	english	254	1315	1410	1154	96	685	725	508

TABLE 6: Examples of the negation and shift words in English and Chinese.

English	negation words	never, none, nor, not, nothing, no
	shift words	but, however, although, though
Chinese	negation words	没有, 不是, 不够, 不怎么, 不算, 不能
	shift words	但是, 但, 不过, 就是, 虽然

To validate the efficacy of extracted linguistic hints, we have also conducted ablation test on both ACSA and ATSA tasks. The detailed evaluation results have been shown in Table 7 and 8, where *SenHint(w/o easy)*, *SenHint(w/o senti-feats)* and *SenHint(w/o relations)* denote the ablated models with the components of easy instances, sentiment features and polarity relations being removed from SenHint respectively. We can observe that: 1) SenHint achieves better performance than the ablated models in most cases with only a few exceptions. It means that all the extracted linguistic hints are helpful for polarity reasoning; 2) Among the ablated models, SenHint(w/o relations) achieves the overall worst performance, followed by SenHint(w/o senti-feats) and SenHint(w/o easy). It means that the influence of polarity relations on the performance of SenHint is the greatest, followed by sentiment features and easy instances.

It can also be observed that the improvement margins of SenHint over *SenHint(w/o easy)* and *SenHint(w/o senti-feats)* are very similar on the English and Chinese datasets; however, the influence of polarity relations is greater on the English datasets than the Chinese datasets. In our experiments, we have observed that more polarity relations can be extracted from the English datasets than the Chinese datasets, and they are generally accurate. Therefore, as shown in Table 6, *SenHint* outperforms the ablated model of *SenHint(w/o relations)* by more considerable margins on the English datasets than the Chinese datasets.

8.3 Effect of easy instances

In this subsection, we first evaluate the performance of the technique proposed for identifying easy instances, and then evaluate its effect on SenHint.

We compare the performance of our proposed technique with the best DNN model of GCAE. Note that the easy instances are identified by SenHint using the pre-specified rules. Therefore, for SenHint, the percentage of easy instances, which is calculated by dividing the number of easy instances by the total number of instances in a test dataset, is fixed for each test dataset. For fair comparison, we also select the same number of least uncertain instances in a test dataset based on the output of GCAE, and then compare the achieved accuracy of SenHint and GCAE. The detailed results on the ACSA and ATSA tasks are presented in Table 9, in which the first row denotes the percentage of easy instances identified by SenHint, and the following two rows denote the accuracy of GCAE and SenHint respectively. It can be observed that

- A considerable percentage of the instances in a test workload can be identified as easy instances by SenHint: the percentage varies from 35% to 58%;
- SenHint detects the polarities of easy instances with the consistently higher accuracy than GCAE, and the improvement margins are considerable. On PHO16 and CAM16 for ACSA and LAP16 and LAP15 for ATSA, the margins are as large as 9-10%;

We then evaluate the effect of identified easy instances on the performance of SenHint by comparing SenHint-easy with GCAE, in which SenHint-easy represents the MLN model using the outputs of DNN and easy instances but not mined sentiment features and polarity relations. The detailed results are presented in Table 10. It can be observed that the MLN model of using easy instances alone can effectively improve the performance of polarity classification. On the difference between the English and Chinese datasets, we have observed that a higher percentage of instances can be identified as easy on the English datasets, but the achieved accuracy is generally lower. Their effect on the performance of SenHint are however quite similar on the English and Chinese datasets.

8.4 Effect of Polarity Relations

In this subsection, we first evaluate the performance of the technique proposed for mining polarity relations, and then evaluate its effect on the performance of SenHint.

The detailed results on the performance of the mining technique are presented in Table 11, which reports the accuracy of mined relations on both training and test data. As expected, the achieved accuracies on the test data are generally similar to the results obtained on the training data. Most importantly, the accuracy of mined relations is high ($\geq 80\%$) in most cases.

We then compare SenHint-rel with GCAE, in which SenHint-rel denotes the MLN model integrating DNN outputs and mined polarity relations but not easy instances and sentiment features. The comparative results are presented in Table 12. It can be observed that SenHint-rel can effectively improve the performance of DNN. These observations validate the effectiveness of the proposed strategy, which assigns different weights to relations such that a relation with

TABLE 7: Accuracy comparison for ACSA on benchmark datasets.

Model	**PHO16**		**CAM16**		**LAP16**		**RES16**		**LAP15**		**RES15**	
	Acc	**Macro-F1**	**Acc**	**Macro-F1**	**Acc**	**Macro-F1**	**Acc**	**Macro-F1**	**Acc**	**Macro-F1**	**Acc**	**Macro-F1**
H-LSTM	73.30%	72.59%	78.80%	73.04%	78.90%	77.18%	83.10%	79.48%	80.00%	78.25%	77.10%	76.15%
AT-LSTM	72.40%	72.16%	81.70%	77.42%	76.03%	74.73%	85.03%	80.57%	81.03%	79.10%	77.25%	77.00%
ATAE-LSTM	74.48%	73.85%	83.36%	79.59%	79.07%	77.10%	84.66%	80.50%	80.68%	78.97%	79.13%	77.83%
GCAE	76.03%	75.49%	82.49%	76.72%	80.75%	79.24%	86.87%	83.07%	81.96%	80.56%	81.49%	80.45%
SenHint(demo)	80.45%	**80.20%**	86.58%	82.89%	83.07%	81.71%	88.09%	84.73%	84.60%	83.46%	82.50%	81.78%
SenHint(w/o easy)	80.72%	80.08%	87.82%	84.29%	85.57%	84.26%	**89.32%**	**86.01%**	87.28%	86.20%	85.24%	84.58%
SenHint(w/o senti-feats)	80.08%	79.53%	87.53%	83.87%	84.69%	83.28%	89.00%	85.73%	86.84%	85.75%	85.43%	84.84%
SenHint(w/o relations)	80.00%	79.40%	87.82%	84.37%	82.61%	81.24%	87.07%	83.40%	86.08%	85.01%	83.83%	83.06%
SenHint	**80.89**%	80.15%	**88.10%**	**84.47%**	**85.60**%	**84.28%**	89.09%	85.72%	**87.46**%	**86.40%**	**85.84**%	**85.34%**

TABLE 8: Accuracy comparison for ATSA on benchmark datasets.

Model	**LAP16**		**RES16**		**LAP15**		**RES15**	
	Acc	**Macro-F1**	**Acc**	**Macro-F1**	**Acc**	**Macro-F1**	**Acc**	**Macro-F1**
AT-LSTM	74.85%	72.39%	84.43%	77.50%	77.51%	74.41%	75.43%	71.57%
ATAE-LSTM	75.08%	71.93%	84.60%	76.82%	77.66%	73.83%	74.13%	69.67%
GCAE	78.34%	75.74%	88.86%	81.93%	81.37%	79.08%	77.60%	71.81%
IAN	74.02%	71.90%	85.12%	77.01%	79.27%	76.30%	75.00%	69.34%
RAM	77.47%	75.33%	85.81%	78.44%	78.58%	76.33%	73.23%	66.33%
AOA	74.94%	72.27%	87.02%	75.83%	80.73%	77.84%	73.43%	69.71%
TNet	75.86%	73.85%	87.20%	80.20%	80.00%	78.88%	75.20%	71.32%
SenHint(demo)	82.75%	80.98%	89.65%	83.25%	86.47%	84.75%	81.17%	77.53%
SenHint(w/o easy)	85.47%	83.82%	**89.79%**	84.08%	87.90%	86.28%	80.87%	76.73%
SenHint(w/o senti-feats)	84.78%	83.22%	89.69%	84.03%	87.66%	86.10%	**81.77%**	**78.10%**
SenHint(w/o relations)	84.32%	82.53%	88.93%	82.91%	87.27%	85.66%	81.02%	77.02%
SenHint	**86.19%**	**84.65%**	89.68%	**84.12%**	**87.98%**	**86.41%**	81.66%	77.98%

TABLE 9: Effectiveness evaluation of identifying easy instances (*Prop* and *Acc* denote the proportion and achieved accuracy of identified easy instances).

	ACSA						**ATSA**			
	PHO16	**CAM16**	**LAP16**	**RES16**	**LAP15**	**RES15**	**LAP16**	**RES16**	**LAP15**	**RES15**
Prop	35.73%	43.87%	46.34%	55.70%	54.72%	47.17%	44.83%	58.82%	58.05%	50.39%
Acc(GCAE)	86.35%	87.49%	90.80%	92.75%	88.76%	88.54%	86.46%	93.94%	87.06%	87.34%
Acc(SenHint)	95.24%	98.58%	93.68%	93.01%	95.16%	93.57%	96.92%	93.24%	96.22%	93.75%

TABLE 10: Performance comparison between GCAE and SenHint-easy.

	ACSA						**ATSA**			
	PHO16	**CAM16**	**LAP16**	**RES16**	**LAP15**	**RES15**	**LAP16**	**RES16**	**LAP15**	**RES15**
GCAE	76.03%	82.49%	80.75%	86.87%	81.96%	81.49%	78.34%	88.86%	81.37%	77.60%
SenHint-easy	79.23%	87.32%	82.13%	86.97%	85.50%	83.82%	83.03%	88.51%	86.75%	80.88%

TABLE 11: Effectiveness evaluation of polarity relation mining.

		ACSA						**ATSA**			
Relation type	**Data type**	**PHO16**	**CAM16**	**LAP16**	**RES16**	**LAP15**	**RES15**	**LAP16**	**RES16**	**LAP15**	**RES15**
similar relations	train	89.39%	88.89%	92.57%	95.12%	93.39%	96.07%	91.20%	94.55%	91.94%	95.60%
	test	85.71%	92.13%	93.38%	95.34%	90.51%	92.53%	92.02%	95.82%	85.00%	91.28%
opposite relations	train	75.00%	89.29%	83.33%	72.22%	80.00%	75.00%	75.00%	65.71%	71.43%	72.00%
	test	100%	90.00%	50.00%	66.67%	100%	60.00%	50.00%	83.33%	100%	63.64%

TABLE 12: Performance comparison between GCAE and SenHint-rel.

	ACSA						**ATSA**			
	PHO16	**CAM16**	**LAP16**	**RES16**	**LAP15**	**RES15**	**LAP16**	**RES16**	**LAP15**	**RES15**
GCAE	76.03%	82.49%	80.75%	86.87%	81.96%	81.49%	78.34%	88.86%	81.37%	77.60%
SenHint-rel	76.88%	82.58%	83.70%	90.93%	84.72%	82.33%	79.61%	90.94%	83.09%	78.08%

higher accuracy can have greater impact on its connected variables.

8.5 Effect of sentiment features

In this subsection, we evaluate the effect of extracted sentiment features on the performance of SenHint by comparing GCAE with SenHint-sent, in which SenHint-sent denotes the MLN model integrating DNN output and extracted sentiment features but not easy instances and mined polarity relations. Their comparative results are presented in Table 13. We can observe that SenHint-sent can effectively improve the performance of DNN, and the improvements can be as large as 3.0% on the ATSA tasks of LAP16 and LAP15. These experiments validate the effectiveness of the proposed strategy for integrating common sentiment features into the MLN model.

8.6 Error Analysis

For the improvement of SenHint in the future, it is helpful to scrutinize its failure cases. We have categorized the failure cases into the following categories:

- **Lack of linguistic hints.** This type of error occurs when no linguistic hint has been extracted from a sentence. If an instance does not any extracted linguistic hint, its predicted polarity is the same as the DNN output. For instance, consider the single sentence in a review, "I would have kept it but that was the sole reason for my purchase" , which expresses the opinion about "laptop#general". It contains neither sentiment feature nor polarity relation. Since it is mislabeled by DNN, SenHint also fails.
- **Incorrect linguistic hints.** This type of error occurs when the extracted linguistic hints are incorrect. Most of the errors under this category can be further categorized into the following two subcategories: 1) the instances are incorrectly identified as easy; 2) the extracted polarity relations are erroneous. For the first subcategory, consider the sentence, "I have to clean it regularly for it to stay looking good". SenHint identifies it as an easy instance with the positive polarity. However, its true polarity is negative. For the second subcategory, consider two neighboring sentences, "it looks sleek ad gorgeous" and "i find myself adjusting it regularly". Since they are not connected by any shift word, SenHint reasons that their polarities are similar. However, they are indeed opposite. SenHint first identifies the polarity of the first sentence as positive and then incorrectly labels the polarity of the second sentence as positive based on the extracted polarity relation.
- **Ineffectual linguistic hints.** In this case, even though the extracted linguistic hints are correct, they fail to correct the erroneous outputs of DNN. For instance, consider two neighboring instances with the same positive polarity. Even though SenHint correctly extracts the similar polarity relation between them, it may still fails under the following two circumstances: 1) DNN erroneously labels both instances as negative. Since the erroneous outputs of DNN happen to satisfy the supposed relation, SenHint can not flip their polarities; 2) DNN correctly identifies one of them as positive with a lower confidence (e.g. 0.6) while erroneously identifying the other one as negative with a higher confidence (e.g. 0.05). Instead of correcting the error of DNN, SenHint may flip the polarity of the correctly identified instance from positive to negative.

Using the ACSA task on LAP16 as the test case, we have given the relative percentages of different error classes in Table 14. It can be observed that the error class of Lack of Linguistic Hints occupies the largest portion, followed by Incorrect Linguistic Hints, which comes second. Thus, improving the accuracy and coverage of linguistic hints extraction may greatly enhance the performance of SenHint.

9 Conclusion

In this paper, we have proposed the SenHint framework for aspect-level sentiment analysis that can integrate deep neural networks and linguistic hints in a coherent MLN inference model. We have presented the required techniques for extracting linguistic hints, encoding their implications into the model, and joint inference. Our extensive experiments on the benchmark data have also validated its efficacy.

Built on DNN, SenHint still requires considerable training data. It is interesting to observe that provided with sufficient review corpus, employing easy instance detection, extracted sentiment features and polarity relations can potentially make it unnecessary to classify aspect polarity by DNN. In future work, we will explore how to make SenHint perform well while requiring little or even no labeled training data.

Acknowledgments

This work is supported by the Ministry of Science and Technology of China, National Key Research and Development Program (2016YFB1000703), National Natural Science Foundation of China (61332006, 61732014, 61672432, 61472321 and 61502390), and Natural Science Basic Research Plan in Shaanxi Province of China (2018JM6086).

References

[1] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. Al-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. D. Clercq, V. Hoste, M. Apidianaki, X. Tannier, N. V. Loukachevitch, E. Kotelnikov, N. Bel, S. M. J. Zafra, and G. Eryigit, "Semeval-2016 task 5: Aspect based sentiment analysis," in *Proc. 10th Int. Workshop Semantic Eval., SemEval@NAACL-HLT*, 2016, pp. 19–30.

[2] B. Liu, *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers, San Rafael, 2012.

[3] W. Xue and T. Li, "Aspect based sentiment analysis with gated convolutional networks," in *Proc. 56th Ann. Meeting Assoc. Computational Linguistics*, 2018, pp. 2514–2523.

[4] H. H. Do, P. W. C. Prasad, A. Maag, and A. Alsadoon, "Deep learning for aspect-based sentiment analysis: A comparative review," *Expert Syst. Appl.*, vol. 118, pp. 272–299, 2019.

[5] K. Schouten and F. Frasincar, "Survey on aspect-level sentiment analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 3, pp. 813–830, 2016.

[6] X. Ding, B. Liu, and P. S. Yu, "A holistic lexicon-based approach to opinion mining," in *Proc. Int. Conf. Web Search and Web Data Mining*, 2008, pp. 231–240.

TABLE 13: Performance comparison between GCAE and SenHint-sent.

	ACSA						ATSA			
	PHO16	**CAM16**	**LAP16**	**RES16**	**LAP15**	**RES15**	**LAP16**	**RES16**	**LAP15**	**RES15**
GCAE	76.03%	82.49%	80.75%	86.87%	81.96%	81.49%	78.34%	88.86%	81.37%	77.60%
SenHint-sent	78.26%	85.25%	81.67%	87.39%	84.09%	82.00%	81.31%	89.71%	84.23%	78.52%

TABLE 14: Distribution of classification errors.

No.	**Error category**	**Percentage**
1	Lack of linguistic hints	32.11%
2	Incorrect linguistic hints	30.28%
3	Ineffectual linguistic hints	25.69%
4	Others	11.92%

[7] M. Taboada, J. Brooke, M. Tofiloski, K. D. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Comput. Linguist.*, vol. 37, no. 2, pp. 267–307, 2011.

[8] T. Mullen and N. Collier, "Sentiment analysis using support vector machines with diverse information sources," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2004, pp. 412–418.

[9] S. Kiritchenko, X. Zhu, C. Cherry, and S. Mohammad, "Nrc-canada-2014: Detecting aspects and sentiment in customer reviews," in *Proc. 8th Int. Workshop Semantic Eval., SemEval@COLING*, 2014, pp. 437–442.

[10] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, "Semeval-2015 task 12: Aspect based sentiment analysis," in *Proc. 9th Int. Workshop Semantic Eval., SemEval@NAACL-HLT*, 2015, pp. 486–495.

[11] P. M. Domingos and D. Lowd, *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan & Claypool Publishers, San Rafael, 2009.

[12] C. J. Hutto and E. Gilbert, "VADER: A parsimonious rule-based model for sentiment analysis of social media text," in *Proc. 8th Int. Conf. Weblogs Social Media*, 2014.

[13] Z. Teng, D. Vo, and Y. Zhang, "Context-sensitive lexicon features for neural sentiment analysis," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2016, pp. 1629–1638.

[14] Z. Hu, X. Ma, Z. Liu, E. H. Hovy, and E. P. Xing, "Harnessing deep neural networks with logic rules," in *Proc. 54th Ann. Meeting Assoc. Computational Linguistics*, 2016.

[15] Y. Wang, Q. Chen, X. Liu, M. H. M. Ahmed, Z. Li, W. Pan, and H. Liu, "Senhint: A joint framework for aspect-level sentiment analysis by deep neural networks and linguistic hints," in *The Web Conf. (WWW), Demonstrations Proc.*, 2018, pp. 207–210.

[16] E. Cambria, "Affective computing and sentiment analysis," *IEEE Intell. Syst.*, vol. 31, no. 2, pp. 102–107, 2016.

[17] K. Ravi and V. Ravi, "A survey on opinion mining and sentiment analysis: Tasks, approaches and applications," *Knowl.-Based Syst.*, vol. 89, pp. 14–46, 2015.

[18] X. Zhang, J. J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Adv. Neural Inf. Process. Syst.*, 2015, pp. 649–657.

[19] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," in *Proc. 55th Ann. Meeting Assoc. Computational Linguistics, Volume 1: Long Papers*, 2017, pp. 562–570.

[20] Q. Qian, M. Huang, J. Lei, and X. Zhu, "Linguistically regularized LSTM for sentiment classification," in *Proc. 55th Ann. Meeting Assoc. Computational Linguistics*, 2017, pp. 1679–1689.

[21] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy, "Hierarchical attention networks for document classification," in *Proc. Conf. North Amer. Chapter Assoc. Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489.

[22] L. Luo, X. Ao, F. Pan, J. Wang, T. Zhao, N. Yu, and Q. He, "Beyond polarity: Interpretable financial sentiment analysis with hierarchical query-driven attention," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 4244–4250.

[23] Z. Lei, Y. Yang, and Y. Liu, "LAAN: A linguistic-aware attention network for sentiment analysis," in *Proc. The Web Conf. (WWW)*, 2018, pp. 47–48.

[24] Y. Long, L. Qin, R. Xiang, M. Li, and C. Huang, "A cognition based attention model for sentiment analysis," in *Proc. Conf. Empirical Methods Natural Language Process*, 2017, pp. 462–471.

[25] G. Letarte, F. Paradis, P. Giguère, and F. Laviolette, "Importance of self-attention for sentiment analysis," in *Proc. Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP*, 2018, pp. 267–275.

[26] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier, "An unsupervised neural attention model for aspect extraction," in *Proc. 55th Ann. Meeting Assoc. Computational Linguistics*, 2017, pp. 388–397.

[27] L. Luo, X. Ao, Y. Song, J. Li, X. Yang, Q. He, and D. Yu, "Unsupervised neural aspect extraction with sememes," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 5123–5129.

[28] M. Saeidi, G. Bouchard, M. Liakata, and S. Riedel, "Sentihood: Targeted aspect based sentiment analysis dataset for urban neighbourhoods," in *Proc. 26th Int. Conf. Computational Linguistics: Technical Papers*, 2016, pp. 1546–1556.

[29] Y. Ma, H. Peng, and E. Cambria, "Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive LSTM," in *Pro. 32nd Conf. Artif. Intell.*, 2018, pp. 5876–5883.

[30] L. Dong, F. Wei, C. Tan, D. Tang, M. Zhou, and K. Xu, "Adaptive recursive neural network for target-dependent twitter sentiment classification," in *Proc. 52nd Ann. Meeting Assoc. Computational Linguistics*, 2014, pp. 49–54.

[31] D. Tang, B. Qin, X. Feng, and T. Liu, "Effective lstms for target-dependent sentiment classification," in *Proc. 26th Int. Conf. Computational Linguistics*, 2016, pp. 3298–3307.

[32] S. Wang, S. Mazumder, B. Liu, M. Zhou, and Y. Chang, "Target-sensitive memory networks for aspect sentiment classification," in *Proc. 56th Ann. Meeting Assoc. Computational Linguistics, Volume 1: Long Papers*, 2018, pp. 957–967.

[33] X. Li, L. Bing, W. Lam, and B. Shi, "Transformation networks for target-oriented sentiment classification," in *Proc. 56th Ann. Meeting Assoc. Computational Linguistics, Volume 1: Long Papers*, 2018, pp. 946–956.

[34] O. Firat, K. Cho, and Y. Bengio, "Multi-way, multilingual neural machine translation with a shared attention mechanism," in *Proc. Conf. North Amer. Chapter Assoc. Computational Linguistics: Human Language Technologies*, 2016, pp. 866–875.

[35] X. He and D. Golub, "Character-level question answering with attention," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2016, pp. 1598–1607.

[36] J. Cheng, S. Zhao, J. Zhang, I. King, X. Zhang, and H. Wang, "Aspect-level sentiment classification with HEAT (hierarchical attention) network," in *Proc. Conf. Inf. Knowl. Manage.*, 2017, pp. 97–106.

[37] B. Wang and W. Lu, "Learning latent opinions for aspect-level sentiment classification," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.

[38] D. Ma, S. Li, X. Zhang, and H. Wang, "Interactive attention networks for aspect-level sentiment classification," in *Proc. 26th Int. Joint Conf. Artif. Intell*, 2017, pp. 4068–4074.

[39] P. Chen, Z. Sun, L. Bing, and W. Yang, "Recurrent attention network on memory for aspect sentiment analysis," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2017, pp. 452–461.

[40] B. Huang, Y. Ou, and K. M. Carley, "Aspect level sentiment classification with attention-over-attention neural networks," in *Social, Cultural, and Behavioral Modeling - 11th Int. Conf., SBP-BRiMS*, 2018, pp. 197–206.

[41] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier, "Effective attention modeling for aspect-level sentiment classification," in *Proc. 27th Int. Conf. Computational Linguistics*, 2018, pp. 1121–1131.

[42] Q. Liu, H. Zhang, Y. Zeng, Z. Huang, and Z. Wu, "Content attention model for aspect based sentiment analysis," in *Proc. Conf. World Wide Web*, 2018, pp. 1023–1032.

[43] F. Fan, Y. Feng, and D. Zhao, "Multi-grained attention network for aspect-level sentiment classification," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2018, pp. 3433–3442.

[44] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Adv. Neural Inf. Process. Syst.*, 2017, pp. 3856–3866.

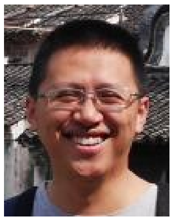
[45] Z. Chen and T. Qian, "Transfer capsule network for aspect level sentiment classification," in *Prco. 57th Conf. Asso. Computational Linguistics*, 2019, pp. 547–556.
[46] S. Ruder, P. Ghaffari, and J. G. Breslin, "A hierarchical model of reviews for aspect-based sentiment analysis," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2016, pp. 999–1005.
[47] Y. Wang, M. Huang, X. Zhu, and L. Zhao, "Attention-based LSTM for aspect-level sentiment classification," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2016, pp. 606–615.
[48] Y. Wang, A. Sun, M. Huang, and X. Zhu, "Aspect-level sentiment analysis using as-capsules," in *Proc. The Web Conf. (WWW)*, 2019, pp. 2033–2044.
[49] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *1st Int. Conf. Learn. Represent., Workshop Track Proc.*, 2013.
[50] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Language Process.: A meeting of SIGDAT, a Special Interest Group of the ACL*, 2014, pp. 1532–1543.
[51] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for twitter sentiment classification," in *Proc. 52nd Ann. Metting Assoc. Computational Linguistics*, 2014, pp. 1555–1565.
[52] D. Tang, F. Wei, B. Qin, N. Yang, T. Liu, and M. Zhou, "Sentiment embeddings with applications to sentiment analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 2, pp. 496–509, 2016.
[53] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, 2011.
[54] P. Fu, Z. Lin, F. Yuan, W. Wang, and D. Meng, "Learning sentiment-specific word embedding via global sentiment representation," in *Prco. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 4808–4815.
[55] P. Singla and P. M. Domingos, "Entity resolution with markov logic," in *Proc. 6th IEEE Int. Conf. Data Mining*, 2006, pp. 572–582.
[56] T. Ye and H. W. Lauw, "Structural constraints for multipartite entity resolution with markov logic network," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 1691–1694.
[57] H. Poon and P. M. Domingos, "Joint inference in information extraction," in *Proc. 22nd AAAI Conf. Artif. Intell.*, 2007, pp. 913–918.
[58] S. Satpal, S. Bhadra, S. Sellamanickam, R. Rastogi, and P. Sen, "Web information extraction using markov logic networks," in *Proc. 20th Int. Con. World Wide Web*, 2011, pp. 115–116.
[59] Z. Chen, Y. Huang, J. Tian, X. Liu, K. Fu, and T. Huang, "Joint model for subsentence-level sentiment analysis with markov logic," *J. Assoc. Inf. Sci. Tech.*, vol. 66, no. 9, pp. 1913–1922, 2015.
[60] C. Zirn, M. Niepert, H. Stuckenschmidt, and M. Strube, "Fine-grained sentiment analysis with structural features," in *Proc. 5th Int. Joint Conf. Natural Language Process.*, 2011, pp. 336–344.
[61] J. Shin, S. Wu, F. Wang, C. D. Sa, C. Zhang, and C. Ré, "Incremental knowledge base construction using deepdive," *PVLDB*, vol. 8, no. 11, pp. 1310–1321, 2015.
[62] Y. Chen and D. Z. Wang, "Knowledge expansion over probabilistic knowledge bases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 649–660.
[63] C. Zhang and C. Ré, "Towards high-throughput gibbs sampling at scale: a study across storage managers," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2013, pp. 397–408.
[64] L. Jia, C. T. Yu, and W. Meng, "The effect of negation on sentiment analysis and retrieval effectiveness," in *Proc. 18th ACM Conf. Inf. Knowl. Manage.*, 2009, pp. 1827–1830.
[65] A. Hogenboom, P. van Iterson, B. Heerschop, F. Frasincar, and U. Kaymak, "Determining negation scope and strength in sentiment analysis," in *Proc. IEEE Int. Conf. Systems, Man and Cybernetics*, 2011, pp. 2589–2594.
[66] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.
[68] E. Pitler, M. Raghupathy, H. Mehta, A. Nenkova, A. Lee, and A. K. Joshi, "Easily identifiable discourse relations," in *Proc. 22nd Int. Conf. Computational Linguistics*, 2008, pp. 87–90.
[67] D. Chen and C. D. Manning, "A fast and accurate dependency parser using neural networks," in *Proc. Conf. Empirical Methods Natural Language Process., A meeting of SIGDAT, a Special Interest Group of the ACL*, 2014, pp. 740–750.
[69] A. Ratner, S. H. Bach, H. R. Ehrenberg, J. A. Fries, S. Wu, and C. Ré, "Snorkel: Rapid training data creation with weak supervision," *PVLDB*, vol. 11, no. 3, pp. 269–282, 2017.
[70] S. H. Bach, B. D. He, A. Ratner, and C. Ré, "Learning the structure of generative models without labeled data," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 273–282.
[71] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

Yanyan Wang Yanyan Wang is a Ph.D student in School of Computer Science in Northwestern Polytechnical University. Her research interests include sentiment analysis and artificial intelligence.

Qun Chen Qun Chen is a professor in School of Computer Science in Northwestern Polytechnical University. His current research interest focuses on the interdisciplinary methodologies and techniques (mostly based on data analysis and machine learning) for a variety of challenging computation tasks (e.g. entity resolution and sentiment analysis).

Murtadha Ahmed Murtadha Ahmed is a Ph.D student in School of Computer Science in Northwestern Polytechnical University. His research interests include sentiment analysis and artificial intelligence.

Zhanhuai Li Zhanhuai Li is a professor in School of Computer Science in Northwestern Polytechnical University. His research interests include data storage and management. He has served as Program Committee Chair or Member in various conferences and committees.



Wei Pan Wei Pan is a associate professor in School of Computer Science in Northwestern Polytechnical University. His research interests include graph processing.

Hailong Liu Hailong Liu is a associate professor in School of Computer Science in Northwestern Polytechnical University. His research interests include data quality management.