A Dual-Channel Framework for Sarcasm Recognition by Detecting Sentiment Conflict

Yiyi Liu[*1,2], **Yequan Wang**[*3], **Aixin Sun**[4], **Xuying Meng**[5], **Jing Li**[6], **Jiafeng Guo**[1,2]
[1]CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
[3]Beijing Academy of Artificial Intelligence, Beijing, China
[4]School of Computer Science and Engineering, Nanyang Technological University, Singapore
[5]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[6]Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates
tshwangyequan@gmail.com,axsun@ntu.edu.sg,jingli.phd@hotmail.com
{liuyiyi17s,mengxuying,guojiafeng}@ict.ac.cn

Abstract

Sarcasm employs ambivalence, where one says something positive but actually means negative, and vice versa. The essence of sarcasm, which is also a sufficient and necessary condition, is the conflict between literal and implied sentiments expressed in one sentence. However, it is difficult to recognize such sentiment conflict because the sentiments are mixed or even implicit. As a result, the recognition of sophisticated and obscure sentiment brings in a great challenge to sarcasm detection. In this paper, we propose a Dual-Channel Framework by modeling both literal and implied sentiments separately. Based on this dual-channel framework, we design the Dual-Channel Network (DC-Net) to recognize sentiment conflict. Experiments on political debates (*i.e.,* IAC-V1 and IAC-V2) and Twitter datasets show that our proposed DC-Net achieves state-of-the-art performance on sarcasm recognition. Our code is released to support research[1].

1 Introduction

Sarcasm is a complicated linguistic phenomenon. Intuitively, it means that one says something positive on surface form, while he/she actually expresses negative, vice versa (Liu, 2012; Merrison, 2008). Take the sentence "*Final exam is the best gift on my birthday*" as an example, the literal sentiment on surface is *positive*, which is reflected by the explicit sentiment words, *i.e.,* "*best gift*". However, the factual part of the text (*i.e.,* "*final exam happens on birthday*") implies that the sentiment expressed is *negative*. This example suggests that it is the sentiment conflict that causes sarcasm linguistically.

However, modeling this linguistic nature of sarcasm is a great challenge due to the difficulty of digging sentiment conflict between the literal and the implied meanings. We know that non-sarcastic texts do not contain implied meaning, so the literal sentiment is consistent with the actual sentiment. But for sarcastic text, there is more than one meaning that coexists in one sentence. The literal meaning and the implied meaning are reflected in different sub-sentences. Even more challenging, sentiments behind the two meanings are mixed or even implicit.

Many existing studies adopt generic classification models for sarcasm recognition (Lou et al., 2021; Ghosh and Veale, 2016). However, these methods directly model the entire sentence without considering the contradictory meanings behind sarcastic texts. There are also studies using contrast patterns (*e.g.,* phrase pair and word pair) as indicators to detect sarcasm, which is approaching the linguistic essence of sarcasm. Riloff et al. (2013); Joshi et al. (2015) detect contrast or incongruity patterns, *i.e.,* the co-occurrence of positive sentiment phrases and negative situational phrases. Tay et al. (2018); Xiong et al. (2019) use attention mechanism to measure the sentiment conflict between word pairs in sarcastic texts. However, these methods emphasize too much on the explicit sentiment conflict on surface form (*i.e.,* word/phrase level), which mainly reflect the literal meaning. As a result, the factual text is underestimated, which expresses the implied sentiment.

Dual-Channel Framework. In this paper, we propose a dual-channel framework to model the lit-

*Indicates equal contribution
[1]https://github.com/yiyi-ict/dual-channel-for-sarcasm

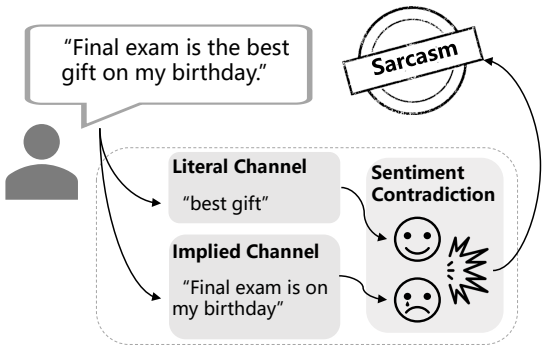


Figure 1: The Dual-Channel Framework for sarcasm recognition.

eral sentiment and the implied sentiment simultaneously. This allows us to leverage the conflict between the two channels in a comprehensive way. Figure 1 depicts the proposed dual-channel framework. In this framework, literal channel and implied channel are used to detect the surface and the hidden meanings separately. Once sentiment conflict is detected, we could determine the existence of sarcasm. The design of our dual-channel framework balances the effect of literal and implied inputs and avoids focusing too much on either one channel while ignoring the other. Our framework covers existing sarcasm patterns, and could be further enhanced to detect more sentiment conflict patterns.

Based on this framework, we develop the Dual-Channel Network (DC-Net) to detect sarcasm. DC-Net contains four modules: decomposer, literal channel, implied channel, and analyzer. In general, sentiment words directly reflect the surface sentiment, while the text without sentiment words reflects the implied sentiment. Hence, we split the sentiment words of input text to literal channel, and the remaining words to implied channel by decomposer. Then we use the literal channel to model surface meaning, and the implied channel to model hidden meaning. Lastly, we use analyzer to recognize the conflict. Experiments on three benchmark datasets (*i.e.,*, IAC-V1, IAC-V2 and Tweets) show that our proposed DC-Net model achieves state-of-the-art performance.

The main contributions of this paper are twofold. First, to the best of our knowledge, the dual-channel framework is the first attempt to explicitly separate literal meaning and implied meaning to recognize sarcasm by detecting sentiment conflict. Second, experiments conducted on benchmark datasets (*i.e.,* IAC-V1/V2 and Tweets) show that DC-Net achieves state-of-the-art performance.

2 Related Work

Prior methods of sarcasm recognition can be divided into traditional models and neural models. There are also methods considering context information, *e.g.,* posting history (Hazarika et al., 2018; Zhang et al., 2016), and user profile (Poria et al., 2016; Kolchinski and Potts, 2018). However, such context may not be always available.

2.1 Traditional Models

Most traditional approaches adopt machine learning methods such as SVM with manually crafted rules or feature engineering. The features include sentiment lexicons (González-Ibáñez et al., 2011; Patra et al., 2016), pragmatic features (*i.e.,* emoticons (González-Ibánez et al., 2011), capitalization, punctuations (Joshi et al., 2015)), and pattern-based features (Riloff et al., 2013) *et al.*. Hee et al. (2018b) utilize common sense to assist sarcasm detection on Twitter. Accordingly, the accuracy of sarcasm recognition highly depends on the quality of features.

Rewriting key parts of a sentence manually is an expensive but effective method. Ghosh et al. (2015) believe that sarcasm involves a figurative meaning which is usually the opposite of literal meaning. They reframe sarcasm recognition as a literal/sarcastic word sense disambiguation problem. Then they paraphrase sarcastic texts manually to obtain target words that cause sarcastic disambiguation. This work is novel but heavily relies on manual paraphrasing and labeling of datasets to find target words. Moreover, target words are mostly limited to sentiment words. As a result, the model is dominated by these explicit sentiment words and ignores the implied channel.

2.2 Neural Models

Ghosh and Veale (2016) propose a model composed of CNN, LSTM and DNN to detect sarcasm. As attention mechanism has led to improvements in various NLP tasks, Tay et al. (2018); Xiong et al. (2019) use attention to capture the relationship of word pairs along with an LSTM to model the entire sentence. Lou et al. (2021) design a GCN-based model combining SenticNet (Cambria et al., 2020), dependency tree and LSTM with GCN (Kipf and Welling, 2017) together, which achieves promising

performance. Similar to previous studies, to better understand sarcasm, many approaches are able to utilize external information such as emoji expressions (Felbo et al., 2017), affective knowledge (Babanejad et al., 2020) and commonsense (Li et al., 2021). Joshi et al. (2017) provide a more comprehensive survey. Moreover, there have been many systems developed for a shared task (Ghosh et al., 2020). These models are rarely designed to reflect the essential features of the sarcasm phenomenon.

3 Dual-Channel Network (DC-Net)

The architecture of the proposed DC-Net is shown in Figure 2. It consists of four modules: *decomposer*, *literal channel*, *implied channel*, and *analyzer*. Given an input text, we use the decomposer to split it into two sub-sentences corresponding to the two channels. Then we use these two channels to derive literal and implied representations independently. Lastly, the analyzer predicts whether the text is sarcastic or not by detecting sentiment conflict.

3.1 Decomposer

The decomposer module is designed to split input text to the literal and implied channels. From numerous sarcastic corpora, we observe that sarcastic texts often contain evident sentiment words. More specifically, the literal channel itself is to reflect the intuitive sentiment. So it is reasonable to use sentiment lexicons as direct keywords. The remaining text expresses the implied sentiment. For example, sentiment words of input text (*e.g.,* "best gift") represent *positive*, while the remaining part (*e.g.,* "Final exam is on my birthday") implies the *negative* sentiment. Shown in Table 1, proportion of texts that contain sentiment words ranges from 88% to 96% in three datasets. Hence, using sentiment words to split input well serves the purpose.

Considering a text $W_T = \{w_1, w_2, \ldots, w_N\}$ with N words, we decompose it into two pieces: the sentiment words W_L, and the remaining text W_D (see Figure 2). W_L is fed to the literal channel, and W_D to the implied channel. In this process, we use the sentiment lexicon released in Wilson et al. (2005) to pick up sentiment words. If no sentiment words are matched from the given text, the original text is used as the literal channel's input, which is the same as the implied channel. Note that in quite a few texts, sentiment words are adjectives or adverbs, deleting them from sentences has no much impact on the overall semantics. Although the text is not normative as expected after decomposing, we do not fill in the full text with placeholders.

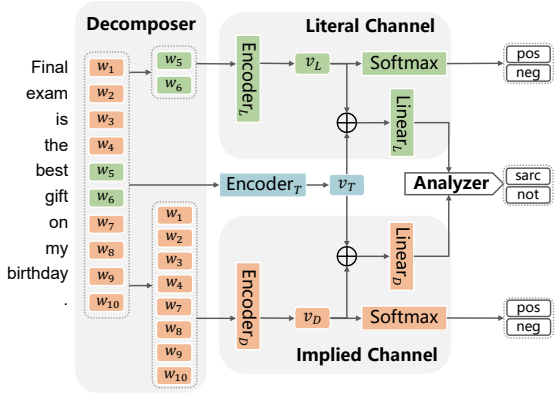


Figure 2: The architecture of the DC-Net.

3.2 Literal Channel

The literal channel includes an encoder, two linear layers, and a softmax classifier. Encoder$_L$ is used to encode the literal text W_L. Then we can get the literal representation v_L through

$$v_L = \text{Encoder}_L(W_L). \tag{1}$$

Next, we use a softmax layer to compute the literal sentiment distribution based on the literal representation v_L.

$$\mathcal{P}_l = \text{softmax}(W_r v_L + b_r), \tag{2}$$

where, W_r and b_r are parameters of the linear layer.

Considering the semantic complexity of sarcastic texts, a single representation of sentiment words may lose context information. So we adopt another Encoder$_T$ to encode the original text W_T and obtain the representation of the entire text v_T through

$$v_T = \text{Encoder}_T(W_T). \tag{3}$$

Last, we concatenate the literal state v_L and the entire text's state v_T, followed by a linear layer and ReLU activation function to reduce dimension. Briefly, the final representation $v_L^{'}$ of the literal channel could be formulated as:

$$v_L^{'} = \text{ReLU}(W_l[v_L; v_T] + b_l), \tag{4}$$

where W_l and b_l are parameters of the second linear layer.

3.3 Implied Channel

In the implied channel, we also adopt an Encoder with the same structure but different parameters to encode the implied input text W_D, and the representation of the implied channel is formulated as:

$$v_D = \text{Encoder}_D(W_D). \tag{5}$$

Similarly, we use $\mathrm{softmax}$ to calculate the implied sentiment distribution based on the implied hidden state v_D:

$$\mathcal{P}_d = \mathrm{softmax}(W_z v_D + b_z), \tag{6}$$

where W_z and b_z are parameters.

Again, we concatenate the implied hidden state v_D with the entire text's hidden state v_T, followed by a linear layer and an activation layer ReLU. The final representation $v_D^{'}$ of the implied channel is formulated as:

$$v_D' = \text{ReLU}(W_d[v_D; v_T] + b_d), \tag{7}$$

where W_d and b_d are parameters.

Note that the structures of the two channels are symmetrical. However, the two encoders in the two channels do not share parameters, and their inputs are different. Since both channels are not specific to particular encoders, the dual-channel framework is able to adapt to mainstream encoders, *e.g.,* LSTM (Hochreiter and Schmidhuber, 1997), CNN (Kim, 2014), Recursive Neural Network (Socher et al., 2011), BERT (Devlin et al., 2019) *et al.*. In DC-Net, we adopt Bi-LSTM as encoders for both channels.

3.4 Analyzer

The analyzer is designed to measure the conflict between the literal and the implied channels. We concatenate the literal representation $v_L^{'}$ and the implied representation $v_D^{'}$ and feed the result to a $\mathrm{softmax}$ layer. Other analyzers such as subtraction or cosine similarity also fit our design.

$$\mathcal{P}_s = \mathrm{softmax}(W_p([v_L^{'}, v_D^{'}]) + b_p), \tag{8}$$

where W_p and b_p are parameters.

Although sarcasm has a strong correlation to literal sentiment and implied sentiment, we do not have gold labels for both sentiments. Hence, requesting the model to directly output sentiments on both channels may confuse the model. For this reason, we develop the objective function of sarcasm classification by adding objectives of the literal and implied channels.

3.5 Training Objective

The training objective of the proposed DC-Net model considers three aspects. One is to minimize the cross-entropy loss of the sarcasm probability distribution. The other two are to minimize the cross-entropy losses of the literal and that of the implied sentiment probability distributions respectively.

Sarcasm Objective. The sarcasm objective is to ensure the basic ability of detection. Hence, we use cross-entropy loss of sarcasm classification. The objective J_s is formulated as:

$$J_s(\theta) = \sum \text{cross-entropy}(y_s, \mathcal{P}_s), \tag{9}$$

where $\mathcal{P}_s$ denotes the sarcasm probability distribution of the text. The groundtruth of the sarcasm label is y_s.

Literal Sentiment Objective. Due to the expensive manual annotations, we use sentiment words for approximate labeling, which is widely used in Eisenstein (2017); Taboada et al. (2011); Hu and Liu (2004). In our implementation, we determine the literal sentiment label based on the number of words with positive sentiment and the words with negative sentiment in input text. For sarcastic texts, if the number of positive words is greater than that of negative words, the literal sentiment label is positive and the implied sentiment label is negative, and vice versa. For non-sarcastic texts, both the literal sentiment label and the implied sentiment label are the same, determined by the number of positive/negative sentiment words.

The literal sentiment classification objective is then formulated as:

$$J_l(\theta) = \sum \text{cross-entropy}(y_l, \mathcal{P}_l), \tag{10}$$

where $\mathcal{P}_l$ is the literal sentiment probability distribution. The label generated by the labeling processing of the literal sentiment is y_l.

Implied Sentiment Objective. We observe that literal sentiment and implied sentiment of sarcastic texts are often opposite. Using the implied labels based on the automatic labeling processing, we calculate the implied sentiment classification objective by

$$J_d(\theta) = \sum \text{cross-entropy}(y_d, \mathcal{P}_d), \tag{11}$$

where $\mathcal{P}_d$ denotes the implied sentiment probability distribution. The label generated by the labeling processing of the implied sentiment is y_d.

Considering these three objectives, we obtain the final objective function L by adding them together:

$$L(\theta) = \lambda_1 J_s(\theta) + \lambda_2 J_l(\theta) + \lambda_3 J_d(\theta), \quad (12)$$

where θ is the parameter set of the model. λ_1, λ_2 and λ_3 are used to leverage the contributions of the three objectives.

4 Experiment

4.1 Datasets and Implementation Details

We conduct experiments on three benchmark datasets: IAC-V1, IAC-V2, and Tweets. These datasets do not contain context information such as historical tweet posts and user profiles. All of them have been widely used in evaluating sarcasm detection.

- **IAC-V1** is collected from online political debates forum[2]. It is the subset of the Internet Argument Corpus (Lukin and Walker, 2013). The written language of IACs is English. Each instance, typically a sentence, is annotated with sarcasm label, either "sarcasm" or "non-sarcasm". Compared to tweets, texts of IAC are much longer and more normative.

- **IAC-V2** (Oraby et al., 2016) contains more data than IAV-V1 (the two versions have a few overlaps). IAC-V2 divides sarcasm into three sub-types, (*i.e.,* general sarcasm, hyperbole, and rhetorical questions). We use the largest subset (general sarcasm) in our experiments.

- **Tweets** dataset written in English is proposed in SemEval 2018 Task 3 Subtask A (Hee et al., 2018a). Each instance (*i.e.,* a sentence) is labeled sarcastic or non-sarcastic. There are three variations of the text in this dataset: (i) original texts, (ii) texts with hashtags removed, and (iii) texts with hashtags and emoji expressions removed. Hashtags like "#not", "#sarcasm", and "#irony", are originally obtained from users. The hashtags are also used as prior knowledge for collecting sarcastic posts. In our experiments, we used the version without hashtags.

Table 1 reports the statistics. We observe that more than 88% of the texts contain sentiment word(s). Hence, it is reasonable to decompose the original text into sentiment words and non-sentiment words, as inputs to the literal channel and implied channel, respectively. The number of instances in the three datasets is between $1k$ and $6k$. All three datasets are class-balanced. The ratio of sarcastic instances and non-sarcastic instances is nearly 1:1. Due to the small size, the split of train/valid/test is important to avoid overfitting. For Tweets dataset, we follow the official train/test split. Then we randomly select 5% from training as valid sub-dataset. There is no official train/valid/test split for the two IAC datasets, so we split IAC datasets following the same ratio of Tweets. The baselines papers do not provide the split (or not conduct experiments on IAC datasets). So we cannot directly adopt the results of baselines reported in their original papers. Hence, we re-implement all baseline models on IAC-V1 and IAC-V2 datasets.

Table 1: Statistics of datasets. Avg ℓ denotes the average length of texts in the number of tokens. s ratio is the proportion of texts that contain sentiment words.

Dataset	Train	Valid	Test	Avg ℓ	s ratio
IAC-V1[3]	1,596	80	320	68	91%
IAC-V2[4]	5,216	262	1,042	43	96%
Tweets[5]	3,634	200	784	14	88%

There are another three datasets for sarcasm detection. Riloff et al. (2013) and Ptáček et al. (2014) propose another two datasets based on Tweets, but they only provide tweet IDs. Due to modified authorization status, lots of tweets are unavailable or deleted. For this reason, we could not experiment on these two Tweet datasets. Khodak et al. (2018) build a large self-annotated dataset from the Reddit forum platform. This dataset contains rich context information including posts, comments, responses, and authors. Since our work focuses on text-based sarcasm recognition, we do not use this dataset.

Implementation Details. We use 300-dimensional Glove (Pennington et al., 2014) embeddings to initialize word vectors. There is a checkpoint every 16 mini-batch, and the batch size is 32. For Tweets dataset, the dropout on embeddings is set to 0, while for IAC datasets it is set to 0.5. Adam (Kingma and Ba, 2015) is used to optimize

[2] http://www.4forums.com/political/

[3] https://nlds.soe.ucsc.edu/sarcasm1

[4] https://nlds.soe.ucsc.edu/sarcasm2

[5] https://github.com/Cyvhee/SemEval2018-Task3

Table 2: The precision, recall, and macro $F1$ of sarcasm recognition. The results marked with * are from Hee et al. (2018a). The best results are in boldface and second-best underlined.

Model	IAC-V1				IAC-V2				Tweets			
	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1	Acc.
UCDCC	58.6	58.6	58.5	58.5	67.1	67.0	67.0	67.0	**78.8***	66.9*	72.4*	**79.7***
THU-NGN	64.4	64.3	64.2	64.3	73.3	73.3	73.3	73.3	63.0*	**80.1***	70.5*	73.5*
Bi-LSTM	64.6	64.6	64.6	64.6	79.8	79.7	79.7	79.7	71.8	71.7	71.7	73.0
AT-LSTM	65.9	65.5	65.3	65.5	76.7	76.2	76.1	76.2	70.8	71.6	70.0	70.2
CNN-LSTM-DNN	61.5	61.2	60.9	61.1	75.4	75.3	75.2	75.3	71.9	72.9	71.9	72.3
MIARN	65.6	65.2	64.9	65.2	75.4	75.3	75.2	75.3	68.6	68.8	68.8	70.2
ADGCN	64.3	64.3	64.3	64.3	81.0	80.9	80.9	80.9	72.6	73.2	72.8	73.6
DC-Net	**66.6**	**66.5**	**66.4**	**66.5**	**82.2**	**82.1**	**82.1**	**82.1**	76.4	77.5	**76.3**	76.7

our model. The parameters β_1 and β_2 of Adam are set to 0.9 and 0.999. The learning rates for model parameters except word vectors are 1e-3, and 1e-4 for word vectors. Our model is implemented with Pytorch[6] (version 1.7.0).

On IAC datasets, all of the loss contributions $\lambda_1, \lambda_2, \lambda_3$ of our DC-Net model are set to 1. On Tweets, they are set to 1, 1e-4, and 3e-1, respectively. The hyperparameters are searched over the validation sub-dataset.

4.2 Compared Methods

We evaluate our model against the following baselines:

UCDCC (Ghosh and Veale, 2018) is a siamese LSTM model exploiting Glove word embedding features. The method designs a lot of rules to preprocess Twitter data. It achieves the best performance on SemEval 2018 Task 3 Subtask A.

THU-NGN (Wu et al., 2018) consists of densely connected LSTMs based on word embeddings, sentiment features, and syntactic features. It ranks second on SemEval 2018 Task 3 Subtask A.

Bi-LSTM (Hochreiter and Schmidhuber, 1997) is a variant of RNN, which could learn long-term dependencies and bidirectional information.

AT-LSTM (Wang et al., 2016) is an LSTM model followed by a neural attention mechanism. It could attend the important part of the input.

CNN-LSTM-DNN (Ghosh and Veale, 2016) is a combination of CNN, LSTM, and DNN. It stacks two layers of convolution and two LSTM layers, then passes the output to a DNN for prediction.

MIARN (Tay et al., 2018) learns the intra-sentence relationships of word pairs and the sequential relationships of a given text.

ADGCN (Lou et al., 2021) is a GCN-based method with sentic graph and dependency graph[7]. The initial input of GCN is the hidden state of Bi-LSTM.

4.3 Main Experiment Results

Table 2 shows that our DC-Net achieves the best macro $F1$ results across all datasets. On Tweets dataset, DC-Net achieves about 3.5% improvement in $F1$ score than the best baseline. On IAC-V2 dataset, our model outperforms the second-best by 1.2% in $F1$. Surprisingly, compared with the basic encoder model Bi-LSTM, our DC-Net boosts the performance up to 5% and 3% respectively on Tweets and IAC-V2, demonstrating the effectiveness of our dual-channel design. For Tweets dataset, the average length of texts is 14 words, which leads to a lack of information for sarcasm recognition. Nevertheless, our DC-Net improve 3.5% on $F1$ compared with the previous state-of-the-art ADGCN.

Interestingly, UCDCC achieves the best precision of 78.8% and accuracy of 79.7% on Tweets dataset. Besides, THU-NGN gets the best recall at 80.1% on Tweets. This is because UCDCC designs targeted rules to preprocess the input text and it achieves the best performance on SemEval 2018 Task 3 Subtask A. Rules could improve precision effectively, but they are hard to take recall into account at the same time. So the $F1$ is not good enough. The last place performance of UCDCC on IAC-V1/V2 also supports this point. These designed rules are hard to fit missing instances and other domains. Similarly, THU-NGN uses linguistic knowledge such as sentiment and syntactic, so it

[6] https://pytorch.org

[7] We employ spaCy toolkit to derive dependency tree.

Table 3: The precision, recall, and macro $F1$ of models including BERT, DC-Net with BERT as Encoder, and DC-Net with Bi-LSTM as Encoder.

Model	Tweets			
	Pre.	Rec.	F1	Acc.
BERT	69.1	67.6	68.1	71.6
DC-Net (w/ BERT)	70.2	70.7	70.4	71.3
DC-Net (w/ Bi-LSTM)	**76.4**	**77.5**	**76.3**	**76.7**

achieves the highest recall on Tweets but it cannot perform equally well on other datasets. That is, rules have limitations in handling this task.

The previous state-of-the-art ADGCN achieves second-best on IAC-V2 and Tweets. However, on IAC-V1 dataset, ADGCN performs not as well as the result reported in their paper. IAC-V1 dataset is relatively small so the train/valid/test split has a significant impact. Our experiments also show that MIARN's performance is not as good as expected. This indicates that the basic utilization of word pair correlation is not enough to improve the performance of sarcasm detection. Bi-LSTM, AT-LSTM, and CNN-LSTM-DNN methods are all based on LSTM. Thus the performances of these models on Tweets and IAC-V1 are close.

4.4 Comparison with BERT

BERT has contributed to significant improvements on various NLP tasks. To do a comprehensive comparison, we apply the dual-channel framework to BERT (Devlin et al., 2019) model by using BERT as the encoder. The new model with BERT is named DC-Net (w/ BERT). Table 3 reports the experimental results.

As expected, the DC-Net (w/ BERT) model achieves significant improvement compared with the basic BERT. This result shows that our dual-channel framework is adaptable and effective. Interestingly, we observe that BERT-based methods perform not well enough compared with its huge improvement on other NLP tasks. This can be attributed to the fact that the corpus of pre-trained BERT contains more deterministic data (*e.g.,* only one meaning without sentiment conflict). However, sarcasm is a niche linguistic phenomenon. The poor performance of BERT further reinforces that sarcasm recognition is a difficult task. It tells us that applying well-performed classification methods directly is difficult to achieve desirable performance.

Table 4: Ablation study on Tweets dataset. J_s denotes using sarcasm loss only. J_s+J_d means using sarcasm and implied loss. J_s+J_l means using sarcasm and literal loss. J_s+J_l+J_d denotes using sarcasm, literal, and implied loss.

Objective	Tweets			
	Pre.	Rec.	F1	Acc.
J_s	74.6	75.4	74.8	75.4
J_s+J_d	74.2	75.2	74.0	74.4
J_s+J_l	73.0	74.0	72.8	73.1
J_s+J_l+J_d	**76.4**	**77.5**	**76.3**	**76.7**

4.5 Ablation Study

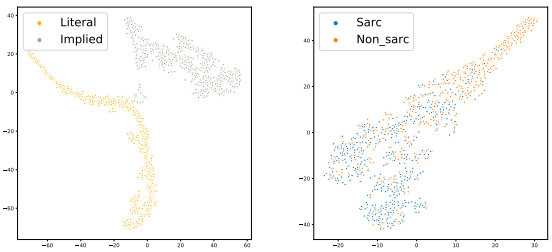
Recall that the model training (see Section 3) contains three objectives: sarcasm recognition, literal sentiment classification, and implied sentiment classification. To study the effect of the three objectives, we conduct ablation study on Tweets dataset.

Table 4 lists the result of ablation study. As expected, the model with both literal and implied losses performs the best. Interestingly, the model using sarcasm recognition loss with single channel loss (*i.e.,* literal and implied) performs worse than the model using only sarcasm recognition loss. This is because adding literal and implied sentiment classification objectives interferes with the judgment of the model. By adding both literal and implied sentiment classification losses, the model's performance improves 1.5 points in $F1$ score. This is very important because it reveals that the dual channels are effective. There is no effect or the opposite effect when single channel is applied alone. However, once dual-channel is used, the performance improves largely. It reveals that the dual channels complement each other. Conflict detection could recognize sarcasm when both of them are considered.

4.6 Effectiveness of DC-Net by Visualization

To verify the rationality and effectiveness of our proposed DC-Net, we adopt t-SNE (Van der Maaten and Hinton, 2008) to visualize high-dimensional vector representations based on the test sub-dataset of IAC-V2 (with largest data).

To figure out the effect of each channel, we visualize the representations of the literal channel and the implied channel. Figure 3(a) shows the visualization of literal representation $v_L^{'}$ and implied representation $v_D^{'}$. Recall that the decomposer module splits the original text into sentiment words and the



(a) Literal and implied reps. of each channel

(b) Sarc. and non-sarc. reps. in analyzer module

Figure 3: Results of t-SNE visualization

remaining. We observe that there is a clear separation between literal and implied representations from Figure 3(a). This strongly indicates that our dual-channel framework is capable of effectively separating the representations of the two channels.

To get into the essence of sarcastic and non-sarcastic texts, we visualize the sarcastic and non-sarcastic representations. Figure 3(b) shows the sarcastic and non-sarcastic representations in the analyzer module. We observe that non-sarcastic texts focus on the upper right corner, while sarcastic texts scatter on the lower left corner. It reveals that the sarcasm patterns are complex and changeable. Nevertheless, the dividing line between the two is relatively clear. To this end, explicitly separating the literal and implied channels is necessary and effective. Further, DC-Net makes a distinct difference between sarcastic and non-sarcastic representations, which greatly promotes the performance of the dual-channel framework. We also plot the sarcastic representation and non-sarcastic representation of each channel respectively, which show the same trend as Figure 3(b). So they are not detailed here.

4.7 Flexibility of Dual-Channel Framework

Flexibility of encoder. The dual-channel framework is flexible and generic, and can be realized by plugging in existing sarcasm recognition models, *e.g.,* MIARN, or classification models, *e.g.,* AT-LSTM, Bi-LSTM, and BERT. Therefore, we use these methods as the encoder to examine the flexibility of our proposed framework. The changing range on macro $F1$ from original baseline models to dual-channel models is shown in Table 5.

As expected, the performance of baseline models has different degrees of improvement on all datasets after applying dual-channel framework. For relatively simple models such as MIARN and Bi-LSTM, the improvement could be up to 4.8%. Interestingly, for complex models like BERT, the improvement is up to 2.3%. As we mentioned earlier, the basic BERT performs not good enough because sarcasm is a niche language phenomenon and the training dataset of BERT contains few sarcasm texts. After applying the dual-channel framework to BERT, the performance for sarcasm recognition improves a lot. These indicate that our designed framework is able to fit various encoders with a significant improvement.

Table 5: The macro $F1$ changes from basic models to dual-channel based models.

Basic Model	Changing Range on F1		
	IAC-V1	IAC-V2	Tweets
AT-LSTM	↑ 0.4	↑ 1.1	↑ 1.5
BERT	↑ 0.4	↑ 1.7	↑ 2.3
MIARN	↑ 1.1	↑ 2.8	↑ 4.8
Bi-LSTM	↑ 1.8	↑ 2.4	↑ 4.6

Table 6: Comparisons of different analyzer methods.

Analyzer	F1		
	IAC-V1	IAC-V2	Tweets
Subtraction	65.1	80.7	75.2
Concatenation	**66.4**	**82.1**	**76.3**

Flexibility of analyzer. The analyzer module is used to measure the difference between the literal channel and the implied channel. As we described in Section 3.4, other analyzer methods such as concatenation and subtraction could be applicable. To this end, we compare different analyzer methods. Table 6 shows the results. We observe that concatenation performs better than subtraction on all datasets. It is because concatenation holds more useful information and DC-Net could compare the difference between the two input representations. However, subtraction only outputs the margin between the two representations. It loses the original values which also contain useful information.

5 Conclusion

In this study, we argue that the essential characteristic of sarcastic text is the conflict between literal and implied sentiments in the same sentence. To this end, we propose a dual-channel framework to recognize sarcasm by decomposing the input text

into the literal channel and the implied channel. Based on this framework, we develop DC-Net. DC-Net is capable of exploiting the literal sentiment by encoding the sentiment words of input text, and exploiting the implied sentiment by encoding the remaining text. Experiments show that the proposed DC-Net achieves state-of-the-art performance.

6 Limitation

Sarcasm as a complex linguistic phenomenon has various patterns, *e.g., text with word/phrase pair sentiment conflict*. Nevertheless, sentiment conflicts are common in sarcasm texts. In this paper, we make the very first attempt to recognize sarcasm by detecting sentiment conflict. More importantly, our proposed dual-channel framework could be further developed to detect more sentiment conflict patterns. For now, we use sentiment words as a static decomposer. This intuitive method can cover common sarcasm patterns but not all. Therefore, how to minimize the dependence on sentiment words is an important research direction.

Another limitation is that we assume that sentiment polarity is decided by the sentiment lexicon approximately in the analyzer module. While the assumption is widely accepted, there is still a gap between approximate label and groundtruth. In the current design, we adopt a soft weighting mechanism to detect sentiment conflict between the two channels. We expect that the model could output the opposite sentiment labels directly, which is a more effective way to express conflict.

Acknowledgments

This work is supported by the National Science Foundation of China (NSFC No. 62106249, No. 61902382 and No. 61972381), the Youth Innovation Promotion Association CAS under Grants No. 20144310, the Lenovo-CAS Joint Lab Youth Scientist Project, and the Foundation and Frontier Research Key Program of Chongqing Science and Technology Commission (No. cstc2017jcyjBX0059).

References

Nastaran Babanejad, Heidar Davoudi, Aijun An, and Manos Papagelis. 2020. Affective and contextual embedding for sarcasm detection. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 225–243.

Erik Cambria, Yang Li, Frank Z. Xing, Soujanya Poria, and Kenneth Kwok. 2020. Senticnet 6: Ensemble application of symbolic and subsymbolic AI for sentiment analysis. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 105–114.

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.

Jacob Eisenstein. 2017. Unsupervised learning for lexicon-based classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3188–3194.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1615–1625.

Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, WASSA@NAACL-HLT 2016, June 16, 2016, San Diego, California, USA*, pages 161–169.

Aniruddha Ghosh and Tony Veale. 2018. Ironymagnet at semeval-2018 task 3: A siamese network for irony detection in social media. In *Proceedings of The 12th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2018, New Orleans, Louisiana, USA, June 5-6, 2018*, pages 570–575.

Debanjan Ghosh, Weiwei Guo, and Smaranda Muresan. 2015. Sarcastic or not: Word embeddings to predict the literal or sarcastic meaning of words. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1003–1012.

Debanjan Ghosh, Avijit Vajpayee, and Smaranda Muresan. 2020. A report on the 2020 sarcasm detection shared task. In *Proceedings of the Second Workshop on Figurative Language Processing, Fig-Lang@ACL 2020, Online, July 9, 2020*, pages 1–11.

Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human*

Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers, pages 581–586.

Roberto I. González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: A closer look. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*, pages 581–586.

Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. CASCADE: contextual sarcasm detection in online discussion forums. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1837–1848.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018a. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2018, New Orleans, Louisiana, USA, June 5-6, 2018*, pages 39–50.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018b. We usually don't like going to the dentist: Using common sense to detect irony on twitter. *Comput. Linguistics*, 44(4).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 168–177.

Aditya Joshi, Pushpak Bhattacharyya, and Mark James Carman. 2017. Automatic sarcasm detection: A survey. *ACM Comput. Surv.*, 50(5):73:1–73:22.

Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762.

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2018. A large self-annotated corpus for sarcasm. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Y. Alex Kolchinski and Christopher Potts. 2018. Representing social media users for sarcasm detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1115–1121.

Jiangnan Li, Hongliang Pan, Zheng Lin, Peng Fu, and Weiping Wang. 2021. Sarcasm detection with commonsense knowledge. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3192–3201.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Chenwei Lou, Bin Liang, Lin Gui, Yulan He, Yixue Dang, and Ruifeng Xu. 2021. Affective dependency graph for sarcasm detection. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1844–1849.

Stephanie Lukin and Marilyn Walker. 2013. Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. In *Proceedings of the Workshop on Language Analysis in Social Media*, pages 30–40, Atlanta, Georgia. Association for Computational Linguistics.

Andrew John Merrison. 2008. Sarcasm and other mixed messages: The ambiguous way people use language, by patricia ann rockwell. *Journal of Politeness Research*, 4(2):331–334.

Shereen Oraby, Vrindavan Harrison, Lena Reed, Ernesto Hernandez, Ellen Riloff, and Marilyn A. Walker. 2016. Creating and characterizing a diverse corpus of sarcasm in dialogue. In *Proceedings of the SIGDIAL 2016 Conference, The 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 13-15 September 2016, Los Angeles, CA, USA*, pages 31–41.

Braja Gopal Patra, Soumadeep Mazumdar, Dipankar Das, Paolo Rosso, and Sivaji Bandyopadhyay. 2016. A multilevel approach to sentiment analysis of figurative language in twitter. In *Computational Linguistics and Intelligent Text Processing - 17th International Conference, CICLing 2016, Konya, Turkey, April 3-9, 2016, Revised Selected Papers, Part II*, volume 9624 of *Lecture Notes in Computer Science*, pages 281–291. Springer.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Soujanya Poria, Erik Cambria, Devamanyu Hazarika, and Prateek Vij. 2016. A deeper look into sarcastic tweets using deep convolutional neural networks. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 1601–1612.

Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on czech and english twitter. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 213–223.

Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 704–714.

Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 129–136. Omnipress.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly D. Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Comput. Linguistics*, 37(2):267–307.

Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Jian Su. 2018. Reasoning with sarcasm by reading in-between. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 1010–1020.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 606–615.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT/EMNLP 2005, Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 6-8 October 2005, Vancouver, British Columbia, Canada*, pages 347–354.

Chuhan Wu, Fangzhao Wu, Sixing Wu, Junxin Liu, Zhigang Yuan, and Yongfeng Huang. 2018. Thu_ngn at semeval-2018 task 3: Tweet irony detection with densely connected lstm and multi-task learning. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 51–56.

Tao Xiong, Peiran Zhang, Hongbo Zhu, and Yihui Yang. 2019. Sarcasm detection with self-matching networks and low-rank bilinear pooling. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2115–2124.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Tweet sarcasm detection using deep neural network. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2449–2460. ACL.