On the Importance of Word and Sentence Representation Learning in Implicit Discourse Relation Classification

Xin Liu[1*], **Jiefu Ou**[1], **Yangqiu Song**[1] and **Xin Jiang**[2]
[1]Department of CSE, the Hong Kong University of Science and Technology
[2]Huawei Noah's Ark Lab
xliucr@cse.ust.hk, jouaa@connect.ust.hk, yqsong@cse.ust.hk, jiang.xin@huawei.com

Abstract

Implicit discourse relation classification is one of the most difficult parts in shallow discourse parsing as the relation prediction without explicit connectives requires the language understanding at both the text span level and the sentence level. Previous studies mainly focus on the interactions between two arguments. We argue that a powerful contextualized representation module, a bilateral multi-perspective matching module, and a global information fusion module are all important to implicit discourse analysis. We propose a novel model to combine these modules together. Extensive experiments show that our proposed model outperforms BERT and other state-of-the-art systems on the PDTB dataset by around 8% and CoNLL 2016 datasets around 16%. We also analyze the effectiveness of different modules in the implicit discourse relation classification task and demonstrate how different levels of representation learning can affect the results.

1 Introduction

Discourse parsing is a fundamental task in natural language processing, which is considered to be a crucial step for many downstream tasks, such as machine translation [Li *et al.*, 2014], question answering [Jansen *et al.*, 2014], text generation [Bosselut *et al.*, 2018], and text summarization [Cohan *et al.*, 2018]. Shallow discourse parsing, such as Penn Discourse Treebank 2.0 (PDTB 2.0) [Prasad *et al.*, 2008] and CoNLL shared task [Xue *et al.*, 2016], focuses on identifying relations between two text spans (i.e., clauses or sentences) that are named discourse units. Among different sub-tasks in PDTB and CoNLL shared task, implicit relation classification between two sentences, with the absence of discourse markers or connectives in between (e.g., however, because), is considered as the most difficult problem.

Machine learning has been widely used in implicit discourse relation classification. Traditional research mainly focuses on linguistic features generation and selection at the text span level and sentence level, and the chosen classifiers are often linear classifiers [Pitler *et al.*, 2009; Lin *et al.*, 2009]. Recently, more flexible deep neural architectures are proposed to learn better representations. A considerable amount of effort has also been put for finding a good neural network architecture for implicit relation classification [Zhang *et al.*, 2015; Liu *et al.*, 2016; Qin *et al.*, 2017; Bai and Zhao, 2018; Dai and Huang, 2018] or some better learning settings such as multi-task learning [Liu *et al.*, 2016; Lan *et al.*, 2017; Varia *et al.*, 2019]. More recently, contextualized word representation learning, such as ELMo [Peters *et al.*, 2018], BERT [Devlin *et al.*, 2019], and RoBERTa [Liu *et al.*, 2019], has been proven to significantly improve many downstream tasks. It has been shown that BERT, especially the *next sentence prediction* (NSP) task, can significantly improve the implicit discourse relation classification task [Shi and Demberg, 2019]. However, in RoBERTa, while using a larger amount of data training and achieving better performance on many different tasks, it is claimed that NSP is not used. Nonetheless, we find RoBERTa cannot surpass BERT in this task without great modifications. Thus, it is unclear how to choose a contextualized encoder and what leads to improvement.

In this work, we show that different levels of representation learning are all important to this difficult implicit relation classification problem: deep contextualized word representations associated with an additional sentence separator can improve argument representations by better integrating context information; a bilateral multi-perspective matching can learn the interaction between two arguments without the heavy burden of feature engineering; a fusion module combining the multi-head attention and the gate mechanism can directly collect important clues to understand the text in depth. To this end, we propose a novel model, BMGF-RoBERTa (**B**ilateral **M**atching and **G**ated **F**usion with **RoBERTa**), to combine these modules together to address the discourse relation recognition with the absence of explicit connectives. Extensive experiments show that our proposed model outperforms BERT and other state-of-the-art systems on the standard PDTB dataset around 8% and CoNLL datasets around 16%. We also systematically analyze the effectiveness of different modules and demonstrate how different levels of semantic representation learning contribute to the discourse relation recognition problem. The source code is available at:

[*] This work was done when Xin Liu was an intern at Huawei Noah's Ark Lab.

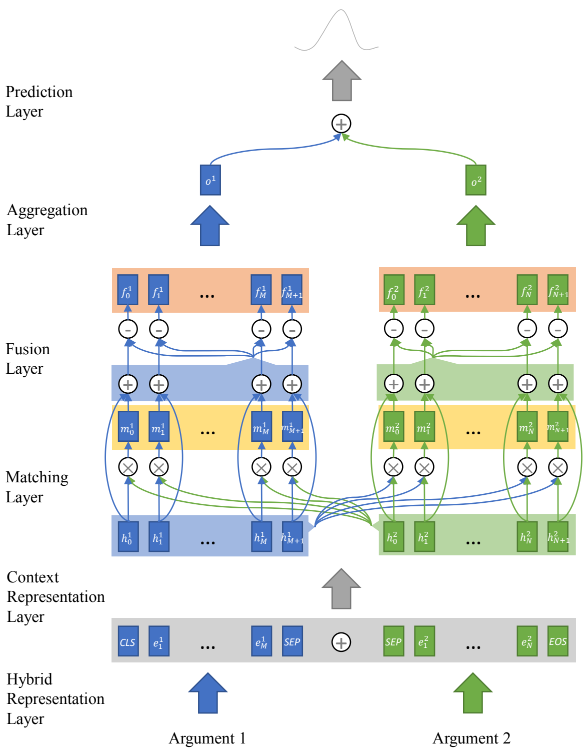
Prediction Layer
o^1
o^2
Aggregation Layer
f_0^1 f_1^1 ... f_M^1 f_{M+1}^1
f_0^2 f_1^2 ... f_N^2 f_{N+1}^2
Fusion Layer
m_0^1 m_1^1 ... m_M^1 m_{M+1}^1
m_0^2 m_1^2 ... m_N^2 m_{N+1}^2
Matching Layer
h_0^1 h_1^1 ... h_M^1 h_{M+1}^1
h_0^2 h_1^2 ... h_N^2 h_{N+1}^2
Context Representation Layer
CLS e_1^1 ... e_M^1 SEP
SEP e_1^2 ... e_N^2 EOS
Hybrid Representation Layer
Argument 1
Argument 2

Figure 1: The Architecture Overview.

https://github.com/HKUST-KnowComp/BMGF-RoBERTa.

2 Related Work

After the release of Penn Discourse Treebank 2.0 (PDTB 2.0) [Prasad *et al.*, 2008], the task of shallow discourse parsing has received much attention. Despite the great progress in classifying explicit discourse relations with F1 scores higher than 93% on four-class classification [Pitler *et al.*, 2008] and 90% on 15-class classification [Wang and Lan, 2015], implicit discourse relation recognition still remains a challenge. Most approaches regard the recognition problem as a classification problem of pairs of arguments. Previous research mainly focuses on feature engineering for linear classifiers to classify implicit discourse relations. The first work [Pitler *et al.*, 2009] selected several surface features to train four binary classifiers for the four top-level classes on PDTB. Results showed word pairs were most important among all features developed. [Lin *et al.*, 2009] was the first work to consider the classification on second level types. They further employed four different feature types to represent the context, the constituent parse trees, the dependency parse trees, and the raw text. [Rutherford and Xue, 2014] improved the baseline performance by as much as 25% with the help of brown cluster pairs and coreference patterns.

Recently, with the development of deep learning, more and more researchers choose to design neural architectures for this task as well as other NLP tasks. [Ji and Eisenstein, 2015] adopted two recursive neural networks to exploit the representation of arguments and entity spans. [Liu and Li, 2016] introduced the multi-level attention to mimic the repeated reading strategy and extract efficient features dynamically. [Qin *et al.*, 2016] found stacking gated convolutional neural networks can further improve the performance, compared with MLP, LSTM, and highway networks. [Bai and Zhao, 2018] proposed a model with different grained text representations, including character-, subword-, word-, sentence-, and sentence pair- levels, to enhance the representation power. [Dai and Huang, 2018] took the paragraph as a whole to and modeled inter-dependencies of discourse units by a conditional random filed layer. [Varia *et al.*, 2019] introduced word/n-gram pair convolutions with a gating mechanism to capture the interaction between the arguments for either explicit or implicit relation classification. [Shi and Demberg, 2019] found that the next sentence prediction played a role in finding relationships between sentences and benefited to the implicit discourse relation classification task. Giant contextualized representation models with pretraining show their power in many natural language understanding tasks even those low-resource tasks [Peters *et al.*, 2018; Devlin *et al.*, 2019; Liu *et al.*, 2019]. We can also easily find the trend to work with them from the latest research work in implicit relation classification, e.g., ELMo [Dai and Huang, 2019] and BERT [Shi and Demberg, 2019].

Some researchers chose different routines by expanding training data or integrating human-selected knowledge. [Liu *et al.*, 2016; Lan *et al.*, 2017] shared word embeddings with explicit relation classification. [Xu *et al.*, 2018] used active learning to sample explicitly-related arguments and expanded informative and reliable instances. External event knowledge and coreference relations in the paragraph-level were incorporated into discourse parsing by [Dai and Huang, 2019].

3 Model

In this section, we introduce the details of our model.

3.1 Model Overview

Figure 1 shows the architecture of our model. Our model consists of six parts, including (1) a hybrid representation layer that maps each word to a hybrid between character- and word-level embedding, (2) a contextualized representation layer that enhances the representation power of embeddings, (3) a matching layer that compares each token of one argument against all tokens of the other one and vice versa, (4) a fusion layer that assigns different importance to each word in arguments based on arguments themselves as well as matching results, (5) an aggregation layer that aggregates fusion results and encodes each argument into a vector, (6) a prediction layer that evaluates the probability distribution P$(y|Arg_1, Arg_2)$. Finally, we name our model as BMGF-RoBERTa (**B**ilateral **M**atching and **G**ated **F**usion with **RoBERTa**).

3.2 Contextualized Argument-aware Representation

Contextualized word representations have helped many models improve their performance in different natural language processing tasks. Some encoders, e.g., ELMO [Peters *et al.*, 2018], BERT [Devlin *et al.*, 2019], and RoBERTa [Liu *et al.*, 2019], have been widely applied. Here, we use RoBERTa since it is reported to have the best performance on various tasks.

We first use a byte-level Byte-Pair Encoding (BPE) in RoBERTa to encode arguments and map encoded results into hybrid representations:

$$\begin{aligned}
Arg_1 &: [\boldsymbol{CLS}, \boldsymbol{e}_1^1, \boldsymbol{e}_2^1, \cdots, \boldsymbol{e}_M^1, \boldsymbol{EOS}] \\
Arg_2 &: [\boldsymbol{CLS}, \boldsymbol{e}_1^2, \boldsymbol{e}_2^2, \cdots, \boldsymbol{e}_N^2, \boldsymbol{EOS}],
\end{aligned} \tag{1}$$

where $\boldsymbol{CLS}$ and $\boldsymbol{EOS}$ are two special token embeddings introduced in BERT and RoBERTa for attention and downstream tasks. M is the length of Arg_1 and N is the length of Arg_2, and $\boldsymbol{e}_i^j$ is the hybrid between character- and word-level embedding of the j^{th} word in Arg_i. Then the embedding lengths of Arg_1 and Arg_2 respectively are $M+2$ and $N+2$.

Then we concatenate the token representations as follows:

$$\begin{aligned}
&[\boldsymbol{e}_0, \boldsymbol{e}_1, \cdots, \boldsymbol{e}_M, \boldsymbol{e}_{M+1}, \boldsymbol{e}_{M+2}, \boldsymbol{e}_{M+3}, \cdots, \boldsymbol{e}_{M+N+2}, \boldsymbol{e}_{M+N+3}] \\
&= [\boldsymbol{CLS}, \boldsymbol{e}_1^1, \cdots, \boldsymbol{e}_M^1, \boldsymbol{SEP}, \boldsymbol{SEP}, \boldsymbol{e}_1^2, \cdots, \boldsymbol{e}_N^2, \boldsymbol{EOS}],
\end{aligned} \tag{2}$$

where $\boldsymbol{SEP}$ is a special token embedding to indicate boundaries of the sentence concatenation. We use two $\boldsymbol{SEP}$s here since we will further split these embeddings into two arguments and perform matching and fusion shown in the following sections.

Position embeddings are necessary for transformer-based models as shown in BERT and RoBERTa because the order and the distance between two words affect their dependence. Moreover, RoBERTa does not use segment embeddings to show the order of two sentences after removing the *next sentence prediction* (NSP) loss. However, as shown in [Shi and Demberg, 2019], the NSP task is very helpful. Therefore, we add back the trainable segment embeddings as what NSP does in addition to the hybrid embeddings and the position embeddings.

After multiple transformer layers, we get the contextualized representations $[\boldsymbol{h}_0, \boldsymbol{h}_1, \cdots, \boldsymbol{h}_{M+N+2}, \boldsymbol{h}_{M+N+3}]$. Then we split the representations for two arguments:

$$\begin{aligned}
[\boldsymbol{h}_0^1, \boldsymbol{h}_1^1, \cdots, \boldsymbol{h}_{M+1}^1] &= [\boldsymbol{h}_0, \boldsymbol{h}_1, \cdots, \boldsymbol{h}_{M+1}] \\
[\boldsymbol{h}_0^2, \boldsymbol{h}_1^2, \cdots, \boldsymbol{h}_{N+1}^2] &= [\boldsymbol{h}_{M+2}, \boldsymbol{h}_{M+3}, \cdots, \boldsymbol{h}_{M+N+3}].
\end{aligned} \tag{3}$$

3.3 Bilateral Multi-Perspective Matching

The resulting contextualized representations are sent to the matching module. Those complex discourse relations usually cannot be easily derived from surface features from argument pairs so that a multiple-perspective matching is required. We follow [Wang *et al.*, 2017] to utilize full-matching, maxpooling-matching, attentive-matching, and max-attentive-matching but reduce parameters by weight sharing.

We first define a multi-perspective cosine similarity:

$$\begin{aligned}
\boldsymbol{c} =& \text{MultiCos}(\boldsymbol{v}^1, \boldsymbol{v}^2; \boldsymbol{W}^c) \\
=& [\cos(\boldsymbol{W}_{1,:}^c \circ \boldsymbol{v}^1, \boldsymbol{W}_{1,:}^c \circ \boldsymbol{v}^2), \cdots, \cos(\boldsymbol{W}_{l,:}^c \circ \boldsymbol{v}^1, \boldsymbol{W}_{l,:}^c \circ \boldsymbol{v}^2)],
\end{aligned} \tag{4}$$

where $\boldsymbol{c} \in \mathbb{R}^l$ is the similarity vector, $\boldsymbol{W}^c \in \mathbb{R}^{l \times d}$ is a trainable matrix, $\boldsymbol{v}^1, \boldsymbol{v}^2 \in \mathbb{R}^d$, and d is the dimension of $\boldsymbol{v}^1$, $\boldsymbol{v}^2$ and l is the number of perspectives.

And then, we compute four matching results based on the above multi-perspective cosine similarity.

Full-Matching

We compute the full matching vector for each token in both arguments as follows:

$$\begin{aligned}
\hat{\boldsymbol{m}}_i^1 &= [\text{MultiCos}(\boldsymbol{h}_i^1, \boldsymbol{h}_0^2; \hat{\boldsymbol{W}}^f), \text{MultiCos}(\boldsymbol{h}_i^1, \boldsymbol{h}_{N+1}^2; \hat{\boldsymbol{W}}^l)] \\
\hat{\boldsymbol{m}}_j^2 &= [\text{MultiCos}(\boldsymbol{h}_0^1, \boldsymbol{h}_j^2; \hat{\boldsymbol{W}}^f), \text{MultiCos}(\boldsymbol{h}_{M+1}^1, \boldsymbol{h}_j^2; \hat{\boldsymbol{W}}^l)],
\end{aligned} \tag{5}$$

where $\hat{\boldsymbol{W}}^f, \hat{\boldsymbol{W}}^l \in \mathbb{R}^{l \times d}$ are trainable matrices for the first token and the last token respectively, $\hat{\boldsymbol{m}}_i^1 \in \mathbb{R}^{2l}$ is the full-matched vector for i^{th} word in Arg_1 and $\hat{\boldsymbol{m}}_j^2 \in \mathbb{R}^{2l}$ is for j^{th} word in Arg_2. As we add special tokens for each argument (*CLS* and *SEP* for Arg_1, *SEP* and *EOS* for Arg_2) and these special tokens contain much semantic information of the whole argument, we use these special tokens to consider the full matching in additional to the individual tokens.

Maxpooling-Matching

The maxpololing-matching is similar to the full-matching but more focused on the most important part. We need to compute all matching results and select the maximum elements.

$$\begin{aligned}
\check{\boldsymbol{m}}_i^1 &= \max_{0 \leq j \leq N+1} \text{MultiCos}(\boldsymbol{h}_i^1, \boldsymbol{h}_j^2; \check{\boldsymbol{W}}) \\
\check{\boldsymbol{m}}_j^2 &= \max_{0 \leq i \leq M+1} \text{MultiCos}(\boldsymbol{h}_i^1, \boldsymbol{h}_j^2; \check{\boldsymbol{W}}),
\end{aligned} \tag{6}$$

where $\check{\boldsymbol{W}} \in \mathbb{R}^{l \times d}$ is a trainable matrix for the maxpooling-matching and $\max$ is the element-wise maximum.

Attentive-Matching

The attentive-matching uses the idea of the weighted mean. We need to compute weights based on the cosine similarity and compute the weighted average vectors:

$$\begin{aligned}
\bar{\boldsymbol{h}}^1 &= \frac{\sum_{0 \leq i \leq M+1} c_{i,j} \cdot \boldsymbol{h}_i^1}{\sum_{0 \leq i \leq M+1} c_{i,j}} \\
\bar{\boldsymbol{h}}^2 &= \frac{\sum_{0 \leq j \leq N+1} c_{i,j} \cdot \boldsymbol{h}_j^2}{\sum_{0 \leq j \leq N+1} c_{i,j}},
\end{aligned} \tag{7}$$

where $c_{i,j} = \cos(\boldsymbol{h}_i^1, \boldsymbol{h}_j^2)$ is the cosine similarity of $\boldsymbol{h}_i^1$ and $\boldsymbol{h}_j^2$. Based on the weighted average representations of Arg_1 and Arg_2, we can compute the corresponding attentive vectors:

$$\begin{aligned}
\bar{\boldsymbol{m}}_i^1 &= \text{MultiCos}(\boldsymbol{h}_i^1, \bar{\boldsymbol{h}}^2; \bar{\boldsymbol{W}}) \\
\bar{\boldsymbol{m}}_j^2 &= \text{MultiCos}(\boldsymbol{h}_i^2, \bar{\boldsymbol{h}}^1; \bar{\boldsymbol{W}}),
\end{aligned} \tag{8}$$

where $\bar{\boldsymbol{W}} \in \mathbb{R}^{l \times d}$ is a trainable matrix for the attentive-matching.

Max-Attentive-Matching

Instead of taking the weighted average of all the contextual embeddings as the attentive vector, the max-attentive-matching picks the contextual embedding with the highest cosine similarity as the attentive vector:

$$\begin{aligned}
\tilde{\boldsymbol{m}}_i^1 &= \text{MultiCos}(\boldsymbol{h}_i^1, \tilde{\boldsymbol{h}}^2; \tilde{\boldsymbol{W}}) \\
\tilde{\boldsymbol{m}}_j^2 &= \text{MultiCos}(\boldsymbol{h}_i^2, \tilde{\boldsymbol{h}}^1; \tilde{\boldsymbol{W}}),
\end{aligned} \tag{9}$$

where $\tilde{\boldsymbol{W}} \in \mathbb{R}^{l\times d}$ is a trainable matrix for the max-attentive-matching, $\tilde{\boldsymbol{h}}^1$ corresponds to the contextualized representation $\boldsymbol{h}_i^1$ with $\arg\max_{0\le i\le M+1} c_{i,j}$, and $\tilde{\boldsymbol{h}}^2$ corresponds to that of $\boldsymbol{h}_j^2$ with $\arg\max_{0\le j\le N+1} c_{i,j}$.

Finally, the bilateral multi-perspective matching results can be obtained by concatenating the four matching results:

$$\begin{aligned}\boldsymbol{m}_i^1 &= [\hat{\boldsymbol{m}}_i^1, \check{\boldsymbol{m}}_i^1, \bar{\boldsymbol{m}}_i^1, \tilde{\boldsymbol{m}}_i^1]\\ \boldsymbol{m}_j^2 &= [\hat{\boldsymbol{m}}_j^2, \check{\boldsymbol{m}}_j^2, \bar{\boldsymbol{m}}_j^2, \tilde{\boldsymbol{m}}_j^2],\end{aligned} \tag{10}$$

where $\boldsymbol{m}_i^1, \boldsymbol{m}_j^2 \in \mathbb{R}^{5l}$.

3.4 Gated Information Fusion

The matching results contain the interaction between two arguments but miss the interaction inside each argument. The most direct method is to apply self-attention on the concatenation of $\boldsymbol{h}$ and $\boldsymbol{m}$. Multi-head attention with layer normalization has been used in the Transformer architecture [Vaswani *et al.*, 2017]. We modify it as a gated multi-head attention without layer normalization to remain the scale of each dimension and retain the non-linear capability:

$$\boldsymbol{Q}' = \text{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) \tag{11}$$

$$a = \text{sigmoid}({\boldsymbol{W}^a}^\top [\boldsymbol{Q}, \boldsymbol{Q}']) \tag{12}$$

$$\text{GatedMultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = a \odot \boldsymbol{Q}' + (1-a) \odot \boldsymbol{Q}, \tag{13}$$

where $\boldsymbol{Q} \in \mathbb{R}^{d_q}, \boldsymbol{K} \in \mathbb{R}^{d_k}, \boldsymbol{V} \in \mathbb{R}^{d_v}, \boldsymbol{Q}' \in \mathbb{R}^{d_q}$ are the query, key, value, and output respectively, $\boldsymbol{W}^a \in \mathbb{R}^{2d_q\times 1}$ is trainable matrix to transform the concatenation of $\boldsymbol{Q}$ and $\boldsymbol{Q}'$ to a scalar value a. The final attention result is controlled by the gate a.

To fuse the contextualized representations $[\boldsymbol{h}_i^1, \cdots, \boldsymbol{h}_{M+1}^1]$ and matching vectors $[\boldsymbol{m}_i^1, \cdots, \boldsymbol{m}_{M+1}^1]$, we apply the gated multi-head attention over the concatenation results:

$$\boldsymbol{Q}_{i,:}^1 = \boldsymbol{K}_{i,:}^1 = \boldsymbol{V}_{i,:}^1 = [\boldsymbol{h}_i^1, \boldsymbol{m}_i^1] \tag{14}$$

$$[\boldsymbol{f}_1^1, \boldsymbol{f}_2^1, \cdots, \boldsymbol{f}_{M+1}^1] = \text{GatedMultiHead}(\boldsymbol{Q}^1, \boldsymbol{K}^1, \boldsymbol{V}^1). \tag{15}$$

We can also get $[\boldsymbol{f}_1^2, \cdots, \boldsymbol{f}_{N+1}^2]$ similarly.

3.5 Aggregation and Prediction

The aggregation layer aggregates the two sequences of fused vectors into a fixed-length matching vector. Inspired by [Kim *et al.*, 2016], we use a convolutional neural network followed by a highway network to capture unigram, bigram, ..., n-gram information. Assuming we apply z convolutional operations Conv$_1$, Conv$_2$, ..., Conv$_z$ followed by max pooling operations to $[\boldsymbol{f}_i^1, \cdots, \boldsymbol{f}_{M+1}^1]$, then the output of each conv-pool operation applied on arguments is

$$\boldsymbol{u}_c^1 = \max_{0\le i\le M+1-c} \text{ReLU}(\text{Conv}_c([\boldsymbol{f}_i^1, \cdots, \boldsymbol{f}_{i+c}^1])), \tag{16}$$

where the kernel size of Conv$_c$ is c, the stride is 1, and the dimension of $\boldsymbol{u}_c^1$ depends on the number of filters (assume we use s filters).

The flatten concatenation of all conv-pool results are sent as the input of a highway network to extract more features:

Relation	training	validation	test
Comp.	1,942	197	152
Cont.	3,342	295	279
Exp.	7,004	671	574
Temp.	760	64	85
Total	12,362	1,183	1,046

Table 1: Statistics of four top level implicit senses in PDTB 2.0.

$$\boldsymbol{u}^1 = \text{Flatten}([\boldsymbol{u}_1^1, \boldsymbol{u}_2^1, \cdots, \boldsymbol{u}_z^1]) \tag{17}$$

$$\boldsymbol{u}^{1'} = \text{ReLU}({\boldsymbol{W}^h}^\top \boldsymbol{u}^1) \tag{18}$$

$$g^1 = \text{sigmod}({\boldsymbol{W}^g}^\top \boldsymbol{u}^1) \tag{19}$$

$$\boldsymbol{o}^1 = g^1 \odot \boldsymbol{u}^{1'} + (1-g^1) \odot \boldsymbol{u}^1, \tag{20}$$

where $\boldsymbol{u}^1 \in \mathbb{R}^{zs}$ is a flatten vector, $\boldsymbol{W}^h \in \mathbb{R}^{zs\times zs}, \boldsymbol{W}^g \in \mathbb{R}^{zs\times 1}$ are trainable. As before, we can get $\boldsymbol{o}^2$ similarly.

Finally, the prediction layer evaluates the probability distribution Pr$(y|Arg_1, Arg_2)$. We use a two layer feed-forward neural network to consume $\boldsymbol{o}^1$ and $\boldsymbol{o}^2$, and apply the softmax function in the last layer. To approximate the data distribution, we optimize cross-entropy loss between the outputs of the prediction layer and the ground-truth labels.[1]

4 Experiments

In this section, we show our experimental results and detailed analysis of different representation learning modules.

4.1 Datasets

The Penn Discourse Treebank 2.0 (PDTB 2.0) PDTB 2.0 is a large scale corpus annotated with information related to discourse structure and discourse semantics, containing 2,312 Wall Street Journal (WSJ) articles. PDTB 2.0 has three levels of senses, i.e., classes, types, and sub-types. Most work focuses on the top level, including *Comparison* (*Comp.*), *Contingency* (*Cont.*), *Expansion* (*Exp.*), and *Temporal* (*Temp.*). We follow [Ji and Eisenstein, 2015] to split section 2-20, 0-1, and 21-22 as training, validation, and test sets respectively. We not only evaluate our model on the four top-level implicit senses but the 11 major second level implicit types. Table 1 shows the statistics of the top-level senses.

The CoNLL-2016 Shared Task (CoNLL) The CoNLL-2016 shared task [Xue *et al.*, 2016] provides more abundant annotation for shadow discourse parsing. The PDTB section 23 (CoNLL-Test) and newswire texts (CoNLL-Blind) following the PDTB annotation guidelines were organized as the test sets. CoNLL merges several labels. For example, *Contingency.Pragmatic cause* is merged into *Contingency.Cause.Reason* to remove the former type with very few samples. Finally, there is a flat list of 15 sense categories to be classified.

[1] Some instances in PDTB and CoNLL were annotated with more than one label. Following previous work and the scorer at https://github.com/attapol/conll16st, a prediction is regarded as correct once it matches one of the ground-truth labels.

Model	PDTB-4 (F1;Acc)		PDTB-11 (Acc)	CoNLL-Test (Acc)	CoNLL-Blind (Acc)
[Ji and Eisenstein, 2015]	-	-	44.59	-	-
[Liu *et al.*, 2016]	44.98	57.27	-	-	-
[Liu and Li, 2016][2]	46.29	57.57	-	-	-
[Rutherford and Xue, 2016]	-	-	-	40.91	34.20
[Lan *et al.*, 2017]	47.80	57.39	-	39.40	40.12
[Bai and Zhao, 2018]	51.06	-	48.22	-	-
[Xu *et al.*, 2018]	44.48	60.63	-	-	-
[Dai and Huang, 2018]	48.82	57.44	-	-	-
[Dai and Huang, 2019]	52.89	59.66	48.23	-	-
[Shi and Demberg, 2019][2]	-	-	53.23 (0.39)	-	-
[Varia *et al.*, 2019][3]	50.20	59.13	-	39.39	39.36
BMGF-RoBERTa-Siamese	53.70 (0.31)	62.27 (0.67)	50.37 (0.59)	47.42 (1.04)	46.24 (1.07)
BMGF-RoBERTa	**63.39 (0.56)**	**69.06 (0.43)**	**58.13 (0.67)**	**57.26 (0.75)**	**55.19 (0.55)**

Table 2: Performance of multi-class classification on PDTB and CoNLL-2016 in terms of accuracy (Acc) (%) and macro-averaged F1 (F1) (%) and corresponding standard deviation.

Model	Comp.	Cont.	Exp.	Temp.
[Ji and Eisenstein, 2015]	35.93	52.78	-	27.63
[Rutherford and Xue, 2015]	41.00	53.80	69.40	33.30
[Liu *et al.*, 2016]	37.91	55.88	69.97	37.17
[Liu and Li, 2016][2]	39.86	54.48	70.43	38.84
[Qin *et al.*, 2016]	38.67	54.91	71.50	32.76
[Lan *et al.*, 2017]	40.73	58.96	72.47	38.50
[Bai and Zhao, 2018]	47.85	54.47	70.60	36.87
[Dai and Huang, 2018]	46.79	57.09	70.41	45.61
[Varia *et al.*, 2019]	44.10	56.02	72.11	44.41
BMGF-RoBERTa-Siamese	44.19	56.36	72.90	39.65
BMGF-RoBERTa	**59.44**	**60.98**	**77.66**	**50.26**

Table 3: Performance of multiple binary classification on the top level classes in PDTB in terms of F1 (%).

4.2 Parameter Settings

We use the RoBERTa-base as our context representation layer. Because the data size is not large enough so that we freeze all parameters but add trainable segment embeddings like BERT. The number of perspectives of the matching layer l, the number of heads of the fusion layer h, the number of convolutional operations z, the number of filters s are set as 16, 16, 2, 64 respectively, and all hidden dimensions are set as 128. We apply dropout to every layer with the dropout rate 0.2, clip the gradient L2-norm with a threshold 2.0, and add the L2 regularization with coefficient 0.0005 to alleviate overfitting. We adopt Adam [Kingma and Ba, 2015] with an initial learning rate of 0.001 and a batch size of 32 to train models up to 50 epochs. All experiments are performed three times with 2×32 GB NVIDIA V100 GPUs and all reported results are averaged performance. Our model and variants are all implemented by PyTorch.

4.3 Experimental Results

We compare **BMGF-RoBERTa** with other recent state-of-the-art baselines on multi-class classification and multiple binary classification to demonstrate the importance of our representation, matching, and fusion modules. A siamese network (denoted as **BMGF-RoBERTa-Siamese**) whose encoder does not concatenate embeddings in two arguments but uses embeddings from RoBERTa separately obtained from each argument is also implemented for analysis.

Macro-averaged F1 is the main metric for the four-class classification, and accuracy is considered as the main metric for remaining multi-class classification problems. For binary classification, F1 is adopted to evaluate the performance on each class.

Multi-class Classification

We first evaluate our model on multi-class classification which is the final part of a shallow discourse parser as usual. Two different settings are set for PDTB in view of the hierarchical categories. One is the four-class classification (denoted as PDTB-4) at the top-level and the other is the 11-class classification (denoted as PDTB-11) at the second level. We conduct 15-class classification for CoNLL dataset, denoted as CoNLL-Test and CoNLL-Blind for the test set and blind set.

Results of our model and baselines are shown in Table 2, and BMGF-RoBERTa achieves the new state-of-the-art performance with substantial improvements on all implicit discourse classification settings. Most previous systems considered siamese architectures except [Shi and Demberg, 2019] which was the first to consider two arguments as a whole and fine-tune BERT based on the next sentence prediction loss for discourse relation classification. Our variant model BMGF-RoBERTa-Siamese can also beat other siamese models and narrow 40% of the gap with [Shi and Demberg, 2019]. Furthermore, BMGF-RoBERTa is the first model that exceeds the 60% macro-averaged F1 and approximates 70% accuracy in the four-class classification on PDTB.

Binary Classification

To compare previous results more directly and analyze model behaviour more comprehensively, we conduct experiments based on one-vs-rest binary classifiers on the four top-level classes in PDTB. Table 3 shows the results of binary classification on the top-level classes in PDTB. It is as expected that

[2] We select the best results from their different settings.

[3] We use the average results of their best single model.

Model	PDTB-4 (F1;Acc)		PDTB-11 (Acc)	CoNLL-Test (Acc)	CoNLL-Blind (Acc)
BMGF-RoBERTa	**63.39**	**69.06**	**58.13**	**57.26**	**55.19**
w/o SE	55.15	63.29	50.72	49.67	48.06
w/o GF	60.40	66.28	55.31	53.92	50.51
w/o BM	61.93	68.83	57.23	56.22	54.39
w/o SE,GF	53.36	62.24	50.14	48.07	46.56
w/o SE,BM	51.42	63.22	50.37	47.51	48.22
w/o BM,GF	59.32	66.89	55.89	55.22	52.18
w/o SE,BM,GF (RoBERTa)	50.97	62.11	49.76	47.33	46.16
RoBERTa w/ SourceAttn	54.42	62.24	49.79	47.85	46.16
RoBERTa w/ SourceAttn, SelfAttn	52.89	62.56	50.02	47.94	47.59

Table 4: Architecture ablation analysis on multi-class classification.

Model	Comp.	Cont.	Exp.	Temp.
BMGF-RoBERTa	**59.44**	**60.98**	**77.66**	**50.26**
BMGF-RoBERTa w/o SE	47.08	55.64	73.26	39.91
BMGF-RoBERTa-Siamese	44.19	56.36	72.90	39.65

Table 5: Encoder ablation analysis on multiple binary classification.

BMGF-RoBERTa has the best performance, but it is still surprising that the improvement on *Comp.* exceeds the previous best more than 11% and the improvement on *Temp.* exceeds the previous best around 5%. In contrast, the performance of the siamese variant looks a bit ordinary. Results of binary classification indicate that RoBERTa encoder can further capture the deeper discourse relations after giving the "previous-next" context relationship. Such hard discourse relations can not be handled well when two arguments are encoded separately. However, it is still unclear whether the self-attention during encoding or segment embeddings provides the most useful information. We discuss more in Section 4.4.

4.4 Ablation Study

We first analyze the effectiveness of the encoding module, the matching module, and the fusion module. To this end, we remove the segment embeddings (SE), the bilateral multi-perspective matching (BM), and the gated information fusion (GF) from BMGF-RoBERTa one by one. To compare the common matching and fusion modules, we implement another two ablation models based on the original RoBERTa: **RoBERTa w/ SourceAttn** is a model that uses the RoBERTa to encode the concatenation of two arguments and then do the source attention between two separated contextualized embeddings with the residual connection; **RoBERTa w/ SourceAttn, SelfAttn** adds the self-attention and the residual connection after the source attention.

Table 4 shows the results. We can see that eliminating any of the three modules would hurt the performance significantly. The original RoBERTa model cannot surpass the previous best siamese network on PDTB-4 too much or outperform BERT on PDTB-11, but the SE, BM, and GF can help it improve around 9.3% on average. At the same time, the segment embeddings improve mostly. When comparing source attention and self-attention with our bilateral multi-perspective matching and gated information fusion, we can also conclude the combination of BM and GF is much better than those vanilla multi-head attention mechanisms when BMGF-RoBERTa w/o SE as the reference. Moreover, the self-attention hurts the F1 score in PDTB-4.

BMGF-RoBERTa-Siamese shown in Table 2 is not comparable with BMGF-RoBERTa w/o SE and show in Table 4. Separately encoding cannot benefit from the longer self-attention after concatenation. And we can also find SE can improve much more in Table 4, indicating that the longer self-attention is not as straightforward as segment embeddings.

We further design the encoder ablation study for the encoder module and results are listed in Table 5. We can find that the main gaps between encoding without concatenating (BMGF-RoBERTa-Siamese) and encoding without segment embeddings (BMGF-RoBERTa w/o SE) come from the *Comp.* and *Temp.* classes. And adding SE brings a qualitative leap in all four classes because BMGF-RoBERTa finally captures clues of two arguments by multiple attention and relative segment order embeddings. That is, a more powerful representation layer can bring unexpected improvement and understand implicit relations better. But how to inject useful information to the representation module looks more important. In our paper, we inject the "previous-next" context relationship by the concatenation and segment embeddings for implicit discourse relation classification.

5 Conclusion

We present a novel model, BMGF-RoBERTa, that combines representation, matching, and fusion modules for implicit discourse relation classification. Experimental results show BMGF-RoBERTa outperforms all previous state-of-the-art systems with substantial margins on the PDTB dataset and CoNLL datasets. We find the "previous-next" context relationship provides much information for this difficult task. Furthermore, the effectiveness of these well-designed modules and what is important in the representation layer are thoroughly analyzed in the ablation study. One important future work would be to find more clues for better representations within and even beyond the word- and sentence-levels.

Acknowledgements

This paper was supported by the Early Career Scheme (ECS, No. 26206717) from the Research Grants Council in Hong Kong and the Gift Fund from Huawei Noah's Ark Lab.

References

[Bai and Zhao, 2018] Hongxiao Bai and Hai Zhao. Deep enhanced representation for implicit discourse relation recognition. In *COLING*, pages 571–583, 2018.

[Bosselut *et al.*, 2018] Antoine Bosselut, Asli Çelikyilmaz, Xiaodong He, Jianfeng Gao, Po-Sen Huang, and Yejin Choi. Discourse-aware neural rewards for coherent text generation. In *NAACL-HLT*, pages 173–184, 2018.

[Cohan *et al.*, 2018] Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. A discourse-aware attention model for abstractive summarization of long documents. In *NAACL-HLT*, pages 615–621, 2018.

[Dai and Huang, 2018] Zeyu Dai and Ruihong Huang. Improving implicit discourse relation classification by modeling interdependencies of discourse units in a paragraph. In *NAACL-HLT*, pages 141–151, 2018.

[Dai and Huang, 2019] Zeyu Dai and Ruihong Huang. A regularization approach for incorporating event knowledge and coreference relations into neural discourse parsing. In *EMNLP*, pages 2974–2985, 2019.

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

[Jansen *et al.*, 2014] Peter Jansen, Mihai Surdeanu, and Peter Clark. Discourse complements lexical semantics for non-factoid answer reranking. In *ACL*, pages 977–986, 2014.

[Ji and Eisenstein, 2015] Yangfeng Ji and Jacob Eisenstein. One vector is not enough: Entity-augmented distributed semantics for discourse relations. *TACL*, 3:329–344, 2015.

[Kim *et al.*, 2016] Yoon Kim, Yacine Jernite, David A. Sontag, and Alexander M. Rush. Character-aware neural language models. In *AAAI*, pages 2741–2749, 2016.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[Lan *et al.*, 2017] Man Lan, Jianxiang Wang, Yuanbin Wu, Zheng-Yu Niu, and Haifeng Wang. Multi-task attention-based neural networks for implicit discourse relationship representation and identification. In *EMNLP*, pages 1299–1308, 2017.

[Li *et al.*, 2014] Junyi Jessy Li, Marine Carpuat, and Ani Nenkova. Assessing the discourse factors that influence the quality of machine translation. In *ACL*, pages 283–288, 2014.

[Lin *et al.*, 2009] Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. Recognizing implicit discourse relations in the penn discourse treebank. In *EMNLP*, pages 343–351, 2009.

[Liu and Li, 2016] Yang Liu and Sujian Li. Recognizing implicit discourse relations via repeated reading: Neural networks with multi-level attention. In *EMNLP*, pages 1224–1233, 2016.

[Liu *et al.*, 2016] Yang Liu, Sujian Li, Xiaodong Zhang, and Zhifang Sui. Implicit discourse relation classification via multi-task neural networks. In *AAAI*, pages 2750–2756, 2016.

[Liu *et al.*, 2019] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[Peters *et al.*, 2018] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *NAACL-HLT*, pages 2227–2237, 2018.

[Pitler *et al.*, 2008] Emily Pitler, Mridhula Raghupathy, Hena Mehta, Ani Nenkova, Alan Lee, and Aravind K. Joshi. Easily identifiable discourse relations. In *COLING*, pages 87–90, 2008.

[Pitler *et al.*, 2009] Emily Pitler, Annie Louis, and Ani Nenkova. Automatic sense prediction for implicit discourse relations in text. In *ACL*, pages 683–691, 2009.

[Prasad *et al.*, 2008] Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K. Joshi, and Bonnie L. Webber. The penn discourse treebank 2.0. In *LREC*, 2008.

[Qin *et al.*, 2016] Lianhui Qin, Zhisong Zhang, and Hai Zhao. A stacking gated neural architecture for implicit discourse relation classification. In *EMNLP*, pages 2263–2270, 2016.

[Qin *et al.*, 2017] Lianhui Qin, Zhisong Zhang, Hai Zhao, Zhiting Hu, and Eric P. Xing. Adversarial connective-exploiting networks for implicit discourse relation classification. In *ACL*, pages 1006–1017, 2017.

[Rutherford and Xue, 2014] Attapol Rutherford and Nianwen Xue. Discovering implicit discourse relations through brown cluster pair representation and coreference patterns. In *EACL*, pages 645–654, 2014.

[Rutherford and Xue, 2015] Attapol Rutherford and Nianwen Xue. Improving the inference of implicit discourse relations via classifying explicit discourse connectives. In *NAACL-HLT*, pages 799–808, 2015.

[Rutherford and Xue, 2016] Attapol Rutherford and Nianwen Xue. Robust non-explicit neural discourse parser in english and chinese. In *CoNLL*, pages 55–59, 2016.

[Shi and Demberg, 2019] Wei Shi and Vera Demberg. Next sentence prediction helps implicit discourse relation classification within and across domains. In *EMNLP-IJCNLP*, pages 5789–5795, 2019.

[Varia *et al.*, 2019] Siddharth Varia, Christopher Hidey, and Tuhin Chakrabarty. Discourse relation prediction: Revisiting word pairs with convolutional networks. In *SIGDIAL*, pages 442–452, 2019.

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

[Wang and Lan, 2015] Jianxiang Wang and Man Lan. A refined end-to-end discourse parser. In *CoNLL*, pages 17–24, 2015.

[Wang *et al.*, 2017] Zhiguo Wang, Wael Hamza, and Radu Florian. Bilateral multi-perspective matching for natural language sentences. In *IJCAI*, pages 4144–4150, 2017.

[Xu *et al.*, 2018] Yang Xu, Yu Hong, Huibin Ruan, Jianmin Yao, Min Zhang, and Guodong Zhou. Using active learning to expand training data for implicit discourse relation recognition. In *EMNLP*, pages 725–731, 2018.

[Xue *et al.*, 2016] Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Attapol Rutherford, Bonnie L. Webber, Chuan Wang, and Hongmin Wang. Conll 2016 shared task on multilingual shallow discourse parsing. In *CoNL*, pages 1–19, 2016.

[Zhang *et al.*, 2015] Biao Zhang, Jinsong Su, Deyi Xiong, Yaojie Lu, Hong Duan, and Junfeng Yao. Shallow convolutional neural network for implicit discourse relation recognition. In *EMNLP*, pages 2230–2235, 2015.