# Implicit Discourse Relation Classification with Syntax-Aware Contextualized Word Representations

Diana Nicoleta Popa, Julien Perez, James Henderson, Éric Gaussier

The Thirty-Second International Florida
Artificial Intelligence Research Society Conference (FLAIRS-32)

# Implicit Discourse Relation Classification with Syntax-Aware Contextualized Word Representations

**Diana Nicoleta Popa,**[12] **Julien Perez,**[2] **James Henderson,**[3] **Eric Gaussier**[1]
[1]Université Grenoble Alpes, LIG, Grenoble, France
[2] Naver Labs Europe, Meylan, France, [3] Idiap Research Institute, Martigny, Switzerland
diana.popa@imag.fr, julien.perez@naverlabs.com, james.henderson@idiap.ch, eric.gaussier@imag.fr

## Abstract

Automatically identifying implicit discourse relations requires an in-depth semantic understanding of the text fragments involved in such relations. While early work investigated the usefulness of different classes of input features, current state-of-the-art models mostly rely on standard pre-trained word embeddings to model the arguments of a discourse relation. In this paper, we introduce a method to compute contextualized representations of words, leveraging information from the sentence dependency parse, to improve argument representation. The resulting token embeddings encode the structure of the sentence from a dependency point of view in their representations. Experimental results show that the proposed representations achieve state-of-the-art results when input to standard neural network architectures, surpassing complex models that use additional data and consider the interaction between arguments.

## Introduction

Automatically identifying discourse relations is helpful for downstream NLP tasks such as question answering, machine translation or automatic summarization. Much research focused on the task along with the release of the Penn Discourse Treebank (PDTB) (Prasad et al. 2008), the largest annotated corpus of discourse relations. In PDTB, documents are annotated following the predicate-argument structure: a discourse connective (e.g. *but, because*) is a predicate that takes two text spans around it as its arguments, further denoted as *Arg*-1 and *Arg*-2.

To approach the task of implicit discourse relation classification (IDRC), the focus of most work has been on modelling the interaction between arguments and less on their representations. Although earlier work uses different feature sets as input: word pairs, part-of-speech tags, context information etc. (Pitler, Louis, and Nenkova 2009), little attention has been offered to varying the input features of recent deep neural network-based approaches to IDRC and to how these can influence the output quality of such models. That is, most current approaches rely on standard pre-trained word type embeddings, which have been successful across a variety of tasks and have been proven to perform best so far on IDRC as well (Braud and Denis 2015).

To account for the difficulty to fully recover the semantics of arguments using only surface level features (Ji and Eisenstein 2015), additional linguistic and structural information regarding the arguments can be leveraged (Dai and Huang 2018; Qin, Zhang, and Zhao 2016a). However, information from syntactic dependencies, previously proven beneficial for the task (Lin, Kan, and Ng 2009), is not integrated in any of these models. The current work aims precisely at investigating the use of syntactic information in such a context.

On the other hand, there has been a strong recent interest to replace the generic word type embeddings by *token embeddings* which have proven to be successful within various applications (McCann et al. 2017; Peters et al. 2018). The idea of token embeddings is to represent a word in its context, with the same word bearing different representations in different contexts. This contrasts to a generic word embedding representation, which is the same in every context.

Since the task of detecting implicit discourse relations requires semantic understanding which consequently relies on encoding the word meaning in its context (Qin, Zhang, and Zhao 2016a), it is natural to investigate the use of token representations for this task. For this we propose a set of token representations that offer improvements over traditional word representations and have the benefit of encoding the information about the structure of the sentence from a dependency point of view in the representations themselves. To the best of our knowledge we are the first to investigate the use of token embeddings for the task of IDRC and to analyze the impact of using syntactic dependencies information as input to deep learning models for this task.

We evaluate the proposed token embeddings as input to two most common basic neural network architectures and an additional gated mechanism to model the interaction between the arguments. Our main contributions are:

- We propose a method to explicitly integrate syntactic information into token embeddings to model the arguments for IDRC;
- We analyze and compare the contribution of token embeddings to the task as opposed to word type representations;

- We reach state-of-the-art performance with simple architectures, surpassing complex models that focus on the interaction between arguments or use additional data as well as models that use other token embeddings.

The remaining of this paper is organised as follows: We first present the related work, then we introduce our token embeddings proposal, the architectures used to model the arguments in a discourse relation and the methods for discourse relation identification. We then provide details about the data used and the implementation. Finally we present the results along with an analysis of the proposed contribution.

## Related work

### Implicit discourse relation classification

The task of IDRC is typically approached as a classification problem, with the two arguments as input and their implicit discourse relation as the label to predict. Much work leverages both labeled and unlabeled data (implicit and explicit) from different corpora in order to classify discourse relations in multi-task learning frameworks (Liu et al. 2016). Another tendency is to focus on the interaction between the two arguments either through attention mechanisms (Lan et al. 2017), by using features derived from word pairs (Chen et al. 2016) or by modelling the argument pair jointly (Liu et al. 2016).

Regardless of the chosen approach, accurately representing the arguments is key to building a reliable model. For this, most work relies on standard word embeddings while some employs complementary features to integrate additional knowledge: (Ji and Eisenstein 2015) represent each argument using bottom-up compositional operations over its constituency parse tree, while other work complements word embeddings with extra linguistic features like part-of-speech tag embeddings (Dai and Huang 2018) or character-level information (Qin, Zhang, and Zhao 2016a). (Braud and Denis 2016) learn distributional word representations tailored specifically for IDRC. We propose to improve the representation of words by using *token embeddings* computed using dependency information. Dependency information has been shown to be beneficial for detecting implicit discourse relations in traditional models (Lin, Kan, and Ng 2009). However, this information is not used in more recent neural architectures. We separate the token embeddings computation from the task at hand which enables assessing the benefits of using them in comparison to standard word embeddings.

### Token embeddings methods

There has been a growing interest recently in representing text using token embeddings. (Dasigi et al. 2017) propose token embeddings by estimating a distribution over semantic concepts (synsets) extracted from WordNet. (Tu, Gimpel, and Livescu 2017) use a feed forward neural network to produce token embeddings which are further evaluated as features for part-of-speech taggers and dependency parsers.

(McCann et al. 2017) provide context-aware vectors (CoVe) by transferring a pre-trained deep LSTM encoder from a model trained for machine translation to a variety of other NLP tasks. Recent work leverages information from language models in semi-supervised settings: (Peters et al. 2017) use the parameters from a pre-trained language model to induce contextual information to token representations. (Peters et al. 2018) extend the idea by learning a task specific linear combination of the intermediate layers of a deeper bidirectional language model (ELMo). However, their representation is somewhat task dependent in that the linear combination is learned with respect to the task at hand. Moreover, training these models requires large amounts of data.

A parallel could be drawn between the current proposal and the work of (Salant and Berant 2018) who show that adding contextualized representations to a basic model for question-answering achieves state-of-the-art results, despite using only minimal question-document interaction. This validates further the importance of having contextually-informed features as input to deep learning models, even when these models are not the most complex.

## Main approach

We propose to approach the task of IDRC through a two-step process: unsupervised computation of syntactically-aware contextualized representations of words and a supervised model for the prediction of discourse relations. The proposed token embeddings will constitute an informed and complete encoding as they are trained to predict the relations holding between them in the sentence graph.

### Computing the token embeddings

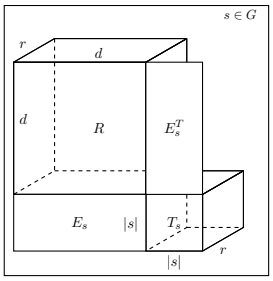
We compute unsupervised token embeddings for all the words in the corpora. For this we leverage dependency relations obtained from a dependency parser (Honnibal and Johnson 2015), using the CLEAR labels as implemented in the spaCy toolkit. We additionally use information regarding immediate local context in the form of adjacency relations.

Formally, given the graph of a sentence $G_s$ as provided by its dependency parse tree, we model the interactions between the tokens in the sentence as shown in Figure 1. We use a rank-3 tensor $T_s$ to specify the binary relations between tokens that are given by the parse tree of the sentence along with an additional adjacency relation.

We optimise a ranking loss within the tensor ($T^s_{loss}$) that aims at scoring positive triples $t^s_{ijk} = (i, k, j)$ higher than negative ones. An additional regularisation term ($R^s_{loss}$) is used to minimise the gap between the token embeddings representation and a pre-trained word type representation of the words they denote [1]. This is conceptually similar to the vector space preservation term in (Mrkšic et al. 2016) in that it controls for how much the token embeddings can deviate from their corresponding word representations.

Our overall goal is to create embeddings that are close, through $R^s_{loss}$, to the original word embeddings (known to capture semantics) and at the same time are syntactically-informed, through $T^s_{loss}$, so as to capture fine-grained semantic differences according to the role a given word plays in a sentence. It is thus important to stress that optimising only $T^s_{loss}$ would be insufficient as it would lack the notion of semantics provided through $R^s_{loss}$.

[1] In the current work we use GloVe (Pennington, Socher, and Manning 2014) pre-trained word type embeddings.




(a)

| | |
|---|---|
| $G$ | corpus of sentence graphs |
| $T_s : r \times \|s\| \times \|s\|$ | relations between tokens in sentence $s \in G$ |
| $R : r \times d \times d$ | relation tensor embeddings |
| $E_s : \|s\| \times d$ | token vector embeddings of tokens in $s \in G$ |
| $d$ | size of the embedding vectors |
| $r$ | size of binary relations |
| $\|s\|$ | number of tokens in sentence $s$ |

$\forall s,\ T_s \approx E_s \cdot R \cdot E_s^T$

(b)

Figure 1: Sentence graph decomposition

The optimisation problem is formulated as

$$\min \sum_{s \in S} \alpha(T^s_{loss}) + (1-\alpha)(R^s_{loss}) \quad (1)$$

where:

$$T^s_{loss} = \sum_{\substack{t^s_{ijk} \in G_s, \\ t^s_{i'j'k'} \in \neg(t^s_{ijk})}} \max(0, \gamma + \langle e^s_{i'}, R_{k'}, e^s_{j'} \rangle - \langle e^s_i, R_k, e^s_j \rangle)$$

and

$$R^s_{loss} = \sum_{e^s_i \in G_s} -\log \sigma(e^s_i \cdot w^s_i)$$

with $G_s$ the graph of sentence $s$ holding all tokens and all relations present in the sentence, $e^s_i$ the token embedding of token $i$ in sentence $s$, $R_k$ the matrix embedding for the relation $k$, $w^s_i$ the pre-trained word type embedding corresponding to the token $e^s_i$, $\gamma$ the margin hyperparameter, $\sigma$ the softmax function, and $\neg(t^s_{ijk})$ the set of negative triples associated with $t_{ijk}$. $E_s$ is the matrix holding all token embeddings for sentence $s$, with one token embedding per row, and $R$ is the tensor of all relations. Given all tokens in one sentence $s$, that results in approximating the relations tensor for sentence $s$, $T_s$ by $E_s^{(|s| \times d)} \cdot R^{(d \times r \times d)} \cdot E_s^{T(d \times |s|)}$.

The regularisation term $R^s_{loss}$ can be seen as a particular case of cross entropy $-y \cdot \log \hat{y} - (1-y) \cdot \log(1-\hat{y})$ with $y = 1$ and $\hat{y} = \sigma(e^s_i \cdot w^s_i)$.

A negatively sampled example in the tensor is obtained by altering one element of the triple while fixing the remaining two: this element can be either one of the entities or the relation holding between them. As mentioned above, given a triple $t^s_{ijk} = (i,k,j)$, we denote by $\neg(t^s_{ijk})$ the set of negative examples associated to it. We consider here that $\neg(t^s_{ijk})$ is formed of the following elements: $\neg(t^s_{ijk}) = \{(i',k,j),(i,k',j),(i,k,j')\}\ \forall i' \neq i, j' \neq j, k' \neq k$.

We optimise Eq.(1) using mini-batch stochastic gradient descent to optimize all $R_k$ and $e^s_i$. In the current proposal we aim to learn a representation for each word token in each sentence as well as for each relation holding between these tokens from scratch. Alternatively, one could leverage pre-trained relations embeddings - a setup whose exploration we leave for future work. We additionally note that the method described can be applied to any other dataset of sentences provided access to parsing information and pre-trained word type embeddings. The obtained representations can be used to predict discourse relations as described further.

## Modelling the discourse arguments

Let us consider the two arguments *Arg*-1 and *Arg*-2 with lengths $n$ and $m$ respectively. We associate each word $w$ with a vector representation $\boldsymbol{x}_w \in \mathbb{R}^d$. Let $\boldsymbol{x}^1_i$ and $\boldsymbol{x}^2_i$ be the $d$-dimensional $i$-th word vector in *Arg*-1 and *Arg*-2 respectively. Then the word representations of the two arguments are: *Arg*-1 = $[\boldsymbol{x}^1_1, \boldsymbol{x}^1_2, \ldots, \boldsymbol{x}^1_n]$ and *Arg*-2 = $[\boldsymbol{x}^2_1, \boldsymbol{x}^2_2, \ldots, \boldsymbol{x}^2_m]$.

**Recurrent architecture.** One of the most widely used models to encode sequences, that enables learning long-term dependencies taking into account contextual information, is the long-short term memory LSTM (Hochreiter and Schmidhuber 1997), a variant of the recurrent neural network. We adopt two LSTM neural networks to model the two arguments separately. Given a word sequence representation $[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k]$ as input, an LSTM computes the hidden state sequence representation $[\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_k]$. At each time step $i$, the model reads $w_i$ as input and updates the $h_i$ hidden state. The final representations for the two arguments are then given by the last hidden state representation for each of them: *Arg*-1 $= \boldsymbol{h}_n$ and *Arg*-2 $= \boldsymbol{h}_m$.

**Convolutional architecture.** In order to represent each argument using convolutional neural networks (CNNs), we follow the approach of (Kim 2014). Given a word sequence representation $[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k]$, let $\boldsymbol{W} \in \mathbb{R}^{h \times d}$ be a filter applied to a window of $h$ words to produce a feature $c_i$, $b$ a bias term and $f$ a non-linear function. Then $c_i = f(\boldsymbol{W} \cdot \boldsymbol{x}_{i:i+h-1} + b)$. A feature map $\mathbf{c} \in \mathbb{R}^{n-h+1}$ is created by applying the filter to each possible window of words in the argument $\{\boldsymbol{x}_{1:h}, \boldsymbol{x}_{2:h+1}, \ldots, \boldsymbol{x}_{k-h+1:k}\}$. Thus $\mathbf{c} = [c_1, c_2, ..., c_{k-h+1}]$, followed by a max-pooling operation $\hat{c} = max(\mathbf{c})$ to obtain the most important feature, the one of highest value, corresponding to the particular filter. For $m$ filters with different window sizes we obtain $m$ features: $z = [\hat{c}_1, ..., \hat{c}_m]$. The representation of each argument is the $m$-dimensional vector $\boldsymbol{z}$: *Arg*-1 $= \boldsymbol{z}_1$ and *Arg*-2 $= \boldsymbol{z}_2$.

## Predicting the discourse relation

The discourse relation holding between two arguments can be predicted with or without modelling the interaction between the two arguments, as presented in the following.

**No interaction between arguments** Once we obtain the

vector representations of each argument (through LSTM, CNN), we concatenate these vectors into a vector of the pair $\boldsymbol{v} = [\boldsymbol{h}_n, \boldsymbol{h}_m]$ for an LSTM encoding and $\boldsymbol{v} = [\boldsymbol{z}_1, \boldsymbol{z}_2]$ for a CNN encoding, which is further passed to a fully connected layer, followed by a softmax layer to obtain the probability distribution over labels. This approach, however, does not focus on modelling the interaction between the two arguments in the discourse relation, an important aspect when predicting discourse relations, as pointed out by previous work (Chen et al. 2016; Lan et al. 2017).

**Collaborative gated neural network** An alternative approach that enables modelling argument interaction is the collaborative gated neural network (CGNN) (Qin, Zhang, and Zhao 2016b). In CGNN the arguments are modelled using CNNs that share parameters and an additional gated unit is used for feature transformation. The input to the CGNN unit is the vector of the pair $\boldsymbol{v} = [\boldsymbol{z}_1, \boldsymbol{z}_2]$ and the set of transformations are: $\hat{c} = \tanh(\boldsymbol{W}^c \cdot \boldsymbol{v} + \boldsymbol{b}^c)$, $\boldsymbol{g}_i = \sigma(\boldsymbol{W}^i \cdot \boldsymbol{v} + \boldsymbol{b}^i)$, $\boldsymbol{g}_o = \sigma(\boldsymbol{W}^o \cdot \boldsymbol{v} + \boldsymbol{b}^o)$, $\boldsymbol{c} = \hat{c} \odot \boldsymbol{g}_i$, $\boldsymbol{h} = \tanh(\boldsymbol{c}) \odot \boldsymbol{g}_o$, where $\odot$ denotes the element-wise multiplication, $\sigma$ denotes the sigmoid function, $\hat{c}$ and $\boldsymbol{c}$ are inner cells, $\boldsymbol{g}_i$ and $\boldsymbol{g}_o$ are the two gated operations and $\boldsymbol{W}^i$, $\boldsymbol{W}^o$ and $\boldsymbol{W}^c$ are parameters of the model. The output of the CGNN unit is the transformed vector $\boldsymbol{h}$ which is further passed to a softmax layer.

We define the training objective as the cross-entropy loss between the output of the softmax layer and the class labels.

## Experimental setup

### Data

Throughout all experiments we use the ( PDTB 2.0 dataset (Prasad et al. 2008) considering only argument pairs annotated with implicit discourse connectives. To enable comparisons with previous work, we follow two popular experimental setups and perform multi-class classification on both level-1 and level-2. We adopt the same split as in (Lin, Kan, and Ng 2009), further denoted as PDTB-Lin and perform multi-class classification for level 2 classes. Similarly to (Lin, Kan, and Ng 2009), we select the most frequent 11 classes. A second split is that of (Pitler, Louis, and Nenkova 2009) denoted PDTB-Pitl, for which we report results for level-1 multi-class classification similarly to previous work.

### Implementation details

For token embeddings computation we run multiple threads varying the initial learning rate, the negative sampling factors and the margin $\gamma$. We fix the ratio between the two losses to $\alpha = 0.5$, optimize using Adam (Kingma and Ba 2014) and do early stopping based on the validation loss. All token embeddings are randomly initialised . We obtain the best results when using an initial learning rate of $10^{-3}$, a sampling factor of 5x and $\gamma$ set to 10. We consider all dependency relations with a frequency higher than 1000 in the corpus.

In all the IDRC experiments we fix the input embeddings to the type or token embeddings. For the optimisation we use Adam and tune the parameters on the development set. Parameters used for the reported results are: for PDTB-Lin the initial learning rate and dropout values are $10^{-4}$ and $0.7$ for LSTM and CGNN and $10^{-5}$ and $0.8$ for CNN. The CNN uses 600 filters and CGNN uses 128. For the PDTB-Pitl split the initial learning rate is $10^{-5}$ for LSTM and CNN and $10^{-4}$ for CGNN. The dropout is $0.6$ for CNN and CGNN and $0.7$ for LSTM. Both CNN and CGNN use 600 filters.

## Results and discussion

We present a comparison of the proposed token embeddings to standard word type embeddings followed by a parallel between our results and state-of-the-art systems and a series of experiments with model variations.

### Comparison to word type embeddings

Table 1 presents the results of three sets of experiments on the PDTB-Lin data, using different input features: standard word embeddings often employed in literature (Pennington, Socher, and Manning 2014; Mikolov et al. 2013), word embeddings trained using dependency contexts (Levy and Goldberg 2014) and our proposed syntactically-aware token representations (SATokE).

| | LSTM | CNN | CGNN |
|---|---|---|---|
| GloVe | 38.97 | 38.25 | 39.03 |
| Word2Vec | 36.92 | 37.33 | 37.07 |
| Deps-WE | 36.00 | 34.98 | 34.98 |
| SATokE | **40.51** | **42.55** | **43.08** |

Table 1: Results for level-2 classification on PDTB-Lin

Using the proposed token embeddings as input yields important improvements over all word embeddings we compare to, across all architectures considered. We obtain improvements between 1.5% and 4.5% when using an LSTM architecture, 3.7% to 7.5% with a CNN encoder and 4% to 8% with CGNN. Unsurprinsingly, feeding SATokE as input to an LSTM encoder only improves results by up to 4.5%: SATokE encode positional information by their construction using adjacency information, and thus they complement less the advantages of using an LSTM encoder.

Embeddings trained on dependency contexts yield consistently worse results than all the other embeddings considered. This is consistent with the analysis in (Ghannay et al. 2016) who show that such embeddings obtain lower results on semantic tasks despite high performance on POS tagging or chuncking. Contrary to a first interpretation, dependency information is useful for semantic tasks: its value is higher when this information is injected into the token representations directly, by leveraging the parse tree of the sentence, than when used as context for creating generic word embeddings from a large corpus. The observed results support this statement with improvements of 4.5% to 8% in absolute accuracy between Deps-WE and SATokE.

### Comparison to related work

Table 2 compares the previously presented results to the related work for the fine-grained multi-class classification of the PDTB-Lin split. Although our results are close to state-of-the-art systems, it is important to note they use additional data to train their models: (Qin, Zhang, and Zhao 2016a)

| Model | Accuracy (%) |
|---|---|
| (Lin, Kan, and Ng 2009) | 40.2 |
| (Qin, Zhang, and Zhao 2016a) | 43.81 |
| (Qin et al. 2017) | 44.65 |
| CGNN+SATokE | 43.08 |

Table 2: Comparison to related work on level-2 PDTB-Lin.

enhance word embeddings with character level information and (Qin et al. 2017) use a more complex model in an adversarial framework leveraging explicit discourse connectives.

To enable further comparisons to more complex models, we run a set of experiments on level-1 multi-class classification on the PDTB-Pitl split, which we compare to related work in Table 3. Some work uses additional data from different corpora or with explicit connectives: (Liu et al. 2016) leverage different discourse corpora, (Ji, Haffari, and Eisenstein 2016) model jointly the discourse relation and the arguments and use additional data from non-implicit relations. Other work focuses on complex architectures to model the interaction between arguments: (Lan et al. 2017) use an attention-based LSTM leveraging explicit discourse relations and unlabelled external data. (Liu and Li 2016) use a multi-level attention mechanism to repeatedly read the arguments involved in a discourse relation along with an external short-term memory to keep track of information. (Dai and Huang 2018) model inter-dependencies between arguments and the sequence patterns of their discourse connectives by positioning them in the wider context of paragraphs. Lastly, (Wang et al. 2017) propose Tree-LSTM and Tree-GRU models to encode the structure of the constituency parse trees and use information from the constituent tags to control for semantic composition. and induce grammatical information. However, none of these investigates the use of syntactic dependencies or can encode arbitrary graphs.

We observe that by using a simple CNN encoder with SATokE as input, we obtain state-of-the-art results, even surpassing most complex models which exploit external data, have a higher number of parameters and/or model the interaction between arguments. It is important to note that out of the models that use neither additional data nor argument interaction, SATokE yields the best results, above ELMo (Peters et al. 2018) that is considered as the current state-of-the-art token embedding method. Lastly, the use of CNN in SATokE yields better results than the use of CGNN, suggesting that even though the CGNN architecture models the interaction between arguments, it may not constitute a powerful enough architecture for this setup.

### Impact of syntax - model variations

While modelling the words in their context seems beneficial, we want to further investigate to what extent syntax plays an important role. In Table 4, an additional set of experiments analyze the impact of using dependency information in the computation of the token embeddings with a CNN architecture. We set the parameters for the token embeddings computation to the ones that obtained the best results in the default scenario. Then we consider two comparative settings: token embeddings computed without the adjacency relation *SATokE-adj*, and without all syntactic relations *SATokE-syntax* respectively. The decrease of performance in results shows that both adjacency relation and syntax play an important role in the final result. However, the results seem to degrade more when information about syntax is removed from the token computation than when adjacency information is not present.

Finally, we consider a set of experiments in which we iteratively remove certain dependency relations considered important, from the token embeddings computation. We obtain tokens computed without information coming from the SUBJ, OBJ and MOD dependency relations. We observe that, for the most part, results degrade the more information is removed from the computation of the tokens.

## Conclusion

The task of IDRC requires an in-depth understanding of the arguments involved in a discourse relation. To tackle this challenging aspect, we propose to use syntactically-informed contextualized word representations. We show that the proposed embeddings outperform standard pre-trained word representations as well as state-of-the-art token embeddings for this task. We additionally show that using simple neural network architectures, we can integrate the proposed representations into a model for IDRC that achieves state-of-the-art results without additional data and without modelling the interaction between arguments.

## References


Braud, C., and Denis, P. 2015. Comparing word representations for implicit discourse relation classification. In *Proceedings of EMNLP*.

Braud, C., and Denis, P. 2016. Learning connective-based word representations for implicit discourse relation identification. In *Proceedings of EMNLP*.

Chen, J.; Zhang, Q.; Liu, P.; Qiu, X.; and Huang, X. 2016. Implicit discourse relation detection via a deep architecture with gated relevance network. In *Proceedings of ACL*.

Dai, Z., and Huang, R. 2018. Improving Implicit Discourse Relation Classification by Modeling Inter-dependencies of Discourse Units in a Paragraph. *ArXiv e-prints*.

Dasigi, P.; Ammar, W.; Dyer, C.; and Hovy, E. 2017. Ontology-aware token embeddings for prepositional phrase attachment. In *Proceedings of ACL*.

Ghannay, S.; Favre, B.; Estève, Y.; and Camelin, N. 2016. Word embeddings evaluation and combination. In *Proceedings of JLRE*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation*.

Honnibal, M., and Johnson, M. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of EMNLP*.


[2]Estimations for parameter count are provided whenever the details in the corresponding work are not sufficient to compute an exact number. All numbers assume an input embedding size of 300.

| Model | Accuracy (%) | Additional data | Arg Interaction | Multi-task | $\|\theta\|$ |
|---|---|---|---|---|---|
| (Liu et al. 2016) | *57.27* | ✓ | ✓ | ✓ | $> 1M$ |
| (Lan et al. 2017) | *57.39* | ✓ | ✓ | ✓ | $> 1M$ |
| (Liu and Li 2016) | *57.57* | - | ✓ | - | $6.7M$ |
| (Dai and Huang 2018) | *58.2* | ✓ | ✓ | - | $> 4M$ |
| (Ji, Haffari, and Eisenstein 2016) | *59.5* | ✓ | ✓ | - | $5.5M$ |
| (Wang et al. 2017) | 56.04 | - | - | - | $6.7M$ |
| CNN+ELMo | 57.81 | - | - | - | $4M$ |
| CNN+SATokE | **58.83** | - | - | - | $4M$ |
| CGNN+SATokE | 57.90 | - | ✓ | - | $41M$ |

Table 3: Comparison to related work on level-1 PDTB-Pitl. In italic and above the double line are reported scores. All non-italic scores below the double line are (re)produced in the current work. [2]

| Model variations | Accuracy (%) |
|---|---|
| **SATokE** | **58.83** |
| SATokE-adj | 57.81 |
| SATokE-syntax | 57.65 |
| SATokE-SUBJ | 58.83 |
| SATokE-SUBJ-OBJ | 57.90 |
| SATokE-SUBJ-OBJ-MOD | 57.56 |

Table 4: Results for level-1 classification on PDTB-Pitl using variations of the proposed token embeddings model.


Ji, Y., and Eisenstein, J. 2015. One vector is not enough: Entity-augmented distributed semantics for discourse relations. In *TACL*.

Ji, Y.; Haffari, G.; and Eisenstein, J. 2016. A latent variable recurrent neural network for discourse-driven language models. In *Proceedings of NAACL HLT*.

Kim, Y. 2014. Convolutional neural networks for sentence classification. In *Proceedings of EMNLP*.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Lan, M.; Wang, J.; Wu, Y.; Niu, Z.-Y.; and Wang, H. 2017. Multi-task attention-based neural networks for implicit discourse relationship representation and identification. In *Proceedings of EMNLP*.

Levy, O., and Goldberg, Y. 2014. Dependency-based word embeddings. In *Proceedings of ACL*.

Lin, Z.; Kan, M.-Y.; and Ng, H. T. 2009. Recognizing implicit discourse relations in the penn discourse treebank. In *Proceedings of EMNLP*.

Liu, Y., and Li, S. 2016. Recognizing implicit discourse relations via repeated reading: Neural networks with multi-level attention. In *Proceedings of EMNLP*.

Liu, Y.; Li, S.; Zhang, X.; and Sui, Z. 2016. Implicit discourse relation classification via multi-task neural networks. In *Proceedings of AAAI*.

McCann, B.; Bradbury, J.; Xiong, C.; and Socher, R. 2017. Learned in translation: Contextualized word vectors. In *Proceedings of NIPS 30*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS 26*.

Mrkšic, N.; Ó Séaghdha, D.; Thomson, B.; Gašic, M.; Rojas-Barahona, L.; Su, P.-H.; Vandyke, D.; Wen, T.-H.; and Young, S. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of NAACL HLT*.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*.

Peters, M.; Ammar, W.; Bhagavatula, C.; and Power, R. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of ACL*.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. In *Proceedings of NAACL HLT*.

Pitler, E.; Louis, A.; and Nenkova, A. 2009. Automatic sense prediction for implicit discourse relations in text. In *Proceedings of ACL*.

Prasad, R.; Dinesh, N.; Lee, A.; Miltsakaki, E.; Robaldo, L.; Joshi, A.; and Webber, B. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of LREC*.

Qin, L.; Zhang, Z.; Zhao, H.; Hu, Z.; and Xing, E. 2017. Adversarial connective-exploiting networks for implicit discourse relation classification. In *Proceedings of ACL*.

Qin, L.; Zhang, Z.; and Zhao, H. 2016a. Implicit discourse relation recognition with context-aware character-enhanced embeddings. In *Proceedings of COLING*.

Qin, L.; Zhang, Z.; and Zhao, H. 2016b. A stacking gated neural architecture for implicit discourse relation classification. In *Proceedings of EMNLP*.

Salant, S., and Berant, J. 2018. Contextualized word representations for reading comprehension. In *Proceedings of NAACL HLT*.

Tu, L.; Gimpel, K.; and Livescu, K. 2017. Learning to embed words in context for syntactic tasks. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*.

Wang, Y.; Li, S.; Yang, J.; Sun, X.; and Wang, H. 2017. Tag-enhanced tree-structured neural networks for implicit discourse relation classification. In *Proceedings of IJCNLP*.