Learning Clause Representation from Dependency-Anchor Graph for Connective Prediction

Yanjun Gao and **Ting-Hao (Kenneth) Huang** and **Rebecca J. Passonneau**
Pennsylvania State University
{yug125,txh710,rjp49}@psu.edu

Abstract

Semantic representation that supports the choice of an appropriate connective between pairs of clauses inherently addresses discourse coherence, which is important for tasks such as narrative understanding, argumentation, and discourse parsing. We propose a novel clause embedding method that applies graph learning to a data structure we refer to as a dependency-anchor graph. The dependency anchor graph incorporates two kinds of syntactic information, constituency structure and dependency relations, to highlight the subject and verb phrase relation. This enhances coherence-related aspects of representation. We design a neural model to learn a semantic representation for clauses from graph convolution over latent representations of the subject and verb phrase. We evaluate our method on two new datasets: a subset of a large corpus where the source texts are published novels, and a new dataset collected from students' essays. The results demonstrate a significant improvement over tree-based models, confirming the importance of emphasizing the subject and verb phrase. The performance gap between the two datasets illustrates the challenges of analyzing student's written text, plus a potential evaluation task for coherence modeling and an application for suggesting revisions to students.

1 Introduction

The clause is a fundamental unit in coherent text. Much work in NLP investigates how clauses combine to form larger units, ultimately spanning a whole discourse (Wang et al., 2017; Ji and Eisenstein, 2014); how to decompose complex sentences into distinct propositions (Wang et al., 2018; Li et al., 2018; Narayan et al., 2017); how to identify explicit or implicit semantic relations between clauses (Lee and Goldwasser, 2019; Rutherford and Xue, 2015), or how to select a connective to link multiple clauses into a complex sentence (Nie et al., 2019; Malmi et al., 2018). In this paper, we focus on clause representation to support accurate connective prediction, a task which is important for coherence modeling (Pishdad et al., 2020), fine-grained opinion mining (Wiegand et al., 2015), argument mining (Kuribayashi et al., 2019; Jo et al., 2020) and argumentation (Park and Cardie, 2014). We present a case for a model that learns from a novel graph we refer to as a *dependency-anchor graph*, which retains information from dependency parses and constituency parses of input sentences that is critical for identification of the core proposition of a clause, while omitting structural information that is less relevant.

P_1	Bob cooked Tia a burger.	P_1, Q_1	alth
P_2	Bob cooked himself a burger.	P_1, Q_2	bec
Q_1	Bob was hungry.	P_1, Q_3	none
Q_2	Tia was hungry.	P_1, Q_4	alth
Q_3	Bob was thirsty.	P_2, Q_1	bec
Q_4	Tia was thirsty.	P_2, Q_2	alth
alth(ough): contrast		P_2, Q_3	alth
bec(ause): precondition		P_2, Q_4	none

Figure 1: For the propositions P_m, Q_n to be joined by *although* or *because*, P_m and Q_n. must have some semantic commonality to allow for contrast or causation. Four cases allow *although*. The two cases that allow *because* have a strong semantic relation between the predicates (*make someone a burger, be hungry*) and, there is no conflict in the *to*-object of the first clause and the subject of the second. The remaining two cases have no commonality, and neither connective can occur.

We assume that determining whether two clauses can be joined by a connective, and what connective to choose, depends primarily on the main verb in each clause, and on the arguments that occur in both clauses, particularly the grammatical subject. There are a large number of connectives and connective phrases in English; e.g., the Penn Discourse Tree Bank (Prasad et al., 2008) has 141. Here we illustrate the nature of the problem with respect to the two connectives, *although* and *because*. Figure 1 illustrates how the choice of connective to join two simple clauses P_m and Q_n, and whether

Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15), pages 54–66

a connective is appropriate at all, depends on the main verbs and their arguments. Use of *although* requires only some dimension of contrast between the joined clauses, while *because* requires that Q_n be a precondition for P_m. The table lists two variants of P_m with *cook* as the main verb, one with three distinct entities (Bob, Tia, a burger) and one with two (Bob, burger). These are considered in turn with four variants of Q_n where the predicate is either closely related to *cook* (e.g.*be hungry*) or not (e.g., *be thirsty*), and the two propositions share an argument or not. In two of the eight cases, neither connective makes sense because there is no other cohesive relation (e.g., coreference, association) between the clauses. In four cases, there is some similarity and some contrast, which licenses *although*, and in two cases the more restrictive condition that licenses *because* is present. These examples illustrate that both the choice of verb, and the grammatical relations of the arguments to the verb, affect whether a connective can be used, and which one. Adding modifiers on the subject or object, or VP or sentence adverbials, would have little effect on choice of connective in these sentences.

We assume that training clause representations based on connective prediction will be useful for developing representations that capture aspects of coherence, such as those shown in Fig 1. Pishdad et al. (2020) examine a series of coherence evaluation tasks that capture different aspects of coherence. They argue that connective substitution is one of four critical tests of coherence modeling. For example, connectives that express temporal succession should not be substitutable for connectives that express simultaneity, as doing so would change the meaning. Studies of students' writing skills look at connectives with respect to quality of students' argumentative writing (Kuhn et al., 2016), and whether automated assessments differ for low-skilled versus high-skilled writers (Perin and Lauterbach, 2018). Although students can fill in correct connectives eliminated from source texts, they typically do not use connectives as precisely in their own writing (Millis et al., 1993). NLP applications aimed at supporting student revision use connectives as an indication of writing quality (Nguyen et al., 2016; Afrin and Litman, 2018), but do not help students choose correct connectives. To better evaluate model performance in connective selection, and to highlight differences between text from skillful versus developing writers, we provide two large datasets of clauses linked by connectives drawn from published fiction and from students' written text. We demonstrate the potential for a model trained on expert data to identify incorrect uses of connectives in students' writing, where students frequently misuse connectives like *and*.

Our contributions are: 1) a data structure we refer to as a *Dependency-Anchor* graph that incorporates information from both dependency and constituency trees; 2) *DAnCE* (**D**ependency-**An**chor graph representation for **C**lause **E**mbedding), a novel neural architecture that exploits bi-LSTMs at the lower layers for learning inter-word influences, and graph learning of relational structure encoded in the dependency-anchor graph; 3) two datasets for carefully edited versus student text. Our approach outperforms the state-of-the-art on connective prediction.

2 Motivation

The question of whether latent representations of sentence meaning can benefit from syntax has been addressed in work that compares recurrence and recursion, and finds the main benefit of recursive models to be better treatment of long-distance dependencies (Li et al., 2015). Two recent works compare tree-based models derived from dependency parses with constituency parses on semantic relatedness tasks, with no clear advantage of one grammar formalism over the other (Tai et al., 2015; Ahmed et al., 2019). As discussed in (Tai et al., 2015), dependency trees provide a more compact structure than constituency trees, through shorter paths from the root to leaf words. Further, all of a verb's arguments are its direct dependents. The recursive structure of constituency trees, on the other hand, facilitates identification of subtrees that span more of the leaf words as one moves up the tree, and that have a compositional contribution to the meaning of the sentence. Tree-based models take input from syntactic parses and compose the latent vectors through a uni-directional traversal, where the parent node representation is the sum of the child nodes. For both formalisms, many parameters are needed to encode the child-to-parent representations. For this reason, previous work strictly limits the model dimensionality (Tai et al., 2015; Ahmed et al., 2019).

To combine advantageous features from both kinds of grammar formalism, we propose dependency-anchor graphs as a compact represen-

tation that highlights the core elements of a proposition. We construct a graph with only selected components from two kinds of parse trees, thus limiting the number of parameters to learn. The subject of a clause and the main verb phrase are the two outer nodes in the graph, where we refer to the verb phrase node as the *anchor*. The subject arc from a dependency parse points from the anchor node to the subject. The anchor node is a subgraph that retains the dependency structure of words within the verb phrase. Other syntactic relations (e.g., involving words in the subject phrase or adverbial phrases), are ignored.

To encode the graph, we propose DAnCE, which applies graph convolution (GCN) (Kipf and Welling, 2017) to encode the arc between the subject and verb phrase. The input to the graph convolution comes from a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) that encodes all the input tokens, including the words outside the subject and verb phrase. The interaction that DAnCE captures between subject and verb phrase has been essential in word representation but missing in tree based models (Weir et al., 2016; White et al., 2018).

We demonstrate the effectiveness of the dependency-anchor graph and DAnCE architecture through its superior performance over baselines, including tree-based models. The rest of the paper is organized as follows: we first present related work and give a detailed discussion of the Dependency-Anchor Graph and DAnCE. Then we present the datasets, experiments and discussion.

3 Related Work

Much research has addressed ways to learn high quality clause representations. Xu et al. (2015) propose a shortest dependency path LSTM for sentence representation in the task of relation classification. Dai and Huang (2018) propose a BiLSTM based model that combines paragraph vectors and word vectors into clause embeddings for a situation entity classification task. Connective prediction has often been addressed: Ji and Eisenstein (2015) and Rutherford et al. (2017) use recursive neural networks with parse trees as input to predict connectives and discourse relations, with solid improvements on PDTB. Malmi et al. (2018) use a decomposable attention model to predict connectives on sentences pairs extracted from Wikipedia. Our work draws on the idea of incorporating syntax into representation for connective prediction, specifically for clauses.

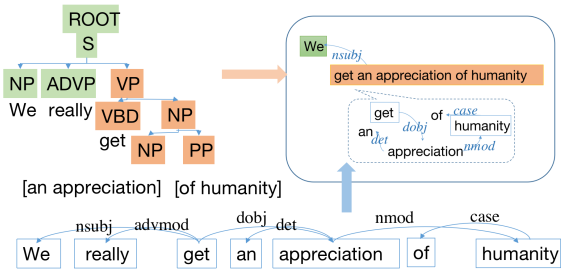

Figure 2: A dependency-anchor graph for a clause (top right) is constructed from its phrase-structure parse (top left) and dependency parse (bottom). Words spanning the VP subtree of the constituency parse (orange nodes) become a single anchor node whose internal structure preserves the dependencies among words in the VP. The nsubj dependent of the main verb is promoted to be a dependent of the entire anchor.

Sileo et al. (2019) propose a large dataset with 170M sentence pairs with connectives for unsupervised sentence representation learning, and apply it on the SentEval task. Nie et al. (2019) develop universal sentence embeddings from a connective prediction task, and create a large corpus extracted from published fiction. They achieve state-of-the-art performance on predicting connectives, as well as on sentence embedding benchmarks from SentEval (Conneau and Kiela, 2018). Our work modifies the corpus from (Nie et al., 2019) to restrict the pairs of sentences for connective prediction to simple sentences. Our goal is to generate clause embeddings specifically for connective prediction, rather than universal sentence representation.

4 DAnCE Architecture

The input to DAnCE is a graph for each simple sentence that includes syntactic information from a phrase structure parse to identify the VP, and from a dependency parse to identify the grammatical subject, and dependencies within the VP.

4.1 Dependency-Anchor graph

The anchor VP and its subject serve as nodes in a graph, as illustrated in Figure 2. The Stanford CoreNLP dependency grammar has 58 dependency relations, eight of which are a type of subject (Van Valin, 2001; Schuster and Manning, 2016). The subject in our dependency-anchor graphs can originate as any of these eight types, and is represented as a node with a single subject edge to the anchor. The anchor node has internal graph structure, that replicates the dependency relations among the words in the VP. We align two syntax parses by the words then extract the depen-

dencies between words inside the anchor. Each dependency-anchor graph constitutes a complete proposition. The dependency relation from the anchor to the subject, and the other dependencies for words within the VP, differentiate words by their closeness to the root verb of the dependency parse. Words outside the subject-anchor are omitted from the graph to maintain the focus of subject-VP, but they are encoded by the BiLSTM as part of the sequence and contribute to the hidden states.

4.2 Neural architecture

To learn a semantic representation from a dependency-anchor graph, DAnCE has the three layers illustrated in Figure 3. An initial embedding lookup layer retrieves word embeddings. A BiLSTM layer captures the hidden states over the input words at each time step. Finally, a graph convolution layer takes the subject word representation from the BiLSTM (the $\mathcal{S}$ node in Figure 3), and an *anchor embedding* that is generated from an separate module (the $\mathcal{A}$ node in Figure 3), to produce the final learned semantic representation.

The input sequence of words $x_i \in X$ is first fed into a pre-trained word embedding lookup layer, using GloVe (Pennington et al., 2014), with a bidirectional LSTM of dimension $2D$, where D is the dimension of hidden states in the BiLSTM:

$$h_i = f(x_i, h_{i-1}), h_i \in \mathbb{R}^{2D} \quad (1)$$

The BiLSTM captures long-term dependencies within the clause. The resulting latent representation for the subject is fed directly to the graph convolution layer. The anchor embedding h^A is computed with two alternative settings: **F**lat-**A**nchor (FA) and **G**raph-**A**nchor (GA). The main difference between the two settings is that *FA* treats the anchor as a sequence of words with their BiLSTM hidden states h_i, and ignores the dependency relations within the anchor. *GA* turns the dependencies into an adjacency matrix and then generates $h_i^{A_G}$ as the anchor node representation by encoding the BiLSTM hidden states within the matrix through graph attention (GAT) (Veličković et al., 2018). GAT will attend to whatever nodes are within the anchor, thus it fits well for learning the anchor representation for any length anchor. We first explain the derivation of $h_i^{A_G}$.

Following (Marcheggiani and Titov, 2017), we treat the dependency arcs within the anchor as directed. Given the latent representations of a pair of nodes within the anchor h_i, h_j, and a one-hot

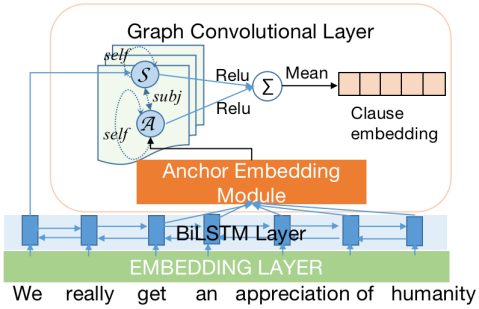

Figure 3: Overall architecture of DAnCE.

vector for each dependency arc $arc_{i,j}$, we compute an attention coefficient $e_{i,j}$:

$$e_{i,j} = a(W^h h_i, W^d[h_j || arc_{i,j}]) \quad (2)$$

where $||$ is the concatenation operation, and a, W^h, W^d are learned parameters for the head and the dependent. Then we apply softmax and a Leaky ReLU activation to normalize the attention weights:

$$\alpha_{i.j} = LeakyReLU(\frac{exp(e_{i,j})}{\sum_{m \in N_A(i)} exp(e_{i,m})}) \quad (3)$$

where $N_A(i)$ represents all nodes in the anchor that are linked to i, including itself. Leaky Relu activation on $e_{i,j}$ enables the network to learn the importance of node j and arc i, j to node i. Therefore, $\alpha_{i,j}$ is a vector, whose length is the number of anchor words, that represents differential attention on word pairs associated with their dependency relations. We apply the attention weights on the node features from the first BiLSTM layer:

$$h_i^{A_G} = \sum_{j \in N_A(i)} \alpha_{i.j} W^{A_G} h_j \quad (4)$$

Again, there are two alternative settings to generate the anchor embedding. We use *maxpool* over all the nodes in anchor N_A:

$$h^A = \begin{cases} Maxpool(||_{i \in N_A} h_i) & \text{if FA} \\ Maxpool(||_{i \in N_A} h_i^{A_G}) & \text{if GA} \end{cases} \quad (5)$$

The third layer applies graph convolution (GCN) to the subject hidden states from BiLSTM and subject and anchor nodes, where the subject node is the hidden state from the BiLSTM and the anchor node is the anchor embedding h^A. Given a node i, we first compute its GCN node embedding $h_{S_i}^{k+1}$ from its neighbor $N(i)$, including a self loop, $i \in N(i)$:

$$h_{S_i}^{k+1} = ReLu(\sum_{j \in N(i)} W^{S_k} h_j^{S_k} + b^{S_k}) \quad (6)$$

where k represents the k-order neighbor (the maximum hop between two nodes). W^{S_k} and b^{S_k} are the learned weights and bias. We use $k = 1$, as there is only one edge between the subject and anchor, thus $h_{S_i}^{k=0}$ is either the anchor embedding h^A or the BiLSTM output for the subject word. The node representation is thus more informative by merging with its relevant neighbor through graph convolution, and enhances the final aggregation. Once the GCN node features are obtained, we compute the final embedding h_S^K as the average over all node features N_k at the last layer K,

$$h_S^K = \frac{1}{|N_K|} \sum_{v \in N_K} h_v^K, h_S^K \in \mathbb{R}^{2D} \quad (7)$$

5 Data Collection

This section introduces two corpora we use in our experiments. They differ in genre, size, and distribution of connectives, as well as a contrast between spontaneous student writing and carefully edited text. They also differ in the way they were annotated, and in whether they include negative examples. *DeSSE* (**De**composed **S**entences from **S**tudent **E**ssays) consists of sentences from students' opinion essays, 78% of which are complex. The annotation of DeSSE rewrites complex sentences into atomic tensed clauses, omitting any discourse connectives. Sentences are considered complex if there are at least two clauses with tensed verbs, thus a sentence consisting of a subject, verb and its clausal argument are not considered complex. The corpus also includes complex sentences with relative clauses rather than connectives, which serve as negative examples for connective prediction. We assume that a model should be able to discriminate between cases where two clauses have a cohesive relation other than one given by a connective. This is analogous to the motivation for inclusion of adversarial examples in a recent corpus for natural language arguments (Niven and Kao, 2019). In that work, it was shown that transformer models that appeared to perform well without the adversarial examples were exploiting accidental correlations, given that performance degraded significantly once adversarial examples were included. Previous work has shown similar results that neural models for summarization learn more about the position of lead sentences in news articles than about the actual meanings of sentences, due to the lead bias in news (Kedzie et al., 2018).

Dataset	Size	Avg Length	Vocab
Book-Simpl	644k	7.61	52,957
DeSSE	70k	10.19	11,186

Table 1: Descriptive statistics comparing Book-Simpl and DeSSE, including the number of clause pairs (Size), average clause length, and vocabulary size.

DeSSE consists of 39K source sentences, with 68 connectives of the 141 connective words and phrases identified in PDTB. Most connectives occur with very low frequency. More than 50% of pairs are connected by *and*, punctuation, or no connective. Fifty-five of the 68 connectives are rare with frequencies below 1% of the total. A detailed distribution is shown in appendix A.[1]

Our second corpus is a modification of the Book corpus, which consists of connective prediction data taken from published novels (Nie et al., 2019). The Book corpus extracts pairs of simple or complex sentences from source texts, where a connective linked the pair. The original Book corpus contains 15 connectives, and two subsets of 8 and 5 connectives. We created subsets consisting of connectives that joined simple clauses: Book-Simpl 5 with their 5 connectives (285K clause pairs), and Book-Simpl 8 with their 8 (359K clause pairs).

Table 1 shows that the average clause length for DeSSE is longer than in Book-Simpl, with one-fifth the total vocabulary. In comparison to Book-Simpl, the language in DeSSE is less formal and coherent.

5.1 DeSSE

DeSSE includes identification of complex sentences with tensed clauses, and excludes infinitival or gerundive clauses, as a first step towards training corpora for clause identification. It covers a wide range of intra-sentential syntactic and semantic phenomena. It includes all tensed clauses occurring in conjoined structures, including subordinating conjunctions, along with relative clauses, parentheticals, and conjoined verb phrases. It excludes clausal arguments of verbs, because the semantic relationship of the clausal argument in its sentence is given by the verb semantics. The annotation process is unique in that it involves identifying where to split the source sentence into distinct clauses, and how to rephrase the source sentence into a set of complete, independent clauses that omit any discourse connectives. It is designed for developing

[1]DeSSE is available at https://github.com/serenayj/DeSSE. DAnCE is available at https://github.com/serenayj/DAnCE.

1. (If you have not experienced *what they have experienced*), **then** you will never truly understand.
2. (I believe that *talking about race more in a civil way can only improve our society*), **but** I can see why other people may have a different opinion.

Figure 4: Original sentences from DeSSE with intra-sentential connectives, where the clause preceding the connective contains a relative clause (example 1), or a clausal argument of the main verb (example 2).

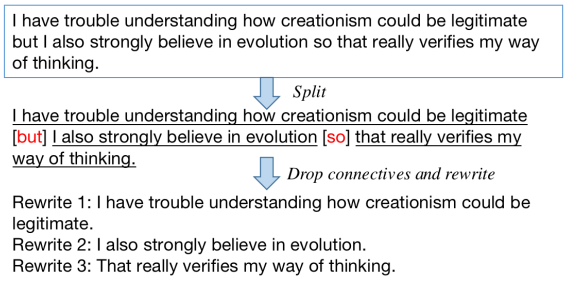

Figure 5: Example annotation from DeSSE. Annotators first split a sentence into segments (underlined text), then rewrite the segments into complete sentences, omitting connectives.

connective prediction, sentence segmentation and decomposition, and semantic representation.

Figure 4 illustrates intra-sentential connectives (*then, but*) that join two clauses. In example 1), the first clause (in parentheses) contains a free relative clause as a verb argument (in italics). In example 2), the first clause contains a clausal argument of the main verb. In both cases, however, the entire first clause is the first argument of the connective.[2]

We collected over 17,000 opinion essays written by U.S. university students in a large undergraduate social science class. Students watched video clips about race relations, and wrote essays in a blog environment to share their opinions with the class. We selected 39K sentences out of 173K for annotation, corresponding to the first 3,592 essays.

Amazon Mechanical Turk (AMT) is a popular crowdsourcing platform for NLP annotation. While it facilitates data collection, using untrained annotators requires care. In a series of pilot tasks on AMT, we iteratively designed annotation instructions and an annotation interface, while monitoring quality. Figure 5 illustrates two steps in the annotation: identification of n split points between tensed clauses, and rephrasing the source into $n+1$ simple clauses, where any connectives are dropped. The final version of the instructions describes the two annotation steps, provides a list of connectives, and illustrates a positive and negative example.[3]

The ten most frequent connectives in DeSSE are *and, because, when, as, so, or, for, if, also, but*. We postprocess the corpus to identify pairs of clauses from complex sentences, and any connectives. The resulting dataset has the following distribution: a single atomic clause (22%), two clauses (45%) or more than two clauses (33%). Given sentences with exactly two atomic clauses in the source, 30% joined them with a discourse connective.

5.2 Book-Simpl

Nie et al. (2019) presented the Book corpus, which has 15 frequently used connectives and 4.7M pairs of sentences. Their goal was to exploit the semantic relationship given by the connective prediction task to improve sentence representation, as noted above. The Book corpus contains two versions, Book-5 with 5 connectives: *and, but, if, because, when*; and Book-8, an extended version with 3 more connectives: *before, though, so*. The sentences linked by a connective can be simple or complex.

To create a subset of the Book corpus that is more parallel to DeSSE, we selected Book corpus examples where the connective linked two simple clauses. The new Book-Simpl dataset has a distribution of connectives similar to the Book corpus (see appendix A).

6 Experiments

For our experiments to predict connectives, we use the same classifier used in (Nie et al., 2019). An input pair of sentence vectors representing the clauses to be joined by a connective are concatenated with vectors resulting from three pairwise vector operations: averaging, subtraction and multiplication. The concatenated vectors are fed into three fully-connected layers, then projected to a lower dimension prior to softmax over the classification categories.

Experiments on Book-Simpl predict the correct connective, given positive examples of clause pairs. Experiments on DeSSE predict the correct connective, given positive and negative examples. We compare DAnCE with four baselines on both datasets, reporting accuracy and F1. Student writing is much less coherent than much of the text that applies NLP to tasks related to discourse structure,

[2] As in (Webber and Joshi, 1998), we take connectives to be predicates whose arguments are the clauses they join.

[3] The interface checked for connectives remaining in step two to warn annotators. Details about the interface and quality control are included in appendix B.

Group	Model	Book-Simpl 5		Book-Simpl 8		DeSSE 5		DeSSE 8	
		Acc. (σ)	F1	Acc. (σ)	F1	Acc. (σ)	F1	Acc. (σ)	F1
BoW	CNN	61.89 (1.64)	49.70	46.31 (1.36)	30.62	**53.57** (0.27)	17.70	**42.95** (0.22)	9.18
SeqLSTM	DisSent	68.58 (1.55)	58.78	62.92 (1.39)	48.11	48.93 (0.31)	**25.27**	39.86 (0.30)	**15.91**
Tree	Tree	67.95 (1.10)	59.67	59.69 (1.58)	45.71	20.35 (0.74)	9.84	16.63 (0.77)	9.29
LSTM	Tr-Attn	69.08 (0.82)	62.30	63.48 (1.40)	49.06	18.51 (0.10)	8.68	17.95 (0.72)	12.01
DAnCE	FA	**71.83** (0.45)	**63.59**	**65.60** (0.55)	**51.26**	52.64 (0.38)	22.29	41.75 (0.25)	13.88
Models	GA	71.51 (1.45)	59.93	65.38 (1.57)	50.28	53.48 (0.46)	14.73	12.01 (0.48)	9.16

Table 2: Performance of baselines and our models on Book-Simpl 5 (N=16,538), Book-Simpl 8 (N=18,946), DeSSE 5 (N=3,466) and DeSSE (N=3,894).

such as discourse connective prediction, discourse parsing, and semantic representation of clauses. We find all models perform better on Book-Simpl than DeSSE, and DAnCE-FA yields good performance on both corpora.

6.1 Baselines and settings

We use three kinds of architecture as baselines: Bag-of-words feed-forward networks (*BoW*), Tree-based LSTM (*Tree-LSTM*), and sequential LSTM (*Seq-LSTM*). For BoW group, we include Glove-CNN (Kim, 2014), a widely used convolutional network for text classification that takes word vectors as input and generates sentence vectors. The Tree-LSTM group includes two models: Dependency Tree-LSTM (Tree) (Tai et al., 2015), which encodes the dependency parse, and an improved version of Tree-LSTM with attention (Tr-Attn) (Ahmed et al., 2019). The Seq-LSTM group consists of DisSent (Nie et al., 2019; Conneau et al., 2017), a BiLSTM model with max-pooling over all hidden units of sentences, and self attention. Hyperparameters are shown in Appendix D.

Our experiments ask two questions: 1) How does DAnCE, which relies on graph convolution, and whose input is a Dependency-Anchor graph, compare with tree-based models? 2) How does DAnCE compare against the two types of models that do not rely on syntax (sequence-based and BOW). The two anchor settings for DAnCE enable us also to test alternative DAnCE settings (DAnCE-FA and DAnCE-GA). Our question here is whether learning from dependency relations within verb phrases through graph attention produces better representations for connective prediction. [4]

6.2 Results

Table 2 reports mean accuracy and the standard deviation from 16 bootstrapped iterations on 90% of the test data, and F1 for the full test data. Bootstrapped F1 standard deviation shows the same magnitude as accuracy therefore is omitted from the table. Overall, all models report higher accuracy and F1 on Book-Simpl than DeSSE, which suggests that including “no connectives" increases the difficulty of the learning task. For Book-Simpl, increasing the number of connectives also increases the prediction difficulty, reflected in lower accuracy and F1 scores for all models on Book-Simpl 8 in comparison to Book-Simpl 5. DAnCE outperforms all baselines, DisSent falls between the two tree variants, and the BoW model has the lowest performance. On DeSSE 5 and 8, however, it is the BoW model that shows the highest accuracy. DisSent achieves the highest F1 on both versions of DeSSE. DAnCE-FA has higher accuracy but slightly lower F1 than DisSent, and both greatly outperform the two tree models and DAnCE-GA.

Recall that DeSSE includes adversarial samples, hence evaluation on DeSSE may be more revealing in comparison to Book-Simpl. Figure 6 gives a breakdown of F1 by connective on DeSSE 5 and 8 for DAnCE-FA, DisSent and Tree-Attn. It is surprising that for DeSSE 5, Tree-Attn fails completely on *and, so, as*, while it outperforms DisSent and DAnCE-FA on *because, no connective*. On DeSSE 8, Tree-Attn fails to predict *and, for, if*. DAnCE-FA and DisSent have similar F1 scores

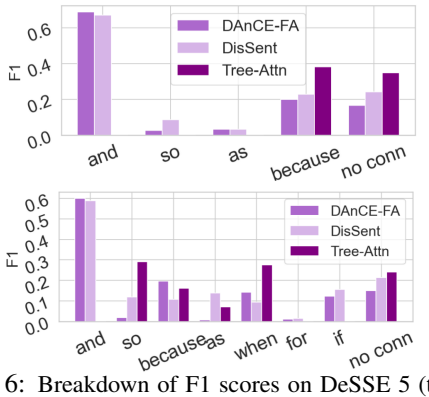

Figure 6: Breakdown of F1 scores on DeSSE 5 (top) and DeSSE 8 (bottom) from DAnCE-FA, DisSent and Tree-Attn.

[4] We attempt to train a fine-tune BERT on our dataset, however due to the size of the training set we make no success in finetuning.

DAnCE-GA	Book-Simpl (F1)		DeSSE (F1)	
Settings	5	8	5	8
-GCN	65.32	37.85	7.75	5.63
-DIR	58.34	47.85	17.44	11.75

Table 3: Ablation studies of DAnCE.

DeSSE 5		DeSSE 8	
(Obs, Pred)	Pairs	*(Obs, Pred)*	Pairs
(and, but)	24.0%	*(and, but)*	18.0%
(and, and)	17.0%	*(and, and)*	12.0%
(because, but)	7.8%	*(None, but)*	7.3%
(None, but)	6.8%	*(because, but)*	5.6%
(so, but)	4.1%	*(and, when)*	4.8%

Table 4: For all pairs of sentences in the DeSSE test sets, we compare the observed student's usage (Obs) with the model prediction (Pred) from DAnCE-FA trained on Book Simpl, and sorted each pattern of observation and prediction by frequency. The five most frequent of these patterns are shown here for DeSSE 5 and DeSSE 8.

on *and, when, if*, but DAnCE-FA rarely or never predicts the connectives *so, for* and *as*. It may be that subtle differences in meaning based on sentence elements apart from the subject and verb phrase are predictive, given the failure of DAnCE-FA to perform at all well on these connectives. To summarize, DAnCE-FA performs comparably to DisSent and shows improvements over tree-based models. DAnCE-GA is worse than DAnCE-FA, which might be attributed to the noisy information introduced by dependency arcs within the anchor.

6.3 Ablation Experiment

We conducted an ablation test on DAnCE-GA to address the following questions: 1) does the performance drop if subject and verb are not highlighted? and 2) do undirected dependency arcs result in better performance within the anchor. To address the first question, we remove the GCN layer (*-GCN*). To address the second, we remove the directionality of dependency arcs inside the anchor to produce a symmetric adjacency matrix (*-DIR*). Table 3 presents F1 scores on the for sets of connectives from Book-Simpl and DeSSE. Compared to the DAnCE variants presented in Table 2, removing the emphasis on subject and verb significantly lowers the performance, especially on DeSSE. Using a symmetric adjacency matrix for graph attention results in lower performance on Book-Simpl, but surprisingly higher F1 on DeSSE. This shows that our emphasis on the subject and verb phrase enhances clause representation. However, incorporating more dependency arcs within the anchor degrades the performance.

Clause.1	He said he grew up as a Christian.		
Clause.2	He then converted to Islam.		
Student	*and*	**DAnCE-FA**	*but*
Clause.1	He trusted his faith.		
Clause.2	It helped him move on.		
Student	*and*	**DAnCE-FA**	*because*

Table 5: Example pairs of clauses from DeSSE 5, showing the connective used by the student alongside the prediction from DAnCE-FA trained on Book Simpl 5.

7 Discussion

Here we discuss the potential to suggest an alternative connective for students when their choice of connective differs from a connective predicted by a model that has been trained on professionally written text. The benefits of this analysis are two-fold: it explores the feasibility of an education application to help students revise their choice of connective, and it allows us to examine DAnCE's ability to model aspects of coherence that pertain to choice of connective. For all pairs of sentences in DeSSE 5 and 8, we compared the observed choice made by the student writer with the prediction from DAnCE-FA trained on Book Simpl 5 or Book Simpl 8. Table 4 shows the five most frequent pairs of student choice in DeSSE 5 or DeSSE 8 versus the prediction from the model trained on Book Simpl 5 (left columns) or trained on Book Simpl 8 (right columns). As illustrated, in many of the cases where students used *and*, the model trained on text from professional writers predicts *but*. Figure 5 shows a few examples where a student used the semantically neutral conjunction *and*, the model predicted a more specific conjunction, and the model's prediction seems more precise. Future work will investigate in detail the feasibility of suggesting alternative connectives.

8 Conclusion

This paper presented the dependency-anchor graph, a new data structure emphasizing the propositional structure of clauses, and DAnCE, a neural architecture with a distinct module for learning verb phrase representation, and graph convolution for semantic relation between the verb phrase and its subject.DAnCE shows good performance on two datasets for connective prediction, and introduces a potential application that could help students revise their writing through improved choice of connectives. Future work will extend DAnCE for coherence modeling within and across sentences, and for applications to support students' revisions.

References

Tazin Afrin and Diane Litman. 2018. Annotation and classification of sentence-level revision improvement. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 240–246.

Mahtab Ahmed, Muhammad Rifayat Samee, and Robert E. Mercer. 2019. Improving tree-LSTM with tree attention. In *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, pages 247–254. IEEE.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Zeyu Dai and Ruihong Huang. 2018. Building context-aware clause representations for situation entity type classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3305–3315.

Chris Fournier. 2013. Evaluating text segmentation using boundary edit distance. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1702–1712.

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

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24, Baltimore, Maryland. Association for Computational Linguistics.

Yangfeng Ji and Jacob Eisenstein. 2015. One vector is not enough: Entity-augmented distributed semantics for discourse relations. *Transactions of the Association for Computational Linguistics*, 3:329–344.

Yohan Jo, Elijah Mayfield, Chris Reed, and Eduard Hovy. 2020. Machine-aided annotation for fine-grained proposition types in argumentation. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 1008–1018.

Chris Kedzie, Kathleen McKeown, and Hal Daumé III. 2018. Content selection in deep learning models of summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1818–1828, Brussels, Belgium. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Deanna Kuhn, Laura Hemberger, and Valerie Khait. 2016. Tracing the development of argumentive writing in a discourse-rich context. *Written Communication*, 33(1):92–121.

Tatsuki Kuribayashi, Hiroki Ouchi, Naoya Inoue, Paul Reisert, Toshinori Miyoshi, Jun Suzuki, and Kentaro Inui. 2019. An empirical study of span representations in argumentation structure parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4691–4698.

I-Ta Lee and Dan Goldwasser. 2019. Multi-relational script learning for discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4214–4226, Florence, Italy. Association for Computational Linguistics.

Jing Li, Aixin Sun, and Shafiq Joty. 2018. SegBot: a generic neural text segmentation model with pointer network. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4166–4172.

Jiwei Li, Minh-Thang Luong, Dan Jurafsky, and Eduard Hovy. 2015. When are tree structures necessary for deep learning of representations? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2304–2314.

Eric Malmi, Daniele Pighin, Sebastian Krause, and Mikhail Kozhevnikov. 2018. Automatic prediction of discourse connectives. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1506–1515.

Keith K Millis, Arthur C Graesser, and Karl Haberlandt. 1993. The impact of connectives on the memory for expository texts. *Applied Cognitive Psychology*, 7(4):317–339.

Shashi Narayan, Claire Gardent, Shay Cohen, and Anastasia Shimorina. 2017. Split and rephrase. In *EMNLP 2017: Conference on Empirical Methods in Natural Language Processing*, pages 617–627.

Huy Nguyen, Wenting Xiong, and Diane Litman. 2016. Instant feedback for increasing the presence of solutions in peer reviews. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 6–10.

Allen Nie, Erin Bennett, and Noah Goodman. 2019. DisSent: Learning sentence representations from explicit discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4497–4510.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4658–4664.

Joonsuk Park and Claire Cardie. 2014. Identifying appropriate support for propositions in online user comments. In *Proceedings of the first workshop on argumentation mining*, pages 29–38.

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.

Dolores Perin and Mark Lauterbach. 2018. Assessing text-based writing of low-skilled college students. *International Journal of Artificial Intelligence in Education*, 28(1):56–78.

Leila Pishdad, Federico Fancellu, Ran Zhang, and Afsaneh Fazly. 2020. How coherent are neural models of coherence? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6126–6138.

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

Attapol Rutherford, Vera Demberg, and Nianwen Xue. 2017. A systematic study of neural discourse models for implicit discourse relation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 281–291.

Attapol Rutherford and Nianwen Xue. 2015. Improving the inference of implicit discourse relations via classifying explicit discourse connectives. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 799–808.

Sebastian Schuster and Christopher D Manning. 2016. Enhanced English universal dependencies: An improved representation for natural language understanding tasks. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2371–2378.

Damien Sileo, Tim Van De Cruys, Camille Pradel, and Philippe Muller. 2019. Mining discourse markers for unsupervised sentence representation learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3477–3486, Minneapolis, Minnesota. Association for Computational Linguistics.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566.

Robert D. Van Valin. 2001. *An introduction to syntax*. Cambridge University Press.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lió, and Yoshua Bengio. 2018. Graph attention networks. In *Sixth International Conference on Learning Representations (ICLR)*.

Yizhong Wang, Sujian Li, and Houfeng Wang. 2017. A two-stage parsing method for text-level discourse analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 184–188.

Yizhong Wang, Sujian Li, and Jingfeng Yang. 2018. Toward fast and accurate neural discourse segmentation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 962–967.

Bonnie Webber and Aravind Joshi. 1998. Anchoring a lexicalized tree adjoining grammar for discourse. In *ACL/COLING Workshop on Discourse Relations and Discourse Markers*, pages 86–92.

David Weir, Julie Weeds, Jeremy Reffin, and Thomas Kober. 2016. Aligning packed dependency trees: a theory of composition for distributional semantics. *Computational Linguistics*, 42(4):727–761.

Aaron Steven White, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2018. Lexicosyntactic inference in neural models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4717–4724.

Michael Wiegand, Marc Schulder, and Josef Ruppenhofer. 2015. Opinion holder and target extraction for verb-based opinion predicates–the problem is not solved. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 148–155.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying relations via long

short term memory networks along shortest dependency paths. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1785–1794.

A Connective distributions in Book-Simpl and DeSSE

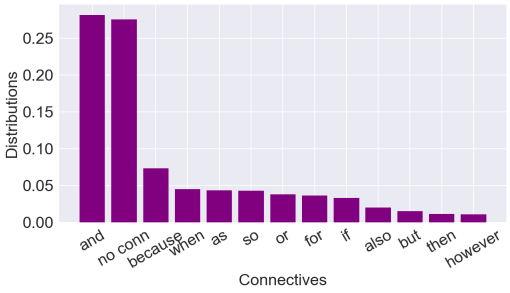

Figure 7: Connective distributions in DeSSE with threshold 1%

Here we present a detailed statistics of connective distributions on DeSSE and Book-Simpl. Figure 7 presents the connectives from DeSSE with distribution above 1%. Among 68 connectives, there are thirteen connectives above the threshold and the rest are low frequency connectives. Table 6 shows the Book-Simpl connective distribution compared to Book corpus. As illustrated, the Book-Simpl shares the same distribution as Book corpus on both Book-Simpl 5 and 8.

B Annotation instruction in DeSSE

Here we present the instructions for annotators, as shown by Figure 8.

Complex sentences are built up from simple ones using connectives like *"and", "or", "but", "however", "because", "when", "until", "if", and relative pronouns like "which", "who", "that"*.
In this HIT, you will see a complex sentence. Please:

1. Segment this sentence into several substrings, and then
2. Modify each substring to make a short, complete, new sentence, leaving out the connectives and relative pronouns.

Please convert the original long sentence into at least 1, at most 5 sub-sentences. (We usually expect 2 or 3 sub-sentences.) Then rewrite the sub-sentences to satisfy ALL these requirements:

1. Each new simple sentence should be a complete (!), readable, grammatical sentence.
2. The content in the simple sentences should add up to the content in the complex sentences, minus all the connectives.
3. Avoid using the words that are not appeared in the original sentence for your rewriting, except the changes from singular to plural, person's name to pronouns.

NOTE: If you get an example that is a simple sentence already, not a complex sentence, just copy it and submit it as is.

Figure 8: Instruction for DeSSE annotation

The instruction illustrate the two phases of annotation. The annotator first chooses whether to add one or more split points to an input sentence, where the word after a split point represents the first word of a new segment. Once an annotator has identified the split points, which happens on the first page of the AMT interface, shown as Figure 9, a second view of the interface appears. Figure 10 shows the second view when annotators rewrite the segments. Every span of words defined by split points (or the original sentence if no split points), appears in its own text entry box for the annotator to rewrite. Annotators cannot submit if they remove all the words from a text entry box. They are instructed to rewrite each text span as a complete sentence, and to leave out the discourse connectives.

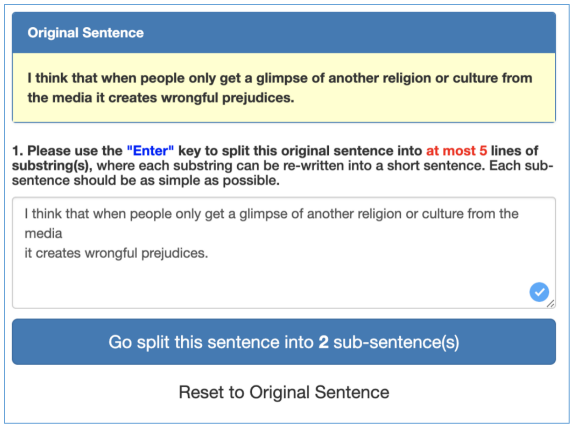

Figure 9: Interface of splitting the sentence

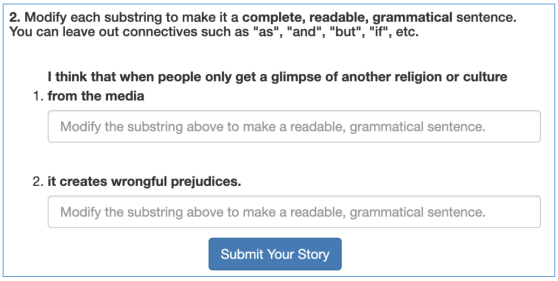

Figure 10: Interface of rewriting the segments from Figure 9 into complete sentences

Several auto-checking and warnings are applied in the interface to reassure the quality. If a rewrite contains a discourse connective, a warning box pops up asking if they should drop the discourse connective before submitting it. A warning box will show up if annotators use vocabulary outside the original sentence. To prevent annotators from failing to rewrite, we monitored the output, checking for cases where they submitted the text spans with no rewriting. Annotators are prohibited to submit if the interface detects an empty rewrite box or

Corpus	Size	and	but	because	if	when	before	though	so
Book 5	3054K	31	32	15	5	16	*na*	*na*	*na*
Book-Simpl 5	285k	33	28	8	5	27	*na*	*na*	*na*
Book 8	3435K	28	28	5	13	14	6	3	2
Book-Simpl 8	359K	29	24	4	7	23	8	3	1

Table 6: Number of sentence pairs (Size), and the distribution of connectives (as percentages) for the original Book corpus and our modified version.

the total lengths of the rewrites are too short compared to the source sentence. We warned annotators by email that if they failed to produce complete sentences in the rewrite boxes, they would be blocked. Some annotators were blocked, but most responded positively to the warnings.

C Quality control in DeSSE

To test the clarity of instruction and interface, the initial 500 sentences were used for evaluating the task quality, each labeled by three turkers (73 turkers overall), using three measures of consistency, all in [0,1]. Average pairwise boundary similarity (Fournier, 2013), a very conservative measure of whether annotators produce the same number of segments with boundaries at nearly the same locations, was 0.55. Percent agreement on number of output substrings was 0.80. On annotations with the same number of segments, we measured the average Jaccard score (ratio of set intersection to set union) of words in segments from different annotators, which was 0.88, and words from rephrasings, which was 0.73. With all metrics close to 1, and boundary similarity above 0.5, we concluded quality was already high. During the actual data collection, quality was higher because we monitored quality on daily basis and communicated with turkers who had questions.

D Experiment Settings

All the methods take GloVe word embeddings as input. Due to the size difference between Book-Simpl and DeSSE, we use different dimensionalities for the word embeddings (w) and classifier hidden layers (h) with the two corpora on all baseline systems: for Book-Simpl, $D_w = 300, D_h = 512$; for DeSSe $D_w = 100, D_h = 256$. We train DisSent using the original $D_h = 4096$ for Book-Simpl, and reduce it to 256 for DeSSE. Apart from this one change to DisSent, we use the published settings for all baseline systems. We train DAnCE using the same vector dimensions as for DisSent. Because DAnCE has twice the number of parameters as DisSent, we use the smaller classifier dimensionality of 256 on both corpora.

We use SGD as optimizer for DAnCE, with the learning rate at 0.01. Learning rates between [0.1,0.001] did not show obvious performance differences, and 0.01 converged faster. We use early-stopping to prevent overfitting. We did not use dropout, due to a negative impact on performance (cf. (Nie et al., 2019)).

All training is done on 4 Nvidia RTX 2080 Ti GPUs. The longest training time is 35 hours, for DAnCE on Book-Simpl 8. During testing, we perform 16 bootstrap iterations