Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference “Dialogue 2019”

Moscow, May 29—June 1, 2019

LEARNING MULTI-PARTY DISCOURSE STRUCTURE USING WEAK SUPERVISION

Badene S. (sonia.badene@irit.fr),
Thompson K. (catherine.thompson@irit.fr),
Lorré J-P. (jplorre@linagora.com),
Asher N. (asher@irit.fr)

Discourse structures provide a way to extract deep semantic information from text, e.g., about relations conveying causal and temporal information and topical organization, which can be gainfully employed in NLP tasks such as summarization, document classification, sentiment analysis. But the task of automatically learning discourse structures is difficult: the relations that make up the structures are very sparse relative to the number of possible semantic connections that could be made between any two segments within a text; furthermore, the existence of a relation between two segments depends not only on “local” features of the segments, but also on “global” contextual information, including which relations have already been instantiated in the text and where. It is natural to try to leverage the power of deep learning methods to learn the complex representations discourse structures require. However, deep learning methods demand a large amount of labeled data, which becomes prohibitively expensive in the case of expertly-annotated discourse corpora. One recent advance in the resolution of this “training data bottleneck”, data programming, allows for the implementation of expert knowledge in weak supervision system for data labeling. In this article, we present the results of our application of the data programming paradigm to the problem of discourse structure learning for multi-party dialogues.

Key words: Discourse Structure, Discursive relations, Weak Supervision, Attachment, Data Programming

ПРЕДСКАЗАНИЕ СТРУКТУРЫ МНОГОСТОРОННЕГО ДИСКУРСА С ПОМОЩЬЮ WEAK SUPERVISION

Баден С. (sonia.badene@irit.fr),
Томпсон К. (catherine.thompson@irit.fr),
Лорре Ж. П. (jplorre@linagora.com),
Ашер Н. (asher@irit.fr)

Ключевые слова: структура дискурса, дискурсивные отношения, weak supervision, связь, data programming

1. Introduction

Discourse structures are relational structures composed of *discourse units* (DUs), or instances of propositional content, and binary coherence relations over them, conveying semantic (causal, temporal) and presentational (thematic, argumentative) information expressed by a text. We represent such structures as graphs containing a set of nodes representing the DUs and a set of labelled arcs representing the coherence relations. In the case of dialogues occurring between multiple interlocutors, extraction of their internal discourse structures can provide useful semantic information to "downstream" models used, for example, in the production of intelligent meeting managers or the analysis of user interactions in online fora.

Such representational schemes serve as annotation models of which discourse theorists have proposed several: (RST[1] [7], LDM[2] [11], Graphbank [15], DLTAG [5], PDTB[3] [12] and SDRT[4] [4]). Much of computational discourse-analysis is based on RST, which supposes that discourse structures are trees. However, this assumption becomes very difficult to maintain for dialogue [4] and even more unnatural for multi-party dialogue, which presents many examples of non-treelike structures [1] like the one shown in the **Figure 1**.

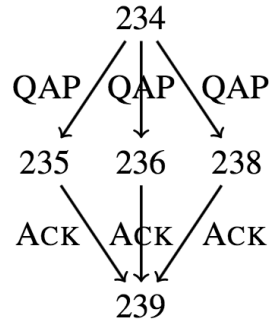

Figure 1: Interlocutors in segments 235, 236 and 238 all respond to a question asked at 234 (linked via edges labelled as Question-Answer Pair), and the interlocutor at segment 239 responds to each answer (linked with Acknowledgement)

Such examples motivate the SDRT annotation model, in which discourse structures are not assumed to be trees but rather directed acyclical graphs (DAGs) [2–4]. SDRT was used to annotate the STAC corpus[5], which is the corpus on which we trained

[1] Rhetorical Structure Theory

[2] Linguistic Discourse Model

[3] The Penn Discourse Treebank

[4] Segmented Discourse Representation Theory

[5] link to STAC corpus: https://www.irit.fr/STAC/index.html

a supervised deep learning model and a weakly supervised model in the discourse structure learning task in order to then compare them.

The data programming paradigm was introduced by Hazy Research in 2016 [14] along with a framework Snorkel [13] for using distant, disparate knowledge sources to apply noisy labels to large data sets, and then using those labels to train classic data-hungry machine learning (ML) algorithms. The data programming paradigm allows us to unify these noisy labels in order to generate a probability distribution for all labels for each data point. This set of probabilities replaces the ground-truth labels in a standard discriminative model outfitted with a noise-aware loss function and trained on a sufficiently large data set.

In this study the structure learning problem is restrained to predicting attachment between DU pairs. After training a supervised deep learning algorithm to predict attachments on the full STAC corpus, we then constructed for comparison a weakly supervised learning system in which we used 10% of the corpus as a development set. SDRT experts who annotated the corpus wrote a set of Labeling Functions (LFs) and tested them against this development set. We treated the remainder of the corpus as raw/unannotated data. After applying the LFs to the unannotated data and obtaining the results from the generative and discriminative models, we found that we gained in accuracy over ten points with respect to the supervised method. When the generative model was used in stand alone fashion, we gained almost 30 points in F1 score over the supervised method, which uses a state of the art deep learning model. When we think of the time it takes experts to hand-annotate dialogues, this means that the generative model and weak supervision may be far preferable to straight deep learning methods, in at least some cases.

2. Attachment Prediction: State of the Art

A discourse structure in SDRT is defined as a graph $\langle V, E_1, E_2, \ell, \text{Last}\rangle$, where: V is a set of nodes or discourse units (DUs) and $E_1 \subseteq V^2$ is a set of edges between DUs representing coherence relations. $E_2 \subseteq V^2$ represents a dependency relation between CDUs[6] and their constituent DUs. $\ell: E_1 \rightarrow R$ is a labeling function that gives the semantic type (an element of R) of an edge in E_1, and *Last* is a designated element of V giving the last DU relative to textual or temporal order.

The process of learning an SDRT structure for a dialogue or text has three natural steps:

1. Segment the text into DUs
2. Predict the attachments between DUs, i.e. identify the elements in E_1 and E_2
3. Predict the semantic type of the edge in E_1

In this paper, we focus on the second step, attachment prediction, the goal of which is to determine a substructure, $\langle V, E_1, E_2, \text{Last}\rangle$, of the complete discourse

[6] SDRT also allows for Complex Discourse Units (CDUs), which are clusters of two or more DUs which can be connected as an ensemble to other DUs in the graph. The CDUs which are themselves graphs that can serve as arguments of coherence relations.

graph. This is a difficult problem for automatic processing: attachments are theoretically possible between any two DUs in a dialogue or text, and often graphs include long-distance relations. The attachment problem as we have stated it is endemic to SDRT and theories that posit dependency structures for discourse. In RST the problem of attachment is less clear (see [8]).

[Muller et al. 2012] [9] is one of the first papers we know of on discourse parsing that targets the attachment problem. It targets a restricted version of an SDRT graph, a dependency tree in which CDUs are eliminated by a "flattening" strategy similar to the one we use below. This means that the target representations are of the form $\langle V^*, E^*_1, \text{Last}\rangle$, where V^* is V with the CDUs removed and E^*_1 results from running the flattening strategy on E_1. It trains a simple MaxEnt algorithm to produce structures probability distributions over pairs of elementary discourse units and exploits then global decoding constraints to produce the targeted structures. It reports F1 scores of 66.2% with the A* algorithm.

In terms of discourse structure prediction for multi-party dialogues, Perret et al. (2016) [10] targets a more elaborate approximation of the SDRT graphs: DAGs in which CDUs are eliminated from V and the relations on CDUs are distributed over their constituents. As with [9], this requires another reworking of E_1. Perret et al. then uses Integer Linear Programming (ILP) to encode both the objective function and global decoding constraints over local scores on multi-party dialogues and achieves an F-measure of 0.689 on unlabelled attachment.

3. The STAC Annotated Corpus

3.1. Overview

STAC is a corpus which captures strategic conversation between the players of an online version of the game Settlers of Catan. The full corpus contains 45 games, each of which is divided into an average of 57 dialogues. A dialogue begins at the beginning of a player's turn, and ends at the end of that player's turn. During the interim, players can bargain with each other or make spontaneous conversation. These player utterances are "linguistic" turns, whereas "announcements" made by the game Server regarding the game state or a certain player status are "non-linguistic" turns. Both types of turns are segmented into discourse units (DUs), and these units are then connected by a semantic relations of one of the 17 types admitted by SDRT. As a result, each dialogue contains a weakly connected DAG which is its discourse structure.

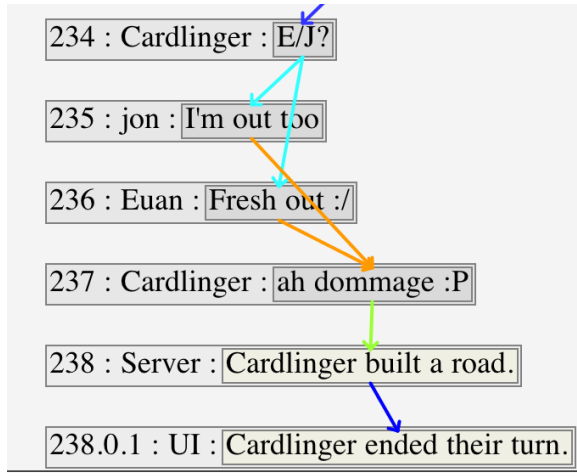

Figure 2: Excerpt of a STAC dialogue illustrating relations like Sequence (dark blue), Result (green), linguistic turns spoken by players and non-linguistic turns emitted by "Server" or "UI"

3.2. Data Preparation

The full STAC corpus includes 2,593 dialogues (or discourse structures), 12,588 linguistic DUs, 31,811 non-linguistic DUs and 31,251 semantic relations.

As discussed above, our task is to predict, for each dialogue, for each possible pair of DUs, whether the DUs are connected by a semantic relation, an operation which eventually yields a discourse structure for the dialogue. Before beginning our experiments, we implemented the following simplifying measures:

1. Roughly 56% of the total dialogues contain only non-linguistic DUs. These represent player turns in which no players bargain or chat with one another. The annotations in these dialogues are fairly regular given the purely mechanical succession of DUs, and are much less difficult and less interesting from a discourse analysis perspective. For this reason we ignore these non-linguistic-only dialogues for our prediction task.

2. In the corpus, shorter relations are more frequent than long-distance relations such that 67% of relations occur between adjacent DUs, and 98% of relations have a distance of 10 or less. (A relation of distance 10 stretches over 9 DUs between the source and the target DU.) In order to avoid a combinatory explosion of possible DU pairs, we restrict the relations we consider to a distance of 10 or less.

3. Out of the 17 possible relation types allowed by SDRT, we consider only the 4 most frequent: Question-answer-pair, Sequence (temporal), Result (causal), Continuation (thematic continuity). We retain about 70% of the

total relations. The reason for this choice will become apparent in following detailed discussion of labeling functions.

4. Because we are here only interested in predicting attachment between single DUs, following [9, 10] we "flatten" the CDUs by connecting all relations incoming or outgoing from a CDU to the "head" of the CDU, or its first DU. This results in shifts in the source and/or target DUs for about 40% of the relations.

5. In order to reduce run-time for each rule during development, we created "sandbox" sets for each relation type: smaller versions of the development set which ignored all candidate pairs except those which could possibly be attached by the relation type in question. We have a sandbox data set for the rules pertinent to a particular discourse relation and a larger sandbox data set for the rules for the four discourse relations that we examined.

After the above preparation, the STAC corpus as we use it in our learning experiments includes 1,130 dialogues, 12,509 linguistic DUs, 18,576 non-linguistic DUs and 22,098 semantic relations. (Here again we are only considering the 4 relations Question-answer-pair, Sequence, Result and Continuation.)

4. The Data Programming Pipeline: Experiments

4.1. Candidates and Labeling Functions

In constructing our weak supervision system, we took inspiration from the Snorkel implementation[7] of the data programming paradigm. The first step in the Snorkel pipeline is candidate extraction, followed by LF creation. Candidates are the units of data from which labels will be predicted; LFs are the simple expert-composed functions which will predict a label for each candidate. The prototypical Snorkel task is to predict whether there is a certain type of relation between two entities in a sentence within a text: candidates are pairs of entities extracted from sentences, and LFs are written using contextual information at the sentence level.

In the case of dialogue attachment prediction, we needed to find a way to give our LFs access to contextual information from the entire dialogue which they could apply to each candidate, or pair of DUs within a dialogue. We did this by fixing the order in which each LF would "see" the candidates such that it would consider adjacent DUs before distant DUs, and thus the LF would know its current position in a dialogue. We also allowed LFs to keep track of previously predicted relations to give them some information about dialogue history. Other information leveraged by the LFs included the DU raw text, speaker identities, the DU dialogue acts, DU types (linguistic or non-linguistic) and the distance between DUs.

[7] https://hazyresearch.github.io/snorkel/

```python
def LF_Result_L_L_case1(row):
    l=0
    if (any(x in row.target_text.lower() for x in resultWords)
                or any(x in row.source_text.lower() for x in resultWords)):
            l=1
    return l


def LF_Result_L_L_case2(row):
    l = 0
    if row.source_surface_act in ["Question", "Request", "Assertion"] \
    and (row.target_dialogue_act in ["Offer", "Counteroffer"] \
         or row.source_emitter == row.target_text.partition(' ')[0] or row.target_surface_act == "Request"):
        l=1
    return l

def LF_Result_L_L_case3(row):
    l = 0
    if row.source_emitter == row.target_emitter and row.source_addressee == row.target_addressee \
    and row.target_dialogue_act == "Counteroffer":
        l=1
    return l


def LF_Result_L_L_case4(row):
    l = 0
    if row.source_dialogue_act == "Counteroffer" and row.target_dialogue_act == "Refusal":
        l=1
    return l
```

Figure 3: *Result* connects a cause to its effect, i.e., the main eventuality of the first argument is understood to cause the eventuality given by the second. Here we show a sample of our rules written in python for the relation ***Result*** connecting two linguistic discourse units

As we are at present concerned only with predicting attachments, each LF returns a 1, a 0 or a −1 ("attached"/"do not know"/"not-attached") for each candidate. However, each of our LFs is written and evaluated with a specific relation type *Result*, *QAP*, *Continuation* and *Sequence* in mind. In this way, LFs also leverage a kind of type-related information. This makes sense from an empirical perspective as well as an epistemological one: an attachment decision concerning two DUs is tightly linked to the type of relation relating the DUs, and so when an annotator decides that two DUs are attached, he or she does so with some knowledge of what type of relation attaches them. Figure 3 shows a sample of our rules used for attachment prediction with the *Result* relation in mind.

4.2. The Generative Model

Once we have applied the LFs to all the candidates, we then move to the generative step. In Snorkel, the generative model unifies the results of the LFs, which is a matrix of labels given by each LF (columns) for each candidate (rows). Though the simplest approach to unification would be to take the majority vote among the LFs for each candidate, this is less effective in cases where all the LFs abstain or give "0". Further, this approach would not take into account the individual performances of the LFs. And so the generative model as specified in (1) provides a general distribution of marginal probabilities relative to n accuracy dependencies $\phi_j(\Lambda_i; y_i)$ between an LF λ_j and true labels y_i that depend on parameters theta θ_i.

$$p_\theta(\Lambda, Y) \propto exp(\Sigma_{i=1}^{m}\Sigma_{j=1}^{n}\theta_j\phi_j(\Lambda_i, y_i)) \quad (1)$$

The parameters are estimated without access to the ground truth labels by minimizing the negative log marginal likelihood of the output of an observed matrix Λ as in (2).

$$argmin_\theta - log\Sigma_Y(p_theta(\overline{\Lambda}, Y) \quad (2)$$

This objective is optimized by interleaving stochastic gradient descent steps with Gibbs sampling ones. The model thus uses the accuracy measures for the LFs in (1) to assign marginal probabilities that two DUs are attached to each candidate. In this model, the true class labels y_i are latent variables that generate the labeling function outputs, which are estimated via Gibbs sampling.

This calculation presupposes that the LFs are independent. However, the LFs are often dependent: one might be a variation of another or they might depend on a common source of distant supervision. If we don't take this into account, we risk assigning incorrect accuracies to the LFs. Getting users to indicate dependencies by hand, however, is difficult and error-prone. The generative model in Snorkel comes with the option of automatically selecting which dependencies to model without access to ground truth. It uses a pseudo-likelihood estimator, which does not require any sampling or other approximations to compute the objective gradient exactly. It is much faster than maximum likelihood estimation, because the estimator relies on a hyper-parameter ∈ that trades off between predictive performance and computational cost. With large values of ∈ no correlations are included and as it reduces the value progressively more correlations are added, starting with the strongest.

4.3. Discriminative Model

While the generative model outputs the marginal probabilities for each of the labels for each candidate, the discriminative model generalizes this output and augments the coverage of the LFs. While this may lead to a small reduction in precision, it is in exchange for a boost in recall.

We used BERT's [6] sequence classification model (source code on the link bellow[8]) with 10 training epochs and all default parameters otherwise. BERT, the Bidirectional Encoder Representations from Transformers, is a text encoder pre-trained using language models where the system has to guess a missing word or word piece that is removed at random from the text. Originally designed for automatic translation tasks, BERT uses bi-directional self-attention to produce the encodings and performs at the state of the art on many textual classification tasks. While in principle we could have used any discriminative model, as is suggested in the Snorkel literature, BERT gave us by far the best results on attachment prediction. For this reason we also used BERT as our model for supervised learning of attachment to compare its results with those of the weak supervision method.

[8] Link to BERT sequence classification model code: https://github.com/huggingface/pytorch-pretrained-BERT/blob/master/examples/run_classifier.py

5. Results and Analysis

In order to write a set of LFs/rules which adequately covered the data, we had to find a way to reasonably divide and conquer the myriad characteristics of the relations. We started by focusing on relation type: for each of the four most frequent relation types, we wrote a separate rule for each of the sets of endpoint types most prevalent for that relation. Result (RES) is the only relation type which was found between all four endpoint permutations: LL (linguistic source-linguistic target), LNL (linguistic source, non-linguistic target), etc. We used the relation behavior observed in our development/sandbox sets to write and revise the rules. The development set consisted of 3 games (10% of our data). The table 1 shows the performance of each rule on its own "sandbox" development set.

Table 1: Number of true positives, true negatives, false positives, false negatives and accuracy score for each LF when applied to the "sandbox" candidates from the STAC data

	TP	TN	FP	FN	Accuracy
QAP LL	294	1798	112	138	0.89
QAP NLNL	84	187	0	0	1.00
RES NLNL	739	2,929	13	55	0.98
RES LNL	13	2158	93	97	0.91
RES LL	25	316	19	37	0.85
RES NLL	2	139	0	2	0.98
Cont LL	16	9,818	110	106	0.97
Cont NLNL	613	3,254	0	1	0.99
SEQ NLL	90	658	2	14	0.97
SEQ NLNL	236	1,220	10	76	0.94

Table 2: Evaluations of the combination of the four LFs (QRSC)' attachment with the weakly supervised and supervised approaches on the sandbox data set

	Generative Model			Discriminative Model on Test	
	Dev	Train	Test	with Marginals	with Gold annotations
Precision	0.67	0.70	0.68	0.45	0.61
Recall	0.84	0.85	0.84	0.54	0.53
F1 score	0.75	0.77	0.75	0.49	0.57
Accuracy	0.92	0.93	0.92	0.84	0.88

We first evaluated the LFs for each discourse relation type individually on the development corpus, providing a measure of their coverage and accuracy on a subset of the data. Then we evaluated the generative model on the combination of the four LF types and the discriminative model on the test set of the corpus. The table 2 presents the results at the end of each step in our weak supervision system. To compare the two approaches, the discriminative model was first trained on the marginals provided

by our generative model, then on the "gold" annotations, which are manual annotations of the Stac corpus.

We find the results of the generative model striking as they are almost 20 points higher in F1 score concerning positive attachment over the discriminative model trained on the gold annotations. This shows the power of the rule based approach even when compared to a state of the art deep learning system. Another interesting point is that the discriminative model can still perform acceptably with the marginals data compared to its performance using the gold annotations; its accuracy is only 4 points lower and its F1 score is 8 points lower but still comparable with results in the literature, showing that the generative model has information to offer that the discriminative model can exploit well—thus opening up the possibility for transfer learning. Rather than naively treat these noisy training labels as ground truth, our noise-aware discriminative model gives a slight improvement in recall with a decrease in precision compared to the supervised approach. With respect to the individual LFs in isolation, we find that, apart from *QAP*, our rules for each relation type have an accuracy, precision and recall comparable to those for the supervised models. One reason for our lower precision for *QAP* may be attributable to a deficiency in the flattening procedure the effects *QAP* more frequently; in some cases the flattening algorithm re-attaches the *QAP* relation to a CDU head which was not in fact the component of the CDU which marked the question. What is interesting is the synergy between the rules such that when they all interact on the test data, they do very well on the generative model.

6. Conclusions and Future Work

Having chosen a single discriminative model for all our experiments, we were able to compare our weak supervision approach, in which we leveraged parts of the Snorkel system, with that of a standard supervised model on the difficult task of discursive attachment. Our approach allows us to model the discourse more precisely and to be generalized to other corpora. In contrast to a supervised algorithm, our results on the generative model are almost 30 points higher without covering all types of rules. In addition, we generate a lot of annotated data in a very short time.

In future work, we plan to enrich our weak supervision system by first covering all 17 types of SDRT relations on all data. We also plan to give LFs access to more sophisticated context which will take into account sequence-constraints in the attachments of the complete conversation and global structuring constraints. We will at that point be in a position to evaluate the structure of the global discourse by using our structured predictions as inputs to a maximum spanning tree (MST) for example. And in a broader scope, our experiments with this paradigm may suggest possible lines of inquiry into how weakly supervised methods might effectively capture the global structural constraints on discourse structures without decoding or elaborate learning architectures.

References

1. *Afantenos, S. et al.:* Discourse parsing for multi-party chat dialogues. Presented at the (2015).
2. *Asher, N.:* Reference to abstract objects in discourse. Springer Science & Business Media (1993).
3. *Asher, N. et al.:* Discourse structure and dialogue acts in multiparty dialogue: The stac corpus. In: LREC. (2016).
4. *Asher, N., Lascarides, A.:* Logics of conversation. Cambridge University Press (2003).
5. *Creswell, C. et al.:* Penn discourse treebank: Building a large scale annotated corpus encoding dltag-based discourse structure and discourse relations. Manuscript [fix this]. (2003).
6. *Devlin, J. et al.:* Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. (2018).
7. *Mann, W. C., Thompson, S. A.:* Rhetorical structure theory: Description and construction of text structures. In: Natural language generation. pp. 85–95 Springer (1987).
8. *Morey, M. et al.:* A dependency perspective on rst discourse parsing and evaluation. Computational Linguistics. 198–235 (2018).
9. *Muller, P. et al.:* Constrained decoding for text-level discourse parsing. Proceedings of COLING 2012. 1883–1900 (2012).
10. *Perret, J. et al.:* Integer linear programming for discourse parsing. In: Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: Human language technologies. pp. 99–109 (2016).
11. *Polanyi, L. et al.:* A rule based approach to discourse parsing. In: Proceedings of the 5th sigdial workshop on discourse and dialogue at hlt-naacl 2004. (2004).
12. *Prasad, R. et al.:* The penn discourse treebank 2.0. Annotation manual. The pdtb research group, (2007).
13. *Ratner, A. et al.:* Snorkel: Rapid training data creation with weak supervision. Proceedings of the VLDB Endowment. 11, 3, 269–282 (2017).
14. *Ratner, A. J. et al.:* Data programming: Creating large training sets, quickly. In: Advances in neural information processing systems. pp. 3567–3575 (2016).
15. *Wolf, F. et al.:* The discourse graphbank: A database of texts annotated with coherence relations. Linguistic Data Consortium. (2005).