Zwitscher
Argument Span Labeling for German Discourse Connectives

Arkadi Schelling*

August 14, 2015

Contents

1 Introduction **2**

2 PDTB-style Discourse Parsing **2**
2.1 The Penn Discourse Tree Bank 2
2.2 The Potsdam Commentary Corpus 3
2.3 The DiMLex 3
2.4 Lin et al. 3

3 A Discourse Parser for German **5**
3.1 Adapting Lin et al.'s approach 5
3.2 Evaluation and Possible Improvements 6
3.3 Further Steps 7

References **8**

*Student of the M. Sc. "Cognitive Systems" at the Univ. of Potsdam, Stud.-ID 779135, arkadi.schelling@gmail.com

1 Introduction

> Discourses are tactical elements or blocks operating in the field of force relations; there can exist different and even contradictory discourses within the same strategy; they can, on the contrary, circulate without changing their form from one strategy to another, opposing strategy.
> Michel Foucault - History of Sexualiy, vol. I, pp. 101–102.

Foucault's discourse theory is notoriously hard to grasp, as computational linguists we therefore restrict ourselves to a less ambitious and extensive theory of discourse. Still, discourse theory in computational linguistics also tries to formalise the idea of a discourse made up of blocks that are operating in some relations to each other. In this report we are focussing on German discourse connectives and their argument spans. For example the following sentence shows the underlined connective "Und" connecting a bold internal argument with a an italicized external arguement.

> *Das Land hat künftig zu wenig Arbeit für zu viele Pädagogen.* Und **die Zeit drängt.**
> Potsdam Commentary Corpus (maz-000001-10)

This report first gives a short overview of the Penn Discourse Tree Bank and Potsdam Commentary Corpus, as well as the first end-to-end discourse parser for the PDTB by [LNK12]. In the second part the scope of the project is explained and the main details of the used architecture and algorithms are layed out. The last part concludes with evaluation of the approach and further steps on the way towards an end-to-end Discourse Parser for German Twitter data.

2 PDTB-style Discourse Parsing

In light of the pragmatical constraints that are imposed when using algorithmical analysis for labeling blocks and discourse relations, two main theories have evolved. Apart from the tree structured Rhetorical Structure Theory, there is work based on the Penn Discourse Tree Bank.

2.1 The Penn Discourse Tree Bank

The Penn Discourse Tree Bank 2.0 (PDTB) is adding discourse information to all the Wall Street Journal texts of the syntactical annotated Penn Tree Bank. Based on this available data, since 2008 several papers have been published to solve separate tasks connected to discourse parsing. The first paper to create an end-to-end discourse parser has been [LNK12]. Since Discourse Parsing is still far from being solved, it seems a wise choice to base the current research on texts for which many NLP tasks have already been solved. The downside to this is a bad generalization to texts outside a financial scope.

The PDTB labels each discourse connective with a relation and two argument spans. The discourse connective itself is part of one of the arguments, the internal argument, also called Arg1. The other argument is called external argument or Arg2. Furthermore, the PDTB does not only label explicit connectives that have a fixed token representation, but also implicit connectives and several further relations [PDT]. The discourse relations labeled by the PDTB are organised in a three layered tree structure.

2.2 The Potsdam Commentary Corpus

The Potsdam Commentary Corpus 2.0 (PCC) is a collection of German newspaper texts, that have been annotated with syntax trees and several discourse information [?]. Apart from nominal coreference and RST information, it also includes an annotation of connectives and arguments, that is similar to the PDTB.

The most important differences between the PCC and the PDTB are the following:

- The PCC does not annotate neither implicit connectives nor any of the other relations, that are annotated in the PDTB.
- The PCC also does not label attribution spans.
- In the PCC discontinuous connectives (e. g. "entweder ... oder") belong to both argument spans and further these argument spans are overlapping with the external argument (connected to "entweder") being a superset of the internal argument (connected to "oder")

Parsing the PCC's XML file structure was not an easy task, even more since some of the annotations are contradictory. It was possible to parse 1081 connectives from 171 discourses, where one discourse consists of one newspaper text. From these connectives 96.6% are continuous and 96.7% of the continuous connectives consist of a single word. None of the total 37 discontinuous connectives extends over sentence boundaries.

2.3 The DiMLex

Another corpus that can be used for alalysing German connectives is the Discourse Markers Lexicon (DiMLex). This lexicon combines orthographical, syntactical and semantical information of nearly all German connectives.

Even though it is still work in progress, the orthography variants and canonical spelling are useful to cluster different spellings of connectives together. One further complete information is the disambiguity of the connective, which can be used to check whether a token disambiguously denotes a connective or can as well be used as a non-connective word.

The PCC and DiMLex are not completely compatible. They are using different and also slightly inconsistent or incomplete relations. Also, even though the DiMLex should be close to complete, there was a list of over 40 PCC connectives, that could not be related to a DiMLex connective, either due to a mislabeling in the PCC, a gap in the DiMLex or a different notion of connectives.

2.4 Lin et al.

This subsection deals with the Lin et al's paper [LNK12], which is the first end-to-end discourse parser for PDTB-style connectives. Disregarding the few differences between PDTB and PCC that were pointed out in subsection 2.2, we are aiming to follow the paper to create an end-to-end parser for German with the PCC as training data.

The general pipeline of Lin et al's parser is as follows:

1. Connective classifier – Find possible explicit connectives in a text and disambiguate them.
2. Argument span labeler – Find the internal and external arguments of the connectives:
    (a) Argument position classifier – Find the sentence, where the external argument is located.
    (b) Argument extractor – Label the spans of the arguments, especially when both are within the same sentence.
3. Explicit classifier – Classify the relations of the explicit connectives.
4. Non-explicit classifier – Add non-explicit connectives between sentences that are not explicitly connected and classifiy their relations.
5. Attribution span labeler – Find attribution spans in all labeled connectives.

The aimed at German parser focusses on the first two points, as the PCC's and DiMLex' relation information are not reliable and the PCC has neither non-explicit connectives, nor attribution spans. Lin et al are giving detailed information about the set of features that they were using for all stages. Of interest for this report are mainly both substeps of the second step.

The argument position classifier is straight forward. Since more than 99.9% of the arguments are either in the same sentence (SS) or in the previous sentence (PS) of the connective, this classifier only has to disambiguate between two values. The list of features is a mixture of connective strings, the position within the sentence as well as part of speech tags of the previous words. The paper does not specify what kind of classifier is used.

In the PS case both arguments are just labeled as the full sentence. Ignoring improvements on that easy task, most of the paper is dedicated to the task of extracting the argument spans in the SS case. To deal with the four cases of the Arg1 preceding/succeeding Arg2 or one of them being embedded in the other, Lin et al. are inspired to a tree subtraction algorithm [LNK12][p. 17]. For this, they are labeling the closest parent node, that spans the full argument. On this data they are training two maximum entropy classifiers to predict the probability of a node to be either the internal or external argument node. With these classifiers they pick the two nodes with the highest probabilities to be the argument nodes of a connective. The features they are using are a collection of string, syntactic category of the connective and a number of constituency tree features.

Even though they are not explicitly writing it, the used features suggest that Lin et al. are not dealing with connectives that consist of multiple words, esp. not with discontinuous connectives.

In the result, they reach a very good F-score[1] of 97.94% to identify the sentence positions of the arguments. Furthermore, they got a reasonable F-scores of 86.63% for identifying the external argument node and 93.41% for the internal argument node, which was calculated without error propagation from stage 1 and with gold standard constituency trees. The overall performance was measured with a partial match of the nouns and verbs and with a perfect match metric. The resulting F-scores without error propagation and gold standard syntax parses were 86.67% (partial) and 59.15% (perfect) for the external argument and 99.13%/82.23% for the internal argument.

3 A Discourse Parser for German

This section explains how the approach of Lin et al. has been adapted to fit into a pipeline to parse discourse from German Tweets. The code and documentation of the code can be found on `https://github.com/arksch/zwitscher`.

3.1 Adapting Lin et al.'s approach

Following the first analysis of Lin et al. regarding the sentence distances between the arguments, it could be shown that about 1% of the arguments are not of the case SS or PS. This might be due to bad sentence boundaries because of abbreviations, but still it seems that the PCC labels longer distances than the PDTB.

The current code architecture allows an easy implementation of new features. The default features used for argument position labeling are:

- connective canonical orthography by DiMLex
- length of previous sentence
- length of the same sentence
- length of next sentence
- words in the connective
- numbe of tokens inside the sentence before the connective
- second previous token
- previous token
- next token

Note that due to time reasons this step does not yet use any part of speech tags, even though they are highly promising to improve the performance. Another feature that could be tried due to Lin et al. is the exact spelling of the connective, e. g. a capital "Und" will always have its external argument in the previous sentence.

The next important step that is not explained by Lin et al. is the creation of gold data for the classification of nodes as argument nodes. This gold data

[1]Whatever an F-score means for a regression problem has to stay a secret of Lin et al...

is created by picking the first parent node that spans the complete argument as the argument node. Again, I did not find time to evaluate the accuracy of this method.

The features to attach argument node probabilites to the nodes are the following:

- connective canonical orthography by DiMLex
- number of node siblings
- node category
- previous node category on the path to the connective

Many technical problems have been solved to combine the syntax trees with the connective information. For all of these features only the first word of the connective is taken. As 96.7% of the connectives consits of one word, this should not influence the results too much. Still, the calculation of features takes a long time. Therefore, the following features have been implemented but remain unused:

- relative position of the connective within the nodes terminals
- left and right siblings of the connective

Lin et al. further use the complete path including directions up and down from the node to the connective and the connectives syntactic category (subordinating, coordinating or adverbial). Especially the combination of these two seems to promising, as many nodes are not on the direct path to the root.

A logistic regression classifier is then able to attach a probability to each node. Lin et al. are not explaining how they solve the conflict, when internal and external argument node are identical. Currently, my approach only deals with this in the next step.

Once the argument nodes are picked, the argument spans have to be labeled. The first labeling step labels all children of the nodes as the respective arguments. The tree subtraction is calculated if one of the spans is a subset of the other as the complementary set. If the argument nodes are identical, this will lead to an empty set. In any case the connective is added to the internal argument. Instead of an empty external argument the original external argument without tree subtraction is picked.

3.2 Evaluation and Possible Improvements

This subsection gives a few evaluation results and tries to investigate why the argument span labeling is performing very poorly. Many points are asking for improvement with an optimistic "not yet".

The baseline for picking the correct sentence for the external argument would be a majority classifier of 57% in the same sentence. A 5-fold cross validation shows an accuracy of 91%, thus immensely improving over the baseline. This result can surely be improved by taking part of speech tags into account. Afterwards it might be closer to Lin et al's "F-score" of 97.94%.

The results of the argument span labeling are very poor. A baseline of just labeling the full sentence as an argument has not yet been calculated and might even outperform our current approach. I did implement neither the partial metric of Lin et al. as the syntactic information is lacking, nor the perfect match metric, as it would be too bad to show any possible improvements. Instead, the implemented metrics are:

1. a percentage of overlap between gold standard and predicted argument spans
2. the micro F1-score of the argument spans, i. e. the global mean of the F1-scores for each argument span.

Note, that in contrast to Lin et al. this is not yet disregarding punctuation. The resulting scores on a held out dataset of 20% are:

	Internal argument	External argument
overlap percentage	24.2	28.3
micro F1	22.9	25.1

Hopefully, these results can be boosted by picking more meaningful features. Note, that for a meaningful evaluation, this would also require to use a triple or nested cross validation, which is not yet implemented in the code.

A first investigation of a dozen connectives suggests that actually many argument nodes are picked correctly. This could mean that:

1. either the algorithm to create the gold argument nodes is not working correctly
2. or that tree subtraction is not a good chocie for the PCC
3. or that this approach does in general not work well for German general newspaper texts.

A step for improvement could be to calculate the scores when using the argument nodes.

The complete pipeline has not yet been evaluated, as it can be expected to perform even worse than the argument span labeling.

A last part for improvement might be the correction of the PCC and DiMLex.

3.3 Further Steps

In this subsection further steps towards an end-to-end Discourse Parser for German Twitter data are explained.

One fellow project is dealing with picking possible connectives from German Tweets and disambiguating them. This output could be passed on to the command line script of my project. Furthermore, a module to solve the very difficult task of cleaning, chunking and parsing POS-tags from Tweets is already written. Still, to my knowledge a module to parse constituence trees is still lacking. An out of the box parser trained on other data will very likely perform poorly, due to the syntactical differences of Tweets that are close to spoken language.

Furthermore, a good model for the non-linear structure of Tweets is missing. An important next step would thus be, to think about a way how to either create a linear discourse structure from Tweets or adapt the above explained algorithms to non-linear structures. As always, real world data, esp. Twitter, is very dirty and unstructured, so it will be a challenging task to implement a script that chunks Twitter data into discourses.

Due to ongoing work on the DiMLex one last implementation step that should soon be feasible is the relation labeling.

References

[LNK12] Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. A pdtb-styled end-to-end discourse parser. *Natural Language Engineering*, 2012.

[PDT] *The Penn Discourse Treebank 2.0 Annotation Manual.*