[e1] Yes, it's annoying and cumbersome to separate your rubbish properly all the time.

[e2] Three different bin bags stink away in the kitchen and have to be sorted into different wheelie bins.

[e3] But still Germany produces way too much rubbish

[e4] and too many resources are lost when what actually should be separated and recycled is burnt.

[e5] We Berliners should take the chance and become pioneers in waste separation!

2

3

4

c2

c3

1

c1

5

**From Peldszus & Stede (2015)'s description in Section 3 - Transformation**

"All relations pointing to edges are rewritten to point to the source node of the original target edge" (p.940)

Note: there wasn't any code available, but the diagram is built from what could be inferred from the text description above.

C1; 4 → ADU5

C2; 1 → ADU1

ADU2

C3; 5

ADU3 ← C3; 3 — ADU4
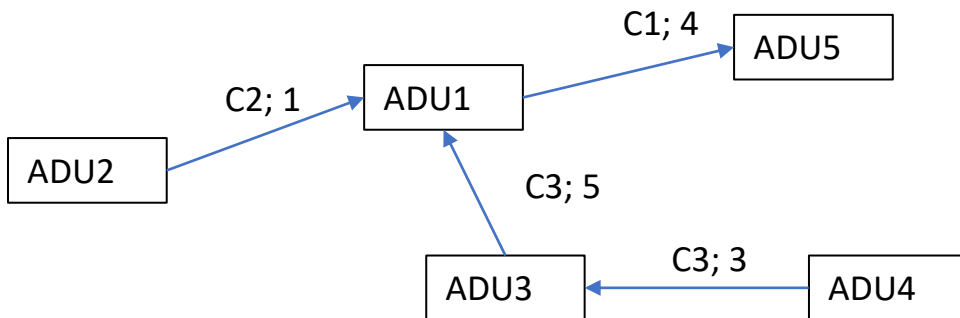
relation_type =
{0:"seg", 1:"sup", 2:"exa", 3:"add", 4:"ɹeb", 5:"und"}

hf_dataset['train'][0] - .json
id = 'micro_b001'
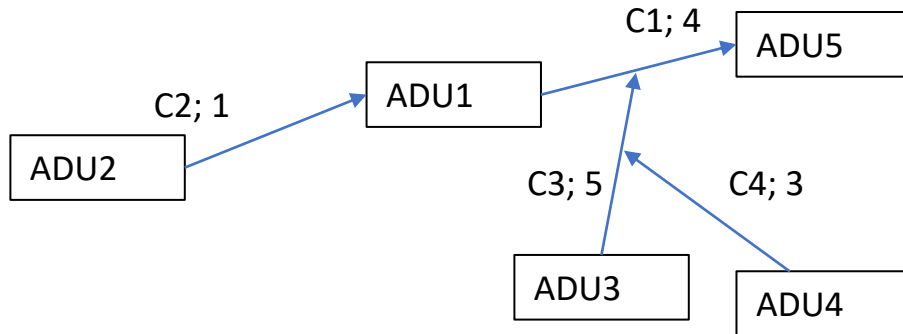
Note: C2; 1 means edge_id = 2; relation_type = 1

zip(hf_doc['edges']['id'], hf_doc['edges']['src'], hf_doc['edges']['trg'], hf_doc['edges']['type']) =
('c1', 'a1', 'a5', 4)
('c2', 'a2', 'a1', 1)
('c3', 'a3', 'c1', 5)
('c4', 'a4', 'c3', 3)

ADU1:
Yes, it's annoying and cumbersome to separate your rubbish properly all the time.
ADU2:
Three different bin bags stink away in the kitchen and have to be sorted into different wheelie bins.
ADU3:
But still Germany produces way too much rubbish
ADU4:
and too many resources are lost when what actually should be separated and recycled is burnt.
ADU5:
We Berliners should take the chance and become pioneers in waste separation!

relation_type[0] = 'seg': segment, mapping EDU(s) to ADU
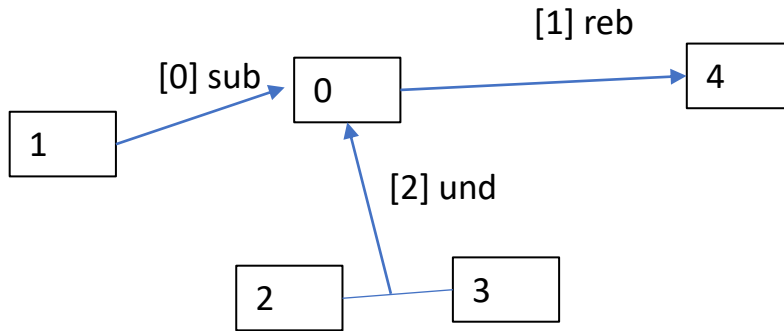relation_type[1] = 'sup': support
relation_type[2] = 'exa': support by example
relation_type[3] = 'add': linked argument
relation_type[4] = 'reb': rebutting
relation_type[5] = 'und': undercutting (a relation)

pie_dataset['train'][0]; ArgMicroDocument
id = 'micro_b001'

[1] reb

[0] sub

0

1

4

[2] und

2

3

Note: 2 and 3 are both heads in this relation

document.adus
[0] = LabeledAnnotationCollection(annotations=(Span(start=0, end=81),), label='opp')
"Yes, it's annoying and cumbersome to separate your rubbish properly all the time."
[1] = LabeledAnnotationCollection(annotations=(Span(start=82, end=183),), label='opp')
"Three different bin bags stink away in the kitchen and have to be sorted into different wheelie bins."
[2] = LabeledAnnotationCollection(annotations=(Span(start=184, end=231),), label='pro')
"But still Germany produces way too much rubbish"
[3] = LabeledAnnotationCollection(annotations=(Span(start=232, end=325),), label='pro')
"and too many resources are lost when what actually should be separated and recycled is burnt."
[4] = LabeledAnnotationCollection(annotations=(Span(start=326, end=402),), label='pro')
"We Berliners should take the chance and become pioneers in waste separation!"

document.relations
[0] = MultiRelation(
heads=(LabeledAnnotationCollection(annotations=(Span(start=0, end=81),), label='opp'),),
tails=(LabeledAnnotationCollection(annotations=(Span(start=326, end=402),), label='pro'),),
label='reb')
[1] = MultiRelation(
heads=(LabeledAnnotationCollection(annotations=(Span(start=82, end=183),), label='opp'),),
tails=(LabeledAnnotationCollection(annotations=(Span(start=0, end=81),), label='opp'),), label='sup')
[2] = MultiRelation(
heads=(LabeledAnnotationCollection(annotations=(Span(start=184, end=231),), label='pro'),
LabeledAnnotationCollection(annotations=(Span(start=232, end=325),), label='pro')),
tails=(LabeledAnnotationCollection(annotations=(Span(start=0, end=81),), label='opp'),), label='und')

Available relation labels and counts in entire dataset:
relation_label_counts = {"reb": 108, "sup": 263, "und": 63, "exa": 9}

dataset_of_text_documents_with_labeled_spans_and_binary_relations['train'][0];
TextDocumentWithLabeledSpansAndBinaryRelations
id = 'micro_b001'
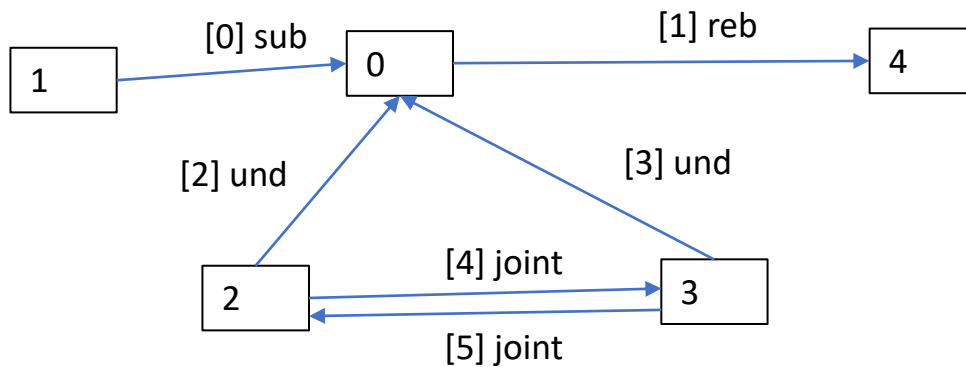
converted_doc.labeled_spans
[0] = Yes, it's annoying and cumbersome to separate your rubbish properly all the time.
[1] = Three different bin bags stink away in the kitchen and have to be sorted into different wheelie bins.
[2] = But still Germany produces way too much rubbish
[3] = and too many resources are lost when what actually should be separated and recycled is burnt.
[4] = We Berliners should take the chance and become pioneers in waste separation!

converted_doc.binary_relations
[0] = AnnotationLayer([BinaryRelation(head=LabeledSpan(start=0, end=81, label='opp', score=1.0), tail=LabeledSpan(start=326, end=402, label='pro', score=1.0), label='reb', score=1.0),
[1] = BinaryRelation(head=LabeledSpan(start=82, end=183, label='opp', score=1.0), tail=LabeledSpan(start=0, end=81, label='opp', score=1.0), label='sup', score=1.0),
[2] = BinaryRelation(head=LabeledSpan(start=184, end=231, label='pro', score=1.0), tail=LabeledSpan(start=0, end=81, label='opp', score=1.0), label='und', score=1.0),
[3] = BinaryRelation(head=LabeledSpan(start=232, end=325, label='pro', score=1.0), tail=LabeledSpan(start=0, end=81, label='opp', score=1.0), label='und', score=1.0),
[4] = BinaryRelation(head=LabeledSpan(start=184, end=231, label='pro', score=1.0), tail=LabeledSpan(start=232, end=325, label='pro', score=1.0), label='joint', score=1.0),
[5] = BinaryRelation(head=LabeledSpan(start=232, end=325, label='pro', score=1.0), tail=LabeledSpan(start=184, end=231, label='pro', score=1.0), label='joint', score=1.0)])

Available relation labels and counts in entire dataset:
relation_label_counts = {"reb": 110, "sup": 281, "und": 65, "joint": 44, "exa": 9}

From the code in Kurabayashi et al.

1 → Support → 0
2 → Attack → 0
3 → Attack → 0
0 → Attack → 4
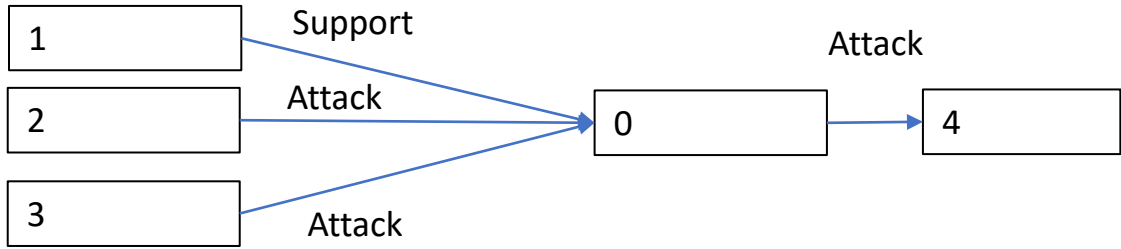
*no explicit details published in the paper what and why the authors decided to modify the relations as such.
- From the code, each document contained one 'Claim', and the rest were labeled as 'Premise'. (see here)
- The relations between ADUs were modified as follows: {"sup": 'Support', "exa": 'Support', "add": take label and target of its previous head, "reb": 'Attack', "und": 'Attack'}. (see here and here)
This modification was evidently appeared as such in the paper (Table 6, p.4698).

Simplified output from the author's preprocessing file on Github repo.

[AC_id] & [argumentation_type] & [distance to *parent*] & [relation_type] & [text]
[0][Premise][4][Attack] = Yes, it's annoying and cumbersome to separate your rubbish properly all the time.
[1][Premise][-1][Support] = Three different bin bags stink away in the kitchen and have to be sorted into different wheelie bins.
[2][Premise][-2][Attack] = But still Germany produces way too much rubbish
[3][Premise][-3] = and too many resources are lost when what actually should be separated and recycled is burnt.
[4][Claim][None][None] = We Berliners should take the chance and become pioneers in waste separation!

Paragraph	AC_id_in_	AC_type	AC_paren	AC_relatic	token
prompt	-	-	-	-	waste
prompt	-	-	-	-	separation
body	0	Premise	4	Attack	Yes
body	0	Premise	4	Attack	,
body	0	Premise	4	Attack	it
body	0	Premise	4	Attack	's
body	0	Premise	4	Attack	annoying
body	0	Premise	4	Attack	and
body	0	Premise	4	Attack	cumbersome
body	0	Premise	4	Attack	to
body	0	Premise	4	Attack	separate
body	0	Premise	4	Attack	your
body	0	Premise	4	Attack	rubbish
body	0	Premise	4	Attack	properly
body	0	Premise	4	Attack	all
body	0	Premise	4	Attack	the
body	0	Premise	4	Attack	time
body	-	-	-	-	.
body	1	Premise	-1	Support	Three
body	1	Premise	-1	Support	different
body	1	Premise	-1	Support	bin
body	1	Premise	-1	Support	bags
body	1	Premise	-1	Support	stink
body	1	Premise	-1	Support	away
body	1	Premise	-1	Support	in
body	1	Premise	-1	Support	the
body	1	Premise	-1	Support	kitchen
body	1	Premise	-1	Support	and
body	1	Premise	-1	Support	have
body	1	Premise	-1	Support	to
body	1	Premise	-1	Support	be
body	1	Premise	-1	Support	sorted
body	1	Premise	-1	Support	into
body	1	Premise	-1	Support	different
body	1	Premise	-1	Support	wheelie
body	1	Premise	-1	Support	bins
body	-	-	-	-	.
body	-	-	-	-	but
body	2	Premise	-2	Attack	still
body	2	Premise	-2	Attack	germany
body	2	Premise	-2	Attack	produces
body	2	Premise	-2	Attack	way
body	2	Premise	-2	Attack	too
body	2	Premise	-2	Attack	much
body	2	Premise	-2	Attack	rubbish
body	-	-	-	-	and
body	3	Premise	-3	Attack	too
body	3	Premise	-3	Attack	many
body	3	Premise	-3	Attack	resources
body	3	Premise	-3	Attack	are
body	3	Premise	-3	Attack	lost
body	3	Premise	-3	Attack	when
body	3	Premise	-3	Attack	what
body	3	Premise	-3	Attack	actually
body	3	Premise	-3	Attack	should
body	3	Premise	-3	Attack	be
body	3	Premise	-3	Attack	separated
body	3	Premise	-3	Attack	recycled
body	3	Premise	-3	Attack	is
body	3	Premise	-3	Attack	burnt
body	-	-	-	-	.
body	4	Claim	None	None	We
body	4	Claim	None	None	Berliners
body	4	Claim	None	None	should
body	4	Claim	None	None	take
body	4	Claim	None	None	the
body	4	Claim	None	None	chance
body	4	Claim	None	None	and
body	4	Claim	None	None	become
body	4	Claim	None	None	pioneers
body	4	Claim	None	None	in
body	4	Claim	None	None	waste
body	4	Claim	None	None	separation
body	-	-	-	-	!

From the code in Morio et al.
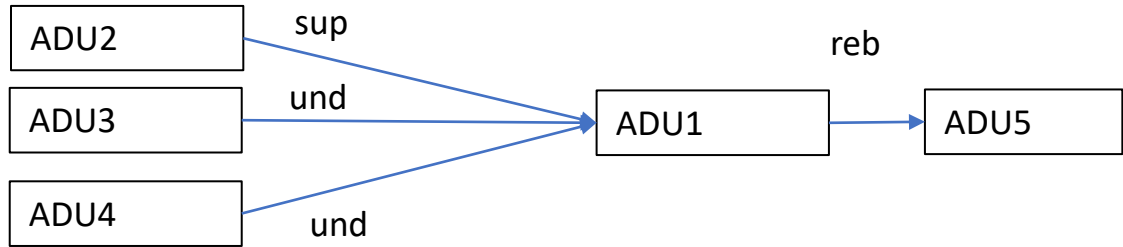
*The code in their repo were not reproduced and no further description on motivation or pre-processing were provided. However, based on the description and the code, it is likely that their data looked like the diagram above.

They seemed to have preserved the ADU's ids and relation's labels from P&S, apart from the modification to the 'add' relation, which they followed Kura et al.

Differences on data modelling

	no. possible relation types	no. of relations in id='micro_b001'	structure
P&S	5	4	tree
Kurabayashi	2	4	tree
Morio	4	4	tree
TextDoc	5	6	graph