WISeN: Widely Interpretable Semantic Networks for Richer Meaning Representation

by

Lydia Feng

Jinho D. Choi
Adviser

Computer Science

Jinho D. Choi

Adviser

Yun Kim

Committee Member

Nosayba El-Sayed

Committee Member

2021

WISeN: Widely Interpretable Semantic Networks for Richer Meaning Representation

By

Lydia Feng

Jinho D. Choi

Adviser

An abstract of
a thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Honors

Computer Science

2021

Abstract

WISeN: Widely Interpretable Semantic Networks for Richer Meaning Representation
By Lydia Feng

Many semantic annotations currently utilize Abstract Meaning Representation and PropBank frameset files to represent meaning. This scheme relies on arbitrary predicate-argument structures comprising unintuitive numbered arguments, fine-grained sense-disambiguation, and high start-up costs. To address these issues, we present a new annotation scheme, WISeN, that prioritizes semantic roles over numbered arguments and does away with sense-disambiguation. This scheme aims to be more intuitive for annotators and more interpretable by parsers. We evaluate this annotation scheme with a two-part experiment. First, we measure speed and accuracy of manual annotations. Second, we train a parser on both AMR and WISeN annotations and measure model accuracy. The results show that WISeN supports improved parser performance and increased inter-annotator agreement without sacrificing annotation speed compared to AMR. As such, we advocate for the adoption of WISeN as an annotation scheme for semantic representations.

WISeN: Widely Interpretable Semantic Networks for Richer Meaning Representation

By

Lydia Feng

Jinho D. Choi

Adviser

A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Honors

Computer Science

2021

Acknowledgements

I am extremely grateful to my friends, family, and the Emory NLP Lab for their support and encouragement throughout this project. Specifically, I am thankful to Gregor Williamson for the hours and insight he put into this project and the WISeN guidelines, for Han He who assisted greatly by training our parsers, and for James Finch for his work on the first iteration of this project and his guidance throughout my time in this lab. I am also thankful to Angela Cao and Maddy Liotta who have seen this project from the beginning to the end with me. Additional thanks to the annotators that comprised this experiment, who gave up their time to help out. I am also grateful to Dr. Yun Kim and Dr. Nosayba El-Sayed for serving on my Honors committee. Your words of guidance throughout my entire undergraduate career are greatly appreciated. Lastly, I am especially thankful to Dr. Jinho Choi, who's unlimited support, positivity, and assurance has not only made this project possible, but also defined my time at Emory. Thank you for introducing me to research and the field of NLP. That is what today means to me.

Contents

1 Introduction **1**

2 Background, Related Work, & Rationale **4**

2.1 Semantic Representations of Language 4

2.2 Thematic Roles 6

2.2.1 Semantics of Numbered Arguments 6

2.2.2 VerbNet Thematic Roles 10

2.3 Sense Disambiguation 14

2.4 Abstract Meaning Representation 15

2.5 AMR Parsing 17

2.6 Rationale 18

3 Methodology **23**

3.1 Overview 23

3.2 Annotation Experiment 25

3.2.1 Corpus 25

3.2.2 Annotators 26

3.2.3 Procedure 27

3.2.4 Evaluation Metrics 28

3.3 Parsing Experiment 28

3.3.1 AMR Corpus 29

3.3.2 WISeN Corpus 30

3.3.3 Parsing AMR and WISeN 36

3.3.4 Evaluation Metrics 39

4 Results 41

4.1 Annotation Experiment 41

4.1.1 Inter-Annotator Agreement 41

4.1.2 Speed of Annotations 44

4.2 Parsing Experiment 45

5 Discussion 49

5.1 WISeN Annotation is More Accurate than AMR 49

5.2 WISeN Annotation is Comparable in Speed to AMR 50

5.3 WISeN Improves Parser Performance 51

5.4 Limitations 52

5.5 Future Work 55

5.6 Conclusions 56

Appendix A - WISeN Guidelines **66**

A.1 Core roles 66

A.1.1 Actor 66

A.1.2 Theme 68

A.1.3 Benefactive 71

A.1.4 Asset 74

A.1.5 Instrument 75

A.1.6 Topic 76

A.1.7 Manner 76

A.1.8 Accompanier 77

A.2 Spatial 78

A.2.1 Location 78

A.2.2 Direction and Path 79

A.2.3 Start and End 80

A.3 Temporal 84

A.3.1 Time 84

A.3.2 Duration 84

A.3.3 Frequency 85

A.3.4 Range 86

A.4 Causal/Conditional/Concessive 87

A.4.1 Cause 87

A.4.2 Purpose 88

A.4.3 Condition 89

A.4.4 Concession 90

A.5 Mereology and Degrees 91

A.5.1 Domain and Mod 91

A.5.2 Attribute 93

A.5.3 Quantity 95

A.5.4 Degree 96

A.5.5 Extent 97

A.5.6 Comparison 97

A.5.7 Possession 100

A.5.8 Part-of and Consist-of 101

A.5.9 Subevent 102

A.6 Operators 102

A.6.1 Op 102

A.6.2 Polarity 103
A.6.3 Polite 104
A.6.4 Mode 104
A.6.5 Example 105
A.6.6 Name 105
A.6.7 Age 106
A.6.8 Value and Ord 106
A.6.9 Unit 107
A.6.10 List 108
A.7 Questions 108
A.8 Relative Clauses 110
A.9 Have-rel-role and have-org-role 112

List of Figures

2.1 AMR graph demonstrating reentrancy 18

3.1 Graph-sequence iteration of Cai and Lam parser [6] 38

4.1 Smatch scores for Group 1 43

4.2 Smatch scores for Group 2 43

4.3 AMR and WISeN Smatch scores per batch 44

4.4 Average time in minutes of annotations in WISeN and AMR . 46

List of Tables

2.1 Predicate-argument structure of `sentence-01` and `fine-01` [3] 7

2.2 List of arguments in PropBank and their semantic roles [4] . . 7

2.3 Distribution of function tags over numbered arguments in PropBank 9

2.4 Distribution of VerbNet thematic roles over numbered arguments in PropBank 13

3.1 Annotator sentence assignments 27

3.2 Predicate-argument structure for `publication-91` 31

3.3 Conversion rules 32

3.4 Predicate-argument structure conversion for `rate-entity-91` 34

3.5 Arguments added to PropBank 35

4.1 Smatch scores for AMR and WISeN 42

4.2 Average minutes per batch for annotators in AMR and WISeN 45

4.3 Performance of AMR and WISeN parsers 48

A.1 List of arguments for `have-degree` 98

A.2 List of arguments for `have-rel-role` 112

A.3 List of arguments for `have-org-role` 112

Chapter 1

Introduction

Natural language processing is concerned with facilitating human interaction with machines. Though natural language is all around us, it is largely meaningless to a computer without any sort of meta data that would allow it to find patterns and make inferences. Because of this, language annotation is an important step toward building devices that can understand human language. Annotation can mark a number of different features of a language corresponding to the six components of language: phonetics, phonology, morphology, syntax, semantics, and pragmatics. This paper focuses on the semantics of language understanding.

Let us first distinguish between the syntax and the semantics of language. Syntax is the grammatical structure of language and is concerned with the order of words in a sentence. This order denotes grammatical subjects, objects, etc. Semantics is the meaning that is conveyed through the words and is concerned with semantic agents, patients, etc. In other words, the subject and object of a sentence can change depending on the order of the words, whereas the agent and patient roles are tied to the meaning of the sentence and do not change with surface grammatical structure.

Graph-based semantic annotation is the process of marking concepts and relations in text in order to create a knowledge graph that can be indexed and referenced. Schemas for annotating the meaning of a sentence have been created to accomplish this task. Most notable is Abstract Meaning Representation (AMR), which represents meaning as a directed acyclic graph [2]. Though this guideline scheme is widely used, it presents several problems regarding its reliance on frameset files from PropBank, a corpus that assigns specific argument structures to predicates [17]. These predicate-argument structures are unintuitive and refer to fine-grained sense-disambiguations. In addition, AMR requires high start-up costs, such as the creation of thousands of frameset files, which restricts the use of AMR to specific domains and to the English language.

In this paper we present a new annotation scheme, WISeN (Widely-Interpretable Semantic Networks) that seeks to rectify these problems by prioritizing semantic roles in capturing ”who is doing what to whom”. WISeN aims to be more intuitive for annotators and more interpretable by parsers. It is evaluated with a two-part experiment where it is directly compared to AMR. The first part of the experiment examines ease of annotation by measuring the time and accuracy of WISeN annotations compared to AMR annotations. The second part of the experiment examines ease of parsing by measuring accuracy of a parser on WISeN and AMR training annotations.

We will first examine prior attempts to create meaningful semantic annotations and see their shortcomings in Chapter 2. Based on these shortcomings, we will present the motivations and rationale for WISeN. After this, we will review the two experiements that we will use to evaluate WISeN in Chapter 3 and discuss their results in Chapter 4. We will demonstrate the advantages of this semantics-based, intuitive annotation scheme, ultimately advocating for the adoption of WISeN in further annotation projects for its ease of annotation, high performance on parsers, and generalizability to different domains and languages.

Chapter 2

Background, Related Work, & Rationale

In this section, we will first examine the ways in which the semantic meaning of text has thus far been represented, introducing the PropBank corpus. Next, we will consider the function of thematic roles in these representations, before discussing the problem of sense-disambiguation. We will then examine the annotation scheme AMR and various attempts to parse it, and end with a summary of the problems associated with these representations that will motivate a new annotation scheme called WISeN.

2.1 Semantic Representations of Language

Early natural language processing work concerned itself with syntactic representations of language, focusing on the part-of-speech (POS) tagging task and the bracketing task, both of which account for the surface grammatical structure of language. The Penn Treebank was created as a large corpus annotated with POS information and skeletal syntactic structure for use in natural language processing and theoretical linguistics [22].

Kingsbury and Palmer [17] attempted to add additional semantic information to the Penn Treebank by assigning specific argument structure labels to predicates in the creation of a corpus called PropBank. In creating these argument structures, syntactic and semantic factors were both considered, but syntactic cues remained prioritized. In following with the Construction Grammar developed by Fillmmore and Kay [15], the meaning of a verb is tied to its syntactic constructions [12]. This was a deliberate decision to avoid the fine-grained division of senses utilized in WordNet, a lexical database where English words are organized into sets of synonyms with semantic relations between the sets [23]. PropBank distinguishes senses only if their argument structures differ. For example, the “render inoperable” sense of *break* and the “cause to fragment” sense have differing argument structures and therefore are distinguished [20]. PropBank lists arguments in order of prominence for each predicate, and they represent different semantic roles. Consider the following two sentences:

1. The flame melted the wax.

2. The wax was melted.

A purely syntactic parser would represent *the wax* of the first sentence as the direct object of *to melt*. In the second sentence, however, *the wax* would be the subject of the verb. PropBank's function tags, on the other hand, represent both instances of *the wax* as the patient of the verb *to melt*, as it is the entity undergoing the melting action. However, there was no attempt to make argument labels consistent with semantic roles across verb senses [20].

2.2 Thematic Roles

2.2.1 Semantics of Numbered Arguments

Arguments in PropBank are specific to the predicate. For instance, the `ARG2` of one verb sense may have a meaning completely different than the `ARG2` of another verb sense. Consider the following two framesets in PropBank and their corresponding numbered argument structure in Table 2.1, where `sentence-01` is used in the context of *The judge sentenced the man to prison for 3 years for his DUI*, and `fine-01` is used in the context of *The judge fined the man $500 for his DUI*.

`sentence-01`	`fine-01`
`:ARG0` is the person doing the sentencing. `:ARG1` is the person being sentenced. `:ARG2` is the sentence (punishment). `:ARG3` is the role or crime. `:ARG4` is the duration of the punishment.	`:ARG0` is the person doing the fining. `:ARG1` is the amount of the fine. `:ARG2` is the person being fined. `:ARG3` is the role or crime.

Table 2.1: Predicate-argument structure of `sentence-01` and `fine-01` [3]

For the sense `sentence-01`, the entity on the receiving end of the sentencing is the `ARG1`. For the sense `fine-01`, the entity on the receiving end of the fining is `ARG2`. Further, `sentence-01` has an argument for the sentence itself as well as an argument for the duration of the sentence, whereas `fine-01` only has an argument for the amount of the fine. It is thus clear that the semantic function of a numbered argument is often predicate-specific. Still, there was an effort made to ensure that `ARG0` and `ARG1` correspond to prototypical agents and prototypical patients respectively, per Dowty's [10] criteria. In general, the numbered arguments in PropBank should correspond to the following semantic roles as outlined in their guidelines.

Numbered Arg	**Semantic Role**
`ARG0`	agent
`ARG1`	patient
`ARG2`	instrument, benefactive, attribute
`ARG3`	starting point, benefactive, attribute
`ARG4`	ending point

Table 2.2: List of arguments in PropBank and their semantic roles [4]

Although `ARG0` and `ARG1` seem to be consistent, it is clear that `ARG2-ARG4` are not. The numbered arguments are thus overloaded, as they account for multiple semantic functions. The extent of the overloading becomes clear when we examine PropBank's function tags, which are semantic roles assigned by predicate for the purpose of marking ways in which a "semantic role can be associated with different syntactic realizations of the same verb" [3].

According to the Uniformity of Theta Assignment Hypothesis [1], identical thematic relationships should be represented by identical structural relationships. However, when we take a closer look at the distribution of PropBank function tags over arguments, it is clear that even `ARG0` and `ARG1`, which, according to Table 2.2, should correspond to semantic roles of agent and patient respectively, are also overloaded.

Perhaps most interesting then, are the roles that are simultaneously `ARG0` and prototypical patients (`PPT`) or the roles that are simultaneously `ARG1` and prototypical agents (`PAG`), since there was a deliberate effort to standardize the opposite function tagging. Taking a closer look at these instances in PropBank, we see some curious annotations.

Consider the predicate `jog-01` which means to run slowly. PropBank

	Numbered Arguments							
	ARG0	**ARG1**	**ARG2**	**ARG3**	**ARG4**	**ARG5**	**ARG6**	**SUM**
PPT	389	8593	1249	49	4	0	0	10284
PAG	8412	664	28	1	0	0	0	9105
GOL	2	503	1436	238	214	2	0	2395
PRD	0	79	701	231	85	10	0	1106
MNR	2	10	808	159	8	11	0	998
DIR	18	147	518	270	14	4	0	971
VSP	1	58	338	214	48	19	0	678
LOC	6	196	268	43	25	4	0	542
EXT	1	5	244	25	3	5	6	289
CAU	75	22	140	30	0	0	0	267
COM	0	83	100	9	4	0	0	196
PRP	0	6	74	32	5	1	0	118
TMP	0	3	15	3	6	1	0	28
ADJ	0	5	10	4	0	0	0	19
ADV	0	2	4	5	1	0	0	12
REC	0	1	2	1	0	0	0	4
SUM	8906	10377	5935	1314	417	57	6	27012

Function Tags

Table 2.3: Distribution of function tags over numbered arguments in PropBank

offers an example sentence: *John jogs 53 miles a day.* In this sentence *John* is said to be the `ARG0` and the patient of *jog*. According to PropBank, the runner of `jog-01` is the `ARG0` and the patient of the predicate [3]. However, since the runner (i.e. John) is the entity performing the act of jogging and the person bringing about the event, the classification of the grammatical subject of *jog* as a thematic patient as opposed to a thematic agent is questionable.

Now let us examine the predicate `free-03` which is the state of costing nothing. PropBank offers the example: *The popcorn is free of charge when you purchase a ticket.* In this sentence, *the popcorn* is said to be the `ARG1` and the agent of *free*. According to PropBank, the thing that costs nothing is the `ARG1` and the agent of the predicate [3]. However, since the thing that costs nothing (i.e. the popcorn) is the entity that is affected by or undergoing the `free-03` event, the classification of the grammatical subject as a thematic agent as opposed to a thematic patient is again questionable.

It is thus clear that numbered arguments in PropBank are semantically overloaded. This leaves automoatic classifiers with a difficult task, since numbered arguments are not distinguished the same way. Further, PropBank data is taken from the Wall Street Journal. This type of text is quite domain-specific. For instance, PropBank may use as an example the bombing of Pearl Harbor, but never encounter more personal text. Since the input data is restricted to a specific domain of news text, there presents further problems of distinguishing numbered arguments for novel domains [28].

2.2.2 VerbNet Thematic Roles

To create a more consistent training data set and improve system performance, Loper, Yi, and Palmer [20] attempted to create a mapping between PropBank and a resource called VerbNet, that consists of twenty-three thematic roles arranged hierarchically by verb class based on syntactic and semantic characterization [26]. VerbNet roles are more verb-independent and more generalizable than PropBank, making them easier for semantic role labeling (SRL) systems to learn.

Unfortunately, PropBank and VerbNet's coverage of words and word senses do not align perfectly. About 25.5% of PropBank framesets (i.e. verb senses) are not covered by VerbNet, and they are unable to be mapped to a thematic role. Of the framesets that could be matched to a VerbNet instance, there were still mismatches in arguments: an argument described in one resource was omitted in the other, or a single argument in one resource is split into multiple arguments in the other. These mismatches reflect both practical and theoretical differences in the resources.

The SRL system performed better with the PropBank numbered arguments and the VerbNet thematic roles, than it did with the PropBank numbered arguments alone [20]. Still, since the SRL system was required to learn more labels (e.g. numbered arguments in addition to thematic roles), it suffered from data sparseness.

Earlier we had noted that there was an effort made to ensure that `ARG0` and `ARG1` of PropBank's predicate-argument structure were consistent with Dowty's [10] criteria that they correspond to prototypical agents and prototypical patients respectively. We saw that this was not always the case with PropBank's function tags, but with the additional information of VerbNet thematic roles, we may attempt again to confirm this effort.

Table 2.4 shows the distribution of VerbNet thematic roles as they were assigned to numbered arguments from PropBank. While `ARG0` does seem to generally coincide with prototypical agent thematic roles, `ARG1` covers quite a bit more than prototypical patient roles, including 231 instances of destination, 172 instances of stimulus, and 145 instances of location.

It is also important to note that while almost 75% of PropBank framesets were covered by VerbNet [20], only about 40.6% of PropBank's 27,012 numbered arguments were able to be mapped to a VerbNet thematic role. This leaves 16,127 arguments in PropBank without semantic information.

Numbered Arguments								
		ARG0	**ARG1**	**ARG2**	**ARG3**	**ARG4**	**ARG5**	**SUM**
VerbNet Thematic Roles	**agent**	3462	30	1	1	0	0	3494
	theme	208	1661	371	13	0	0	2253
	patient	13	1131	20	0	0	0	1164
	experiencer	187	264	5	2	0	0	458
	destination	0	231	183	21	10	1	446
	stimulus	247	172	14	0	0	0	433
	location	7	145	142	30	23	1	348
	source	17	109	194	7	2	0	329
	recipient	0	56	251	10	0	0	317
	instrument	0	2	243	51	0	3	299
	topic	0	192	61	5	0	0	258
	co-patient	0	6	151	4	1	0	162
	beneficiary	0	40	47	44	7	0	138
	attribute	0	9	101	7	2	6	125
	result	0	30	81	5	7	0	123
	co-agent	0	69	25	0	0	0	94
	material	1	25	46	9	0	0	81
	goal	0	8	58	6	1	0	73
	co-theme	0	37	27	5	1	0	70
	product	0	35	17	4	13	0	69
	initial_location	0	9	23	8	0	0	40
	cause	30	3	3	0	0	0	36
	asset	0	21	0	11	1	1	34
	predicate	0	4	18	6	0	0	28
	pivot	26	1	0	0	0	0	27
	extent	0	0	26	6	0	0	26
	value	0	5	13	7	0	0	25
	trajectory	0	3	0	0	0	0	3
	actor	1	0	0	0	0	0	1
	proposition	0	0	0	1	0	0	1
	SUM	4199	4298	2121	257	68	12	10955

Table 2.4: Distribution of VerbNet thematic roles over numbered arguments in PropBank

2.3 Sense Disambiguation

For those interested in machine translation, one of the greatest difficulties encountered is that one word may have several different meanings, though this problem is not restricted to that field [29]. For example, the verb *to buy* can describe a way of obtaining something by exchanging payment, as in the sentence *she bought the dress.*, but it can also mean the act of believing something to be true, as in the sentence *the teacher didn't buy the student's excuse.* These two different meanings of the same phonological representation are two different senses of the word *buy.* Disambiguation between these senses depends on the context of the surrounding words.

While there has been much research on the efficacy of machine learning models on word sense disambiguation, there are few studies focusing on how annotators disambiguate. Experiments that test inter-annotator agreement on word sense disambiguation aim to establish ways to re-code individual annotations to obtain an artificially higher agreement score [27]. Pure sense disambiguation among six human judges resulted in only 29% of verbs obtaining unanimous agreement. These low agreement levels were attributed to sense distinctions that are too-fine grained for NLP purposes, as they are derived from common dictionaries. However, when recomputed with the judges' top-level distinctions, the reduction in disagreement was minimal.

Further work from Bruce [5] and Ng [24] investigates algorithmic derivation of sense classes which correspond to human intuitive judgement while achieving higher agreement rates. Ng notes that inter-annotator agreement for word sense tagging is quite low, concluding that language-users are able to process language without performing this task to the fine-grained resolution available in a traditional dictionary that has been the goal of many language models.

2.4 Abstract Meaning Representation

The Abstract Meaning Representation (AMR) language was introduced by Banarescu et al. [2] as an abstraction away from template-based methods of language generation. Templates avoid the need for linguistic decision-making and large complex knowledge resources, but are not expressive, flexible, nor scalable enough for many domains. Oftentimes, surface syntactic structures do not provide insight into the semantics of text, thus making abstraction from templates an important solution to machine translation, language generation, and other natural language processing tasks.

AMRs are directed, acyclic graphs which have edges labeled with relations and leaves labeled as concepts. AMRs are derived from the PENMAN

Sentence Plan Language [13]. AMR does not annotate individual words in a sentence, nor does it designate elements into categories such as nouns and verbs. Because of this, English function words often do not show up. For example, the sentence *The boy wants the girl to believe him* is represented as the following.

```
The boy wants the girl to believe him
(w / want-01
   :ARG0 (b / boy)
   :ARG1 (b2 / believe-01
            :ARG0 (g / girl)
            :ARG1 b))
```

This AMR says that there exists a wanting event where a boy is the wanter and the thing wanted is a believing event, where the believer is a girl and the thing believed is the same boy from before. The numbered arguments used in AMR as core roles follow from PropBank's predicate-argument structure. In addition to these, there are approximately 100 relations including semantic relations that make up the non-core roles [2].

As AMR and PropBank are sister projects, it should follow that they draw from the same set of semantic roles. However, PropBank's semantic information is based in VerbNet, which, as we have previously noted, covers only 74.5% of PropBank senses and only 40.6% of PropBank arguments [20].

How, then, can AMR properly relay semantic information if the predicate-argument structure it relies upon is not based in AMR's semantic roles, but rather VerbNet's?

2.5 AMR Parsing

Parsing AMR is the task of generating AMR graphs from natural language text input. This requires concept identification, where spans of the input text are mapped to graph concepts, relation identification, where edges among these concepts are identified, and other more nuanced tasks such as the problem of reentrancies, in which one concept can appear in multiple relations, see Figure 2.1. To accomplish these, parsers have traditionally used a pipeline to first train a model on aligning words in a sentence to graph concepts, independent of parsing objectives, before training the parser [11].

Variations on this approach include Lyu and Titov's [21], which treat these alignments as latent variables in a joint probabilistic model and assumes that concepts are triggered by single words from the input, and that each word in the input corresponds to at most one concept. This model achieves an accuracy of 74.4% as evaluated using a Smatch metric [7].

More recent work by Cai and Lam [6] implements a dual graph-sequence

Figure 2.1: AMR graph demonstrating reentrancy
This graph says "the boy wants the girl to believe him." The concept `boy` is involved in an ARG0 relation with `want-01` as well as an ARG1 relation with `believe-01`

model that is based on iterative inferences. This approach builds an AMR graph node by node, performing multiple rounds of attention and reasoning at each step to make harmonious decisions between the concept identification and relation identification tasks. The overall accuracy for this model ranges from 74.5% to 80.2%.

2.6 Rationale

Many current semantic representations of language use AMR, which utilizes PropBank's sense-disambiguated predicate-argument structure. This raises two issues. First, PropBank's numbered arguments are semantically overloaded, despite attempts to add function tags and VerbNet thematic roles. Second, sense-disambiguation is a difficult and fine-grained process that users of natural language do not explicitly engage in when annotating; this process is often unconscious. These issues are realized in two different ways: the parsing task and the annotation task.

Learning semantically overloaded numbered arguments in addition to a wide array of semantic relations is not an easy task. Further, the numbered arguments do not correspond to the same thing in each instance that the parser will see. If the `ARG3` is the benefactive in one predicate but the recipient in another, as outlined in Table 2.2, then it is difficult for the model to map meaning onto these numbered arguments. Furthermore, input to an AMR parser does not arrive sense-disambiguated, and the parser does not perform sense disambiguation as it does not access the PropBank frameset files. This information then, only confuses the parser more.

Sense disambiguation and overloaded arguments also make manual annotation a more difficult and time-consuming task. First, annotators must make fine-grained distinctions between different verb senses, a task that natural language users struggle to do explicitly [24]. For example, PropBank lists twenty instance of the verb *go*, not including multi-word constructions, differentiating between `go-01` expressing motion, `go-02` expressing self-directed motion, and `go-03` expressing pursuit, among others [3]. Sense disambiguation, and thus annotations, also rely heavily on the PropBank frames. This task of sense-disambiguation thus appears unintuitive and would cost annotators time and effort.

Further, annotation relies on these predicate-argument structures outlined in PropBank's frameset files. Annotators must make use of relatively arbitrary numbered arguments. To correctly annotate in AMR, they must constantly refer to the PropBank frameset files which outline the specific argument structure for that predicate. Referring to the more than 27,000 frames each time an annotator needs to know how to annotate an argument is time-consuming. Since annotation relies heavily on these frames, they can get in the way of more intuitive annotations. Take, for instance, the two structurally similar sentences below:

1. What type of person would read this book?

2. What kind of person would read this book?

One would think that the AMRs for sentence 1 and sentence 2 are similar, but they actually differ quite a bit, and the reason for this lies in PropBank.

PropBank has a frameset file for `type-03` but not for *kind*. Therefore, the resulting AMRs are as follows.

What type of person would read this book?

```
(r / read-01
   :ARG0 (p / person
            :ARG1-of (t / type-03
                     :ARG2 (a / amr-unknown)))
   :ARG1 (b / book
            :mod (t / this)))
```

What kind of person would read this book?

```
(r / read-01
   :ARG0 (p / person
            :mod (k / kind
                     :mod (a / amr-unknown)))
   :ARG1 (b / book
            :mod (t / this)))
```

Annotation thus depends heavily on the existence of frameset files in PropBank. If an annotator encounters a new predicate, they must create a frameset file and a predicate-argument structure for it. This raises problems when annotating in novel domains, where a new frameset file would need to be created to accommodate for new predicates. On a more broad level, the dependence on frameset files also restricts AMR to the English language.

Unless frameset files are created for predicates in another language, this annotation scheme would not be able to accommodate anything other than English text.

To solve these issues, we introduce a new annotation scheme called Widely Interpretable Semantic Networks or WISeN. WISeN gets rid of numbered arguments, using only thematic roles to encode the meaning of a sentence, and removes the need to disambiguate senses. This annotation scheme attempts to be more intuitive for annotators and more interpretable by parsers. Instead of having to refer to a set of pre-constructed predicate-argument structures, annotators will be able to draw upon their own knowledge and intuition of their language and annotate according to that, and parsers will have less roles overall to learn. Further, neither annotators nor parsers perform sense disambiguation to the granularity utilized in PropBank, so expecting either to do so is an unreasonable extra step. WISeN attempts to solve these problems and make manual annotation as well as parsing easier.

Chapter 3

Methodology

3.1 Overview

We first created a revised set of annotation guidelines called Widely-Interpretable Semantic Networks (WISeN) to resolve a number of deficiencies present in AMR and PropBank that were discussed in the previous section. Although it may seem that the introduction of specific argument roles would lead to a proliferation of semantic relations, the opposite is actually the case. We began with the non-core roles in AMR, with the exception of *source, destination, and medium*, which we have merged into other roles, and added a small number of new roles based on the PropBank and VerbNet roles. These semantic roles are able to completely replace `Arg2` – `Arg6`. Thus, WISeN actually uses fewer roles and is able to reduce the semantic workload of the numbered argument relations, see Appendix A.

In order to see how effective these guidelines are, and whether or not they solve the problems from AMR and PropBank, we conduct a two-part experiment. The first is an annotation experiment, in which we test whether the accuracy and speed of annotations increases with the removal of numbered arguments and sense-disambiguation as outlined in WISeN. If both increase, then it would seem that the WISeN annotation scheme is more intuitive for human language users. The second part of the experiment is a parsing experiment where we convert a large, pre-existing corpus of AMR annotations into a corpus of WISeN annotations. This conversion requires a mapping of the PropBank numbered arguments to the WISeN thematic roles. Though arguments in PropBank have been mapped to thematic roles in the past, they were mapped to VerbNet roles, which is counterintuitive when we see that PropBank is relied upon mostly by AMR annotations. Why, then were they not mapped to AMR roles? This second experiment will map the PropBank arguments to WISeN roles and then we will examine the parser's performance on the converted corpus. Improved performance of the parser motivates removing numbered arguments and sense-disambiguation in favor of prioritizing thematic roles.

3.2 Annotation Experiment

In this section, we will outline the details of our first experiment, the annotation experiment. We will begin by examining the corpus of sentences used and the participants who annotated. Next, we will discuss the procedure of their annotation, before detailing the metrics with which we will evaluate this experiment.

3.2.1 Corpus

For the annotation experiment, 1,000 sentences from a variety of different dialogue datasets were chosen. 200 sentences were used from DailyDialog, a dataset of human-written conversations [19]. 100 sentences were used from EmpatheticDialogues, a dataset of dialogues gathered using Amazon Mechanical Turk [25]. 200 sentences were used from PersonaChat, a dataset of dialogue based around consistent personalities [14]. 200 sentences were used from the Boston English Centre's English conversational sentences [8], and the last 300 sentences were collected from an Amazon Mechanical Turk task, where turkers were provided with a persona statement from the PersonaChat dataset and were asked to respond with an emotionally driven reaction as well as an engaging follow-up utterance. 100 sentences came from these emotionally driven reactions, and 200 sentences came from the engaging follow-up utterances. These 1,000 sentences were split up into twenty batches of fifty sentences each, and there was an effort to make batches equal in length and complexity. This was done by distributing different sentence structures (i.e. ones with many dependent clauses vs. short simple one-clause sentences). Ten batches were used as part of the annotation experiment, and another ten batches were annotated to provide more information for the parsing experiment in Section 3.3.

3.2.2 Annotators

The annotators in this experiment were comprised of seven undergraduate linguistics students at Emory University and one postdoctoral researcher in computational linguistics. Before beginning, each annotator read the appropriate annotation guidelines and was trained on the scheme with a set of thirty sentences.

Two of the annotators, whom we will call Annotator A and Annotator B created the WISeN guidelines, while annotators C-H simply read and were trained on the guidelines.

3.2.3 Procedure

All annotators were required to annotate in both AMR and WISeN in order to directly compare performance. Annotators were tasked with annotating 150 sentences in AMR and 150 sentences in WISeN. Due to time constraints Annotator H completed 100 sentences in AMR and 100 sentences in WISeN. As such, nine batches of AMR and nine batches of WISeN were doubly-annotated. The distribution of batches of fifty sentences over annotators is summarized in Table3.1.

Batch of 50	AMR annotators	WISeN annotators
01	F, G	A, C
02	F, G	C, D
03	F, H	C, E
04	G, H	D, E
05	B	D, E
06	A, C	F, G
07	C, D	F, G
08	C, E	F, H
09	D, E	G, H
10	D, E	B

Table 3.1: Annotator sentence assignments

It was expected that annotators would get better and faster at annotating as they progressed, especially since there are similarities between the WISeN guidelines and the AMR guidelines. To control for this variable of familiarity, the eight annotators were split in half into two groups. Both groups began with batches 01-05, but Group 1 began in AMR and Group 2 began in WISeN. At the conclusion of these annotations, both groups proceeded to batches 06-10, but Group 1 switched to WISeN and Group 2 switched to AMR.

3.2.4 Evaluation Metrics

Annotators kept track of their speed by recording how many sentences they were able to annotate and the amount of time it took them in minutes. They were asked to do this every time they completed an annotation session. Upon completion, both AMR and WISeN annotations were converted to PENMAN [13], and inter-annotator agreement was calculated by computing a Smatch score. Smatch converts annotations into triples represented by relation(variable, value), where the value could very well be another variable. Of all the possible variable mappings, Smatch computes the maximum number of matched triples and gets the F-1 score [7].

3.3 Parsing Experiment

In this section, we will outline the details of our second experiment, the parsing experiment. We will begin by analyzing the AMR 3.0 corpus and discussing the changes that were necessary to this corpus. Then, we will demonstrate how we converted the AMR corpus into the WISeN annotation scheme using a combination of automatic, rule-based conversions and manual conversions. In this section we will also highlight problems in PropBank that make this conversion difficult. After this, we will examine the model with which a parser will be trained on AMR and WISeN data and discuss the way the parser's performance will be evaluated.

3.3.1 AMR Corpus

In the parsing experiment, 59,255 AMR annotations which were developed by the Linguistic Data Consortium as part of the AMR 3.0 corpus was used. These annotations are of English natural language sentences gathered from broadcast conversations, web text, web discussion forums, Wall Street Journal text, Wikipedia articles, and Aesop's Fables, among other sources [16].

In the AMR 3.0 corpus, there were 574 predicates used that did not reference a PropBank frameset file, including predicates such as `pack-sand-00`, `strawman-00`, and `internationalize-00`. It appears that these predicates were created ad-hoc by annotators. As such, the argument structure for these predicates are fairly arbitrary and are not standardized through Prop-

Bank. Because of this, conversions for these predicates would be similarly arbitrary. To avoid this problem, we removed any annotations from the AMR 3.0 corpus that had mention of these predicates. AMR also uses special frames for reified roles and certain entities. The WISeN guidelines are not yet able to handles some of these special frames, as the argument structure for them are highly specific and non-generalizable, see Table 3.2. In total we removed six of these for ease of handling and conversion: `street-address-91`, `publication-91, byline-91, course-91, distribution-range-91` and `statistical-test-91`.

Ultimately, 2,339 AMRs were deleted from the training set, leaving it with 53,296 AMR annotations. 66 were deleted from the development set, leaving 1,651, and 85 were deleted from the test set, leaving 1,813 AMRs. These resulting training, deveopment, and test sets comprise the AMR corpus used in the parsing experiment.

3.3.2 WISeN Corpus

In order to test the relative performance of a model on AMR annotations and WISeN annotations, we use the same corpus for WISeN as we do for AMR (e.g. the revised AMR 3.0 corpus) and apply rules to convert the

publication-91	
ARG N	**description in AMR**
ARG1	author
ARG2	title
ARG3	abstract
ARG4	text
ARG5	venue
ARG6	issue
ARG7	pages
ARG8	ID
ARG9	editors

Table 3.2: Predicate-argument structure for `publication-91` [18]

numbered arguments into WISeN roles. Using a combination of the argument number, function tag, VerbNet role, and description that was encoded into PropBank, we were able to generate these mappings

For each of the 5,789 predicates in the AMR 3.0 corpus, we attempted to create a mapping into WISeN for it's entire argument structure outlined in PropBank. This resulted in 15,120 unique arguments to be mapped. After running the conversions from Table 3.3, 12,311 mappings were created, leaving 2,809 arguments unmapped. Due to time constraints, we then analyzed which arguments in particular were actually being used in the AMR corpus. We found that of the 2,809 unmapped arguments remaining, 602 of them do not appear anywhere, and these were left unassigned in our mapping. This resulted in 2,207 arguments which could not be automatically mapped to a WISeN role.

ARGN	**F-Tag**	**VerbNet Role**	**Description**	**WISeN Role**	**Count**
+ARG0	+PAG			Actor	4709
+ARG0	+CAU			Actor	43
+ARG1	+PPT			Theme	4712
+ARG1	+PAG		+(entity|thing)	Theme	232
	+MNR	+instrument		Instrument	164
	+MNR	-instrument		Manner	349
	+GOL	+destination		End	246
	+GOL		+ (end point|ending point| state|destination|attach| attached|target)	End	183
	+GOL	+ (beneficiary|recipient| experiencer)		Benefactive	336
	+GOL		(benefactive|beneficiary|recipient| listener|hearer|perceiver|to whom| pay|paid)	Benefactive	297
	+LOC	+destination		End	17
	+LOC	+initial_location		Start	3
	+LOC	+source		Start	1
	+LOC	-destination		Location	270
	+DIR	+initial_location		Start	31
	+DIR	+source		Start	177
	+DIR		+(start|source|from|starting)	Start	260
	+COM	-recipient & -beneficiary		Accompanier	121
	+COM	+(recipient|beneficiary)		Benefactive	0
ARG1	+VSP	+asset		Theme	11
	+VSP		+(price|money|rent| amount|gratuity)	Asset	56
	+PRP		+(purpose|for)	Purpose	52
-ARG1	+CAU	-recipient	+(why|reason|source| cause|crime|because)	Cause	51
	+VSP	+(material|source)		Start	46
	+VSP		+(start|material|source)	Start	12
	+VSP		+(aspect|domain) & -specific	Domain	34

Table 3.3: Conversion rules

The remaining conversions were done manually by analyzing each argument's function tag, VerbNet role, and description in PropBank, as well as the actual usages in the AMR 3.0 corpus to inform our decisions on the

WISeN role mapping. While many were able to be converted this way, there were 218 roles that had inconsistent usage in the AMR 3.0 annotations. For example, the `ARG2` of `send-02` was annotated as the "hospital" in *send her to the hospital*, the "flying" of *sending his eyeglasses flying*, and "fight in the war" in the annotation *sent troops to fight in the war*. Though this argument was mapped to a function tag of secondary predication, and its description involved a project and impelled action, it is clear from the examples that the entities annotated as the `ARG2` of `send-02` represent destination, inverse cause, and purpose semantic relationships respectively. As such, a definitive mapping of the `ARG2` to one WISeN role would result in incorrect annotations.

These remaining 218 roles were flagged and and then mapped by hand to a WISeN role based on their usage in the AMR 3.0 corpus, and if those usages were not consistent, then we deferred to the PropBank descriptions to create these mappings. Some arguments were inconsistent to the point where manual edits to the corpus was the only option, such as the `ARG2` of `pull-06`. The following are examples of sentences from the AMR 3.0 corpus with the `ARG2` of `pull-06` bracketed and the resulting WISeN role conversion following the arrow.

- I'm pulling my hair [out] → direction

- pulling the plug from [his dialysis machine] → start

- you don't pull any punches with [your headlines] → instrument

During these manual annotations, small errors in the automatic conversions were noticed and exceptions were found. As such, 72 of the previous role mappings were manually overwritten. We also removed one role completely: the `ARG2` of `hightail-01` which is fixed as "it" in *hightail [it] out of town.*

Next, we added conversions for many of AMR's special frames, such as `rate-entity-91, score-on-scale-91, have-org-role-91`, as well as 33 others for a total of 93 new roles. See Table 3.4 for an example conversion.

`rate-entity-91`		
ARG N	**description in AMR**	**WISeN Role**
`ARG1`	quantity	quantity
`ARG2`	reference quantity	duration
`ARG3`	the regular interval between events	frequency
`ARG4`	entity upon which recurring events happen	subevent

Table 3.4: Predicate-argument structure conversion for `rate-entity-91`

Lastly there were arguments of predicates used in the corpus of AMR annotations that do not exist in PropBank. In other words, a frameset file for the predicate did exist in PropBank, but this frameset file may not outline all the arguments being used. There were ten of these arguments total and they were added to the conversion mapping. The added arguments and corresponding WISeN role are as follows:

Sense	**Arg N**	**WISeN Role**
`bind-01`	`ARG4`	Manner
`damage-01`	`ARG3`	Extent
`late-02`	`ARG3`	Extent
`misconduct-01`	`ARG1`	Mod
`oblige-02`	`ARG2`	Theme
`play-11`	`ARG3`	Theme
`raise-02`	`ARG3`	Purpose
`rank-01`	`ARG5`	Comparison
`unique-01`	`ARG3`	Benefactive
`unique-01`	`ARG4`	Comparison

Table 3.5: Arguments added to PropBank

With all the roles accounted for, the AMR corpus was converted into WISeN using a conversion of the 5,810 senses and the 14,674 roles. To convert between schemes, we used the PENMAN API [13] to change one directed acyclic graph into another. We also removed the numbers at the end of predicates used for sense-disambiguation during the conversion into WISeN. Furthermore, several AMR relations were encompassed by WISeN relations. For instance, the WISeN relation `end` encompasses AMR's `destination`, and WISeN's `manner` encompasses AMR's `medium`.

It is important to note that although this conversion algorithm does away with numbered arguments in favor of the roles outlined in the WISeN guidelines, WISeN and AMR differ in more ways than just this. WISeN is not reliant upon the existence of PropBank frameset files, allowing unbounded creation of predicates without sense disambiguation. This feature of the annotation scheme was not accounted for when converting the AMR 3.0 corpus so there may be structural differences between the converted WISeN corpus and true WISeN. Thus, the results of these conversions cannot reflect the expressive power of the WISeN annotation scheme, but rather, these results focus in on the elimination of numbered arguments.

3.3.3 Parsing AMR and WISeN

A parser was run on the AMR corpus and the converted WISeN corpus using the training, development, and test split provided in the AMR 3.0 corpus. These parsers aimed to transform natural language text into the graph representation of AMR and WISeN respectively by extracting concepts from input text and constructing nodes. Using Cai and Lam's [6] dual graph-sequence iterative inference model, these graphs are built incrementally by expanding one node at each step. These expansions comprise the parsing model at different steps in the process, starting with an empty graph G^0. Each iteration of the graph G^i is expanded by constructing edge predictions, $g()$, and node predictions $f()$, based on the input sequence W, and the current semantic graph G^i in order to create sequence hypotheses about what part of the input to extract, x_t^i, and graph hypotheses about where in the graph to construct nodes, y_t^i. Thus for each iteration of the expansion,

$$y_t^i = g(G^i, x_t^i), \text{ and}$$

$$x_{t+1}^i = f(W, y_t^i).$$

First, the input sentence is converted into vector representations that generates text memories for each token, W. Then, for each iteration, the graph encoder, takes the current graph G^i and applies a multi-layer Transformer with masked self-attention and source-attention that prevents nodes from attending to positions after itself in the node sequence, while allowing it to attend to all positions in the input sequence, thereby generating graph memories.

Then, using the latest graph decision and the input sequence memories, the concept solver uses attention weights to compute the probability of each new possible concept. First, the concept label prediction probability is compared to a pre-defined vocabulary. Then, the probability of copying a token lemma from the input as a concept node label is computed, and finally, the probability of copying an original string from the input as a node label is computed as well. The final prediction probability of a concept is the summation of the probabilities of each of the three channels.

Next, the relation solver relates nodes in the current graph to the new concept decided on by the concept solver. By examining preceding nodes and using a relation classification task, the relation type between the new concept and the possible source nodes are predicted. The concept solver and the relation solver pass information between each other in order to make coherent and consistent decisions. Then, a classifier predicts the edge label using the concept and the node vector. The resulting concept, edge, and edge label are added to the next iteration of the graph, G^{i+1}. In this way, an input sentence is parsed into the resulting AMR or WISeN graph node by node.

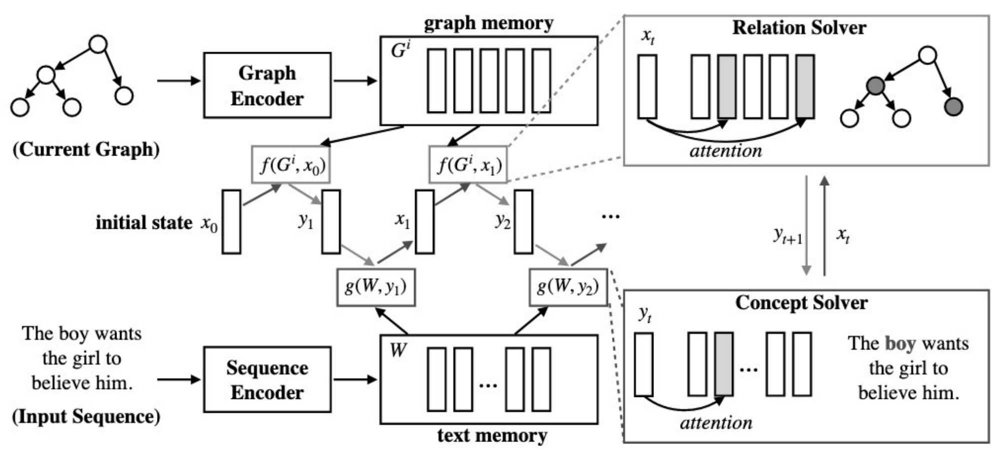

Figure 3.1: Graph-sequence iteration of Cai and Lam parser [6]

Given the input sequence W and the current graph G^i, the model iteratively makes graph hypotheses x_i and sequence hypotheses y_i.

3.3.4 Evaluation Metrics

Following this training, the AMR parser and the WISeN parser were tested on their respective testing splits. Performance was evaluated using Smatch [7] overall and on specific subtasks, including its performance on an unlabeled graph and concept identification. In addition, each parser was tested on our own manual annotations of the dialogue sentences. We adjudicated the annotations produced from our annotation experiment outlined in Section 3.2 for a total of 500 AMR and 500 WISeN annotations. In addition to this, Annotator A and Annotator B completed another 500 AMR annotations and 500 WISeN annotations. Together, this project produced 1,000 AMR annotations and 1,000 corresponding WISeN annotations from the 1,000 dialogue sentences outlined in the corpus for the experiment in 3.2.

However, the WISeN parser was trained on WISeN annotations that were converted from AMR. As previously mentioned, the resulting conversions do not reflect the full expressive power of WISeN. To ensure that evaluation of the parsers is fair, we converted the 1,000 AMR annotations from our annotation experiment into WISeN ones using the same conversion algorithm, and we also tested the WISeN parser on this. Ultimately, the AMR parser was tested on its test split from the corpus and the 1,000 AMR annotations produced in our experiment, and the WISeN parser was tested on its test split from the corpus, the 1,000 true WISeN annotations produced in our experiment, as well the 1,000 converted WISeN annotations produced by converting the 1,000 AMR annotations.

Chapter 4

Results

4.1 Annotation Experiment

4.1.1 Inter-Annotator Agreement

For the nine doubly-annotated batches in AMR and the nine doubly-annotated batches in WISeN, inter-annotator agreement was calculated between annotators using a Smatch score [7]. No score exists for batch05 in AMR and batch10 in WISeN due to time constraints with annotator H. We found that WISeN annotations achieve a higher inter-annotator agreement than AMR annotations ($p < 0.0005$). The resulting Smatch scores for the doubly-annotated WISeN and AMR batches are summarized in Table 4.1, with the bolded scores annotated by Group 1 annotators who began with AMR and switched to WISen, and the non-bolded scored annotated by Group 2 annotators who began with WISeN and switched to AMR. These Smatch scores are averages of the Smatch score for each sentence in the batch.

Batch #	**AMR**	**WISeN**
01	**0.8142**	0.8972
02	**0.7358**	0.779
03	**0.7406**	0.7678
04	**0.7134**	0.7232
05	-	0.7106
06	0.7752	**0.787**
07	0.6956	**0.8142**
08	0.7758	**0.7950**
09	0.7222	**0.7642**
10	0.7064	-
mean	0.7421	0.7820

Table 4.1: Smatch scores for AMR and WISeN

Overall, the average Smatch score for AMR annotations was 0.7421 and the average Smatch score for WISeN annotations was 0.7820. This is a 5.376% difference. Looking at each group individually, we see that Group 1 saw a 4.937% increase in average Smatch score when they switched from AMR to WISeN. Group 2 actually saw a 5.284% decrease in Smatch score when they switched from WISeN to AMR. Therefore, both groups have higher agreement when using WISeN guidelines, even if it was the first scheme learned.

According to Figure 4.3, in nearly every batch, WISeN annotations achieve a higher Smatch score, even from annotators for whom this is their first guideline scheme.

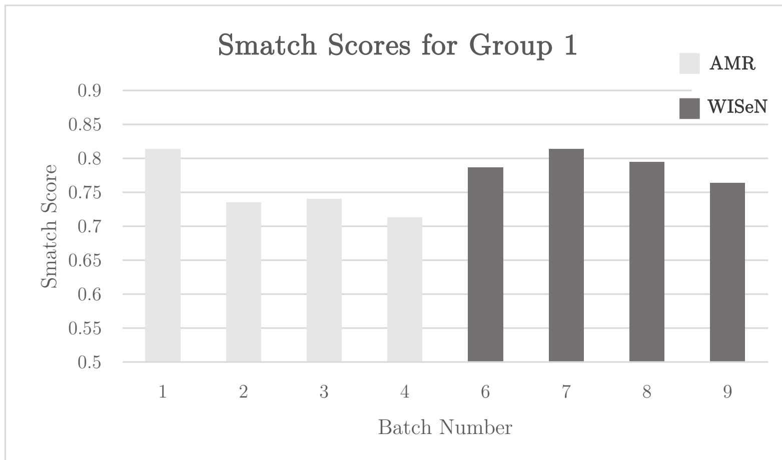

Figure 4.1: Smatch scores for Group 1

Bar graphs represent smatch agreement score for each batch. Light gray bars indicate AMR annotation scheme with a 0.751 average, and dark gray bars indicate WISeN annotation scheme with 0.790 average.

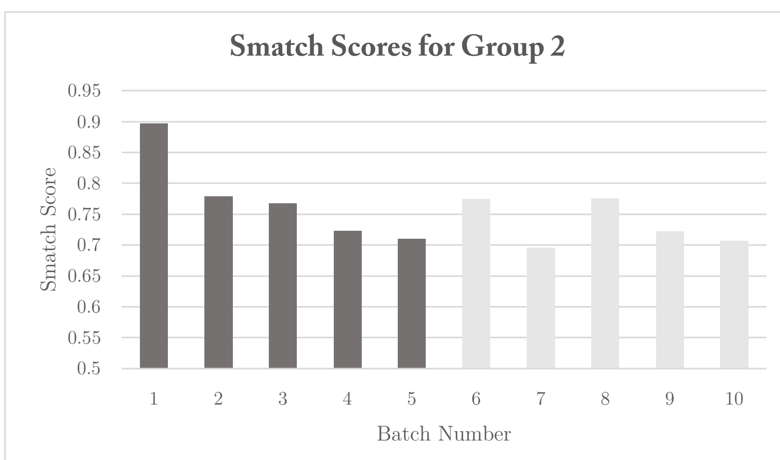

Figure 4.2: Smatch scores for Group 2

Bar graphs represent smatch agreement score for each batch. Dark gray bars indicate WISeN annotation scheme with a 0.776 average, and light gray bars indicate AMR annotation scheme with 0.735 average.

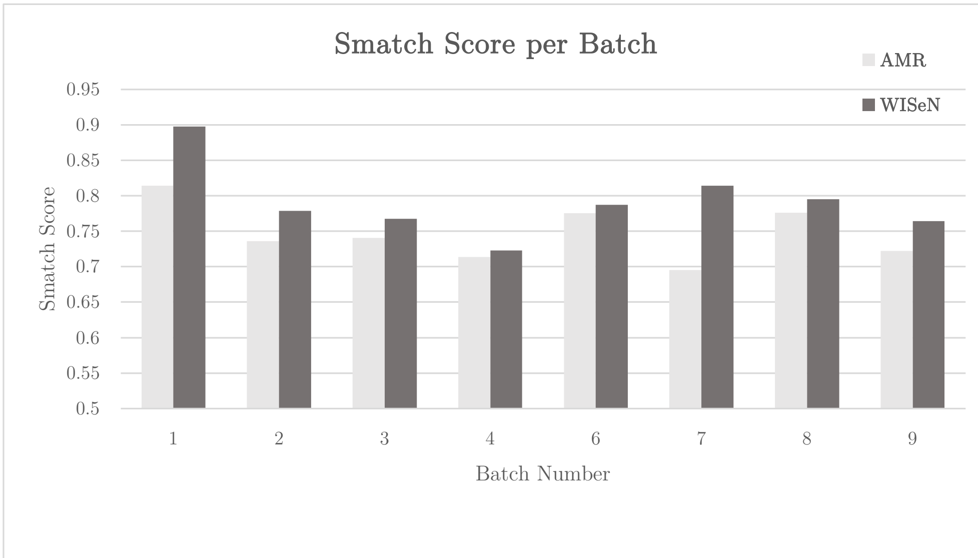

Figure 4.3: AMR and WISeN Smatch scores per batch

Batches 1-4 for WISeN and AMR were both part of round 1 of annotations. Batches 6-9 for WISeN and AMR were both part of round 2 of annotations. Batches 5 and 10 are not shown since these were singly-annotated.

4.1.2 Speed of Annotations

In addition to agreement, the speed of annotations was also measured. Annotators recorded the time it took them to complete each batch, for a total of six data points each for Annotators C-F, and four data points for Annotator H. The average of these times over AMR and WISeN for each annotator is summarized in Table 4.2.

AMR and WISeN speed can only be compared relative to the annotator's own performance, as some annotators generally work faster or slower than others. As such, we compared the average times of different batches in the

Annotator	AMR	WISeN
C	119.67	113.33
D	66.67	66.67
E	103.67	96.67
F	124.00	135.33
G	137.33	105.67
H	103.5	122.5

Table 4.2: Average minutes per batch for annotators in AMR and WISeN

order they were annotated. In other words, we examined how the average time for all annotator's first batch in a guideline scheme compared to their time for their second batch in that same guideline scheme, and finally their third batch. This allowed us to see if annotators improved with subsequent batches. Figure 4.4 summarizes this information.

4.2 Parsing Experiment

The AMR and WISeN parser were tested, and their performances were evaluated overall using Smatch [7], as well as on specific subtasks for fine-grained evaluation [9]. We remove the subtask of Named Entity Recognition, and Named Entity Recognition with Wikipedia, as we are not testing these tasks. What remains is six subtasks. **Unlabeled** is the Smatch score computed when edge labels are removed and only graph structure is compared. **No WSD** is the Smatch score computed when the parser allows matches between

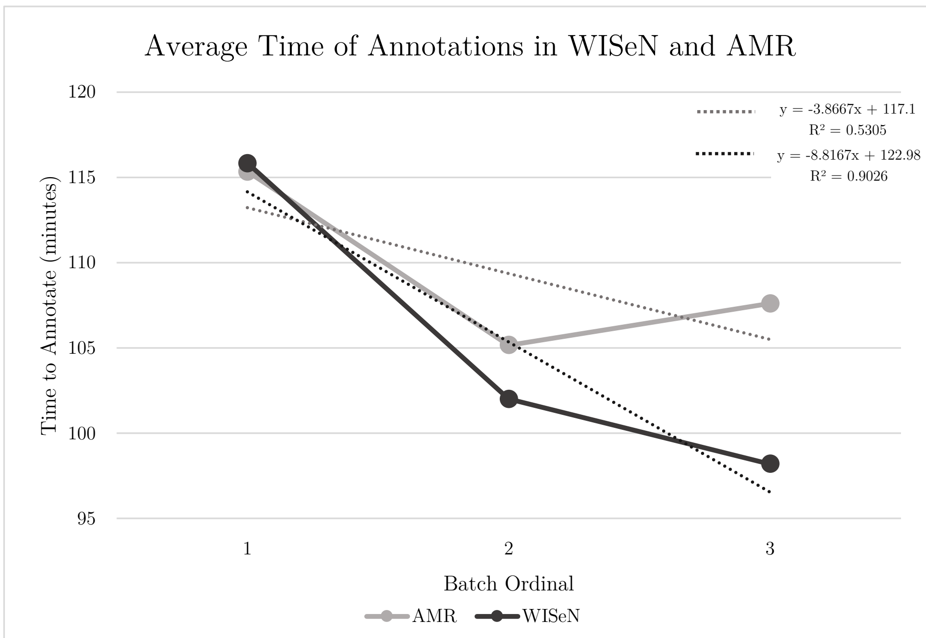

Figure 4.4: Average time in minutes of annotations in WISeN and AMR
The average of all of the annotator's first, second, and third batches in AMR and WISeN were taken. Annotator H is still included in the mean calculation for batch ordinal 1 and 2, even though they did not complete a third batch for either scheme. A linear regression shows a slope of -3.87 for AMR time and a slope of -8.82 for WISeN time. However, the AMR regression is not as linear with an $R^2 = 0.5305$*. Conversely, the WISeN times fit well to a linear model with an* $R^2 = 0.9026$*, as each subsequent batch of 50 sentences took less time than its preceding one*

PropBank frames with the same names but different sense arguments (e.g. matching `buy-01` and `buy-05`). **Concepts** is an F-1 score based on comparing the predicted concepts with the concepts that appear in the actual graph. **SRL** computes the Smatch score on only the numbered argument edges (e.g. `ARG0` - `ARG6`). This task is not evaluated for WISeN, since this task targets numbered arguments, of which there are none in this scheme. **Reentrancy** computes a Smatch score only on reentrant edges, where multiple edges point to the same node as a consequence of one node being involved in multiple relations. Lastly, **Negations** defines an F-1 score based on comparing the negated concepts in the predicted graph to the negated concepts in the actual graph.

The performance of the AMR and WISeN parser on these tasks is summarized in Table 4.3. The leftmost columns of Table 4.3 specify what parser was run and against which data it was evaluated.

These results show that the WISeN parser performs better than AMR parser on corresponding test sets. In most subtasks, the WISeN parser achieves a higher Smatch and F-1 score than the AMR parser. On the test split evaluation, we observe a 0.8% increase in overall Smatch score. This increase reaches 2.1% on the evaluations of the 1,000 dialogue sentence

Parser	Test	Smatch	Unlabeled	No WSD	Concepts	SRL	Reent.	Neg.
AMR	test split	75.2	78.1	75.7	86.9	73.8	55.7	73.5
	1k AMR	74.5	78.6	75.9	84.4	76.6	60	61.6
WISeN	test split	76.0	78.8	76.0	89.3	0	56	77.9
	1k converted	76.6	80.7	76.6	88.4	0	60.5	69.2
	1k true	75.8	80.8	75.8	88.5	0	59.7	67.8

Table 4.3: Performance of AMR and WISeN parsers

The AMR parser was tested on the AMR 3.0 corpus test split as well as the 1,000 AMR annotations produced from our dialogue sentences (1k AMR). The WISeN parser was tested on the test split from the converted WISeN corpus as well as the 1,000 WISeN annotations converted from our 1,000 AMR annotations (1k converted) and the 1,000 true WISeN annotations that we completed during this project (1k true).

annotations. The WISeN parser also improves negation labeling by 4.4% in the test split and 7.6% in the 1,000 dialogue annotations. Concept labeling also increased in F-1 score by 2.4% in the test split and 4.1% in the 1,000 dialogue annotations.

Chapter 5

Discussion

5.1 WISeN Annotation is More Accurate than AMR

The results of the annotation experiment showed a statistically significant increase in accuracy among annotators while using the WISeN annotation scheme over the AMR scheme ($p < 0.0005$) . In seven out of eight batches of annotations, the accuracy among annotators was higher with the WISeN guideline scheme than the AMR guideline scheme, regardless of the annotator's experience or familiarity with semantic annotation. This is especially important when we consider PropBank's frameset files that clearly and unambiguously outline the arguments of a predicate with lots of descriptions and examples. Therefore, the decision-making responsibilities are largely removed from the individual annotator while annotating in AMR. While we recognize that this resource is well-built and broad in its scope, we maintain that referencing these predicate-argument structures is unnecessary. Annotation can only be done if these frameset files are already in place, a costly and time-consuming resource to to build, and the results of the annotation experiment show that without them, annotators are able to perform just as well.

5.2 WISeN Annotation is Comparable in Speed to AMR

Looking now at speed of annotation, we see that the WISeN annotations consistently took less time than AMR annotations, again regardless of the annotator's experience or familiarity with semantic annotation. Though annotator's first batch of each scheme took roughly the same time, the speed of WISeN annotations picked up at a faster rate than AMR, fitting well to a linear regression with a slope of -8.82 minutes per each subsequent batch of 50 sentences. On the other hand, AMR annotations did not drastically speed up over time, and in fact, remained largely constant throughout the experiment. This could be attributed to the fact that the process of annotating in AMR is bound by referring to the PropBank frames. This process of checking

PropBank's frameset files introduces a minimum baseline annotation time, and checking these files does not speed up over time, no matter how well-versed someone may be in annotating. Conversely, WISeN relies on annotator intuition. Once annotators understand the guidelines, they can apply their linguistic knowledge to a variety of sentences, thus allowing them to get faster over time as they encounter more examples.

5.3 WISeN Improves Parser Performance

The results of the parsing experiment show that a parser trained on WISeN annotations performs better than a parser trained on AMR annotations in all relevant subtasks. This uniform increase in scores across subtasks implies that WISeN does not fix particular phenomena, but instead improves the model's overall performance. This is likely due to the more consistent usage of core roles. As noted prior, the numbered arguments of a particular predicate do not necessarily align with the numbered arguments of another. However, WISeN roles aim to remove this variability by utilizing only unambiguous semantic roles. Further, since text input does not arrive to the parser sense-disambiguated, this charges the parser with another layer of learning. These results show that prioritizing semantic roles and removing numbered arguments and sense-disambiguation provides better training data for a model and ultimately supports a more accurate parser.

5.4 Limitations

This work should be considered in the light of several limitations. First, the WISeN guidelines are not deterministic in the way that the PropBank predicate-argument structures are. The labeling of an argument as one semantic role over the other is a result of an annotator's intuition, and there may be multiple plausible answers. Take, for example, a sentence used in the corpus of our annotation experiment: *have you seen your doctor about it?* The WISeN annotation for this sentence is as follows.

Have you seen your doctor about it?

```
(s / see
   :actor (y / you)
   :theme (p / person
             :actor-of (h / have-rel-role
                          :attribute (d / doctor)
                          :theme y))
   :topic (i / it)
   :polarity (w / wisen-question))
```

In this example, we have annotated *it* as the `topic` of the seeing event. However, we could also make an argument that *it* is the `cause` of the seeing event, as the illness or infliction is what prompts the actor to seek out the doctor. Because these ambiguities are not resolved by the WISeN guidelines, annotators must rely on their own intuitions about language to make these judgements. As such, annotators may have had a more difficult and longer time determining which WISeN role to use, affecting the results of the annotation experiment.

Another limitation of this study is the way the AMR corpus was converted to WISeN in the parsing experiment in section 3.3. Although we made an effort to create specific and fine-grained rules for many of the arguments in PropBank, the conversion is nonetheless a generalization. Due to time constraints, it was not feasible to sift through each of the 15,000 numbered arguments by hand in order to determine the WISeN role that best fit it. Instead, we used an argument's function tag, VerbNet role, and description to make generalized rules that could be used to mass-change arguments. As such, the conversion from the AMR corpus to the WISeN corpus is not perfect.

Further, the WISeN corpus that resulted from converting the AMR corpus is not a representation of true WISeN. In addition to some possible flaws in the conversion between numbered arguments and WISeN roles, there are other aspects of the WISeN annotation scheme that could not be accounted for. As we have noted previously, WISeN is not bound by the existence of frameset files in PropBank. Because of this, predicates can be created ad-hoc. Since this is not a possibility in AMR, annotation structure may be driven by what is possible within the realm of the frameset files. Consider the sentence *that's great*. In PropBank, there is no predicate for the word *great*. As a result, the AMR annotation for this sentence is as follows.

```
That's great.
(g / great
   :domain (t / that))
```

In the converted WISeN, this graph would look identical, since the conversion is only concerned with changing numbered arguments. However, if we disregard a conversion from AMR, the WISeN annotation for this sentence may look something like the following.

```
That's great.
(g / great
   :theme (t / that))
```

This WISeN graph is not a result of a conversion from an AMR representation. As such, the converted WISeN outputs may differ from true WISeN in their structure. To see the extent of these differences, a Smatch score was computed between the 1,000 WISeN annotations that were obtained from converting AMR annotations and the 1,000 true WISeN annotations produced by our annotators. Converted WISeN and true WISeN obtained a Smatch score of 0.88, which, although relatively high, does point to a discrepancy between converted WISeN and true WISeN.

Next, our annotation experiment examined the Smatch scores for each sentence, and we averaged the Smatch scores over each batch and over all batches. This gives equal weights to all sentences, even though some sentences were longer and more complex and others were shorter and more simple. Thus, the Smatch scores computed for WISeN and AMR do not account for the varying difficulty of sentences and weights each Smatch score as the same.

Lastly, we had a small sample size for the annotation experiment, only using six annotators. Our results would be more generalizable if we had tested more annotators on WISeN and AMR to account for any anomalies in annotator skill.

5.5 Future Work

Future research should consider which thematic roles are most intuitive to annotators while still aiming to be able to cover all natural language.

Possible starting points include using PropBank's function tags as the totality of thematic roles in an annotation scheme to see if this set strikes a balance between intuition and coverage for annotators. In addition, conversion from one scheme to another might prove an important area for future research. As annotation is timely and costly, perfecting algorithms that could automatically convert large pre-existing corpora into different formats would provide a great tool to the field of natural language processing. Although we converted roles, we did not attempt to change any of the graphs structurally. This automation would prove quite useful for future work in this area.

5.6 Conclusions

The findings of this study show that WISeN is an annotation scheme that supports improved parser performance and improved inter-annotator agreement without sacrificing speed of manual annotations, even without a large database of frameset files detailing which role to use. From this, we can conclude that the removal of numbered arguments and sense-disambiguation in favor of semantic roles solves many of the problems associated with AMR and PropBank, thus making WISeN more intuitive for annotators and more interpretable to parsers. WISeN also requires less start-up costs than AMR, since frameset files for thousands of predicates do not need to be created for annotation to commence. As such, WISeN is a realistic annotation scheme for use in languages and domains not covered by PropBank. Because of its generalizability and improved parser performance while maintaining the speed and accuracy of annotation, we advocate for the adoption of WISeN as an annotation scheme.

Bibliography

[1] Mark Baker. Theta theory and the syntax of applicatives in chichewa. *Natural Language & Linguistic Theory*, 6(3):353–389, 1988. doi: 10.1007/BF00133903. URL `https://doi.org/10.1007/BF00133903`.

[2] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W13-2322`.

[3] Claire Bonial, Julia Bonn, Kathryn Conger, Jena D. Hwang, and Martha Palmer. PropBank: Semantics of new predicate types. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3013–3019, Reykjavik, Iceland,

May 2014. European Language Resources Association (ELRA). URL `http://www.lrec-conf.org/proceedings/lrec2014/pdf/1012`$_Paper.pdf$.

[4] Claire Bonial, Julia Bonn, Kathryn Conger, Jena D. Hwang, Martha Palmer, and Nicholas Reese. English propbank annotation guidelines, July 2015.

[5] Rebecca Bruce and Janyce Wiebe. Word-sense distinguishability and inter-coder agreement. In *Proceedings of the Third Conference on Empirical Methods for Natural Language Processing*, pages 53–60, Palacio de Exposiciones y Congresos, Granada, Spain, June 1998. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W98-1507`.

[6] Deng Cai and Wai Lam. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.119. URL `https://www.aclweb.org/anthology/2020.acl-main.119`.

[7] Shu Cai and Kevin Knight. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the As-*

sociation for Computational Linguistics (Volume 2: Short Papers), pages 748–752, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P13-2131`.

[8] Boston English Centre. Daily english conversations - 900 english conversaional sentences [video]. youtube.

[9] Marco Damonte, Shay B. Cohen, and Giorgio Satta. An incremental parser for Abstract Meaning Representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 536–546, Valencia, Spain, April 2017. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/E17-1051`.

[10] David Dowty. Thematic proto-roles and argument selection. *Language*, 67(3):547–619, 1991. ISSN 00978507, 15350665. URL `http://www.jstor.org/stable/415037`.

[11] Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. A discriminative graph-based parser for the Abstract Meaning Representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long*

Papers), pages 1426–1436, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1134. URL `https://www.aclweb.org/anthology/P14-1134`.

[12] A.E. Goldberg. *Constructions: A Construction Grammar Approach to Argument Structure*. Cognitive Theory of Language and Culture Series. University of Chicago Press, 1995. ISBN 9780226300863. URL `https://books.google.com/books?id=HzmGM0qCKtIC`.

[13] Michael Wayne Goodman. Penman: An open-source library and tool for AMR graphs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 312–319, Online, July 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.acl-demos.35`.

[14] Jia-Chen Gu, Zhen-Hua Ling, Xiaodan Zhu, and Quan Liu. Dually interactive matching network for personalized response selection in retrieval-based chatbots, 2020.

[15] Paul Kay and Charles J. Fillmore. Grammatical constructions and linguistic generalizations: The what's x doing y? construc-

tion. *Language*, 75(1):1–33, 1999. ISSN 00978507, 15350665. URL `http://www.jstor.org/stable/417472`.

[16] Laura Baranescu Claire Bonial Madalina Bardocz Kira Griffitt Ulf Hermjakob Daniel Marcu Martha Palmer Tim O'Gorman Nathan Schneider Kevin Knight, Bianca Badarau. Abstract meaning representation (amr) annotation release 3.0.

[17] Paul Kingsbury and Martha Palmer. From TreeBank to PropBank. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02)*, Las Palmas, Canary Islands - Spain, May 2002. European Language Resources Association (ELRA). URL `http://www.lrec-conf.org/proceedings/lrec2002/pdf/283.pdf`.

[18] Irene Langkilde and Kevin Knight. Generation that exploits corpus-based statistical knowledge. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 704–710, Montreal, Quebec, Canada, August 1998. Association for Computational Linguistics. doi: 10.3115/980845.980963. URL `https://www.aclweb.org/anthology/P98-1116`.

[19] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset. *CoRR*, abs/1710.03957, 2017. URL `http://arxiv.org/abs/1710.03957`.

[20] Edward Loper, Szu ting Yi, and Martha Palmer. Combining lexical resources: Mapping between propbank and verbnet. In *In Proceedings of the 7th International Workshop on Computational Linguistics*, 2007.

[21] Chunchuan Lyu and Ivan Titov. AMR parsing as graph prediction with latent alignment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 397–407, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1037. URL `https://www.aclweb.org/anthology/P18-1037`.

[22] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993. URL `https://www.aclweb.org/anthology/J93-2004`.

[23] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.

ISSN 0001-0782. doi: 10.1145/219717.219748. URL https://doi.org/10.1145/219717.219748.

[24] Hwee Tou Ng, Chung Yong Lim, and Shou King Foo. A case study on inter-annotator agreement for word sense disambiguation. In *SIGLEX99: Standardizing Lexical Resources*, 1999. URL https://www.aclweb.org/anthology/W99-0502.

[25] Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. I know the feeling: Learning to converse with empathy. *CoRR*, abs/1811.00207, 2018. URL http://arxiv.org/abs/1811.00207.

[26] Karin K. Schuler. Verbnet: A broad-coverage comprehensive verb lexicon. Master's thesis, University of Pennsylvania, 2005.

[27] J. Véronis. A study of polysemy judgements and inter-annotator agreement. 1999.

[28] Szu-ting Yi, Edward Loper, and Martha Palmer. Can semantic roles generalize across genres? In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 548–555,

Rochester, New York, April 2007. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/N07-1069`.

[29] Victor H. Yngve. Syntax and the problem of multiple meaning. In William N. Locke and A. Donald Booth, editors, *Machine Translation of Languages*, pages 208–226. MIT Press, Cambridge, MA, 1955.

Appendix A

WISeN Guidelines

A.1 Core roles

A.1.1 Actor

WISeN makes use of the `actor` relation to encompass the traditional thematic role of agent.

The boy wants the girl to believe him

```
(w / want
   :actor (b / boy)
   :theme (b2 / believe
              :actor (g / girl)
              :theme b))
```

However, the `actor` relation is less specific than a thematic agent. An agent must be intentional, while the `actor` relation may also include non-intentional doers. The `actor` role corresponds to the thing which is the impetus behind the event.

The bus hit the curb

```
(h / hit
   :actor (b / bus)
   :theme (c / curb))
```

The role `actor` is also used to annotate the subject of a communication verb.

The boy said that the bus crashed

```
(s / say
   :actor (b / boy)
   :theme (c / crash
             :theme (b / bus)))
```

Importantly, there is no one-to-one correspondence between the role of `actor` and the notion of grammatical subject. Firstly, a subject is not always an actor (See also `theme` theme and `benefactive` benefactive).

Secondly, there are `actor` arguments which are not always grammatical subjects. For instance, WISeN (following PropBank) treats the entity or event which instils an emotion in a `theme` to be an `actor`.

The boy is scared of the monkey
The monkey scares the boy

```
(s / scare
   :actor (m / monkey)
   :theme (b / boy))
```

Even when there is no transitive verbal form of the predicate (e.g., *afraid*) the `actor` is still the entity which instils the emotion in the `theme`. In the following sentence, the monkey is the impetus of the fear.

The boy is afraid of the monkey

```
(a / afraid
   :actor (m / monkey)
   :theme (b / boy))
```

As mentioned, emotive predicates may even have an eventive `actor`.

The boy is glad that the monkey left

```
(g / glad
   :actor (l / leave
             :actor (m / monkey))
   :theme (b / boy))
```

Finally, the subject of perception predicates (e.g., *see* and *hear*) is treated as an `actor` because it is doing the perceiving, even if unintentionally.[1]

The boy saw the horse in the garden

```
(s / see
   :actor (b / boy)
   :theme (h / horse
             :location (g / garden)))
```

A.1.2 Theme

WISeN does not distinguish between the thematic roles patient and theme. The role `theme` is used for arguments which either undergo an action or have some property.

The boy hugged the monkey

```
(h / hug
   :actor (b / boy)
   :theme (m / monkey))
```

A `theme` may also appear as the grammatical subject. For instance, in an unaccusative construction.

The vase broke

```
(b / break
   :theme (v / vase))
```

This retains its role in a causative construction when it occurs as the direct object and the `actor` is added as the grammatical subject.

[1] Those who are familiar with thematic roles might notice that we annotate thematic experiencers sometimes as a `theme` (as with emotive predicates like *afraid*) and sometimes as an `actor` (as with verbs of perception like *see*). Likewise, we sometimes annotate the so-called thematic stimulus as an `actor` (emotive predicates) and sometimes as a `theme` (verbs of perception). This is in keeping with PropBank, and we agree that it is the most natural way to annotate these constructions without introducing more relations.

The wind broke the vase

```
(b / break
   :actor (w / wind)
   :theme (v/ vase))
```

A less obvious case of a `theme` is the subject of a verb like intransitive *roll*.

The boy rolled down the hill

```
(r / roll
   :theme (b / boy)
   :direction (d / down)
   :path (h / hill))
```

Compare this to the following.

The girl rolled the boy down the hill

```
(r / roll
   :actor (g / girl)
   :theme (b / boy)
   :direction (d / down)
   :path (h / hill))
```

If it is clear from the context that the boy is the impetus behind the event, then the annotator can ascribe the concept `boy` both thematic relations.

The boy rolled down the hill on purpose

```
(r / roll
   :actor (b / boy)
   :theme b
   :direction (d / down)
   :path (h / hill)
   :manner (o / on-purpose))
```

An even more striking example is the verb *drive*.

The car drove west

```
(d / drive
   :theme (c / car)
   :direction (w / west))
```

The girl drove the car west

```
(d / drive
   :actor (g / girl)
   :theme (c / car)
   :direction (w / west))
```

As a rule of thumb, if the subject of an intransitive verb can also appear as the object when the verb is transitive, it is likely a `theme`.

The role `theme` is also used to annotate the message communicated by a communication verb.

The boy said that the bus crashed

```
(s / say
   :actor (b / boy)
   :theme (c / crash
             :theme (b / bus)))
```

As well as propositions which are embedded under a modal concept.[2]

The boy can ski
It is possible the boy is skiing
The boy might ski

```
(p / possible
   :theme (s / ski
             :actor (b / boy)))
```

[2]More discussion of modality is included in the AMR guidelines.

```
The boy must clean the house
The boy is obligated to clean the house
It's obligatory that the boy clean the house

(o / obligate
   :theme (c / clean
             :actor (b / boy)
             :theme (h / house)))
```

The theme relation is also used when an argument has the property described by the predicate.

```
The girl is tall

(t / tall
   :theme (g / girl))
```

```
The boy is glad

(g / glad
   :theme (b / boy))
```

A.1.3 Benefactive

The `benefactive` role is used when representing a number of constructions. Most notably, it is used to represent a recipient in a dative or double object construction.

```
The girl gave a book to her friend
The girl gave her friend a book

(g / give
   :actor (g2 / girl)
   :theme (b / book)
   :benefactive (f / friend
                   :poss g2)))
```

It is also used for some (but not all) other arguments introduced by prepositions such as *to* and *for* (See also `asset` asset and `purpose` purpose).

The girl sings to her cat
The girl sings for her cat

```
(s / sing
   :actor (g / girl)
   :benefactive (c / cat))
```

The role `benefactive` is used when the argument is either a recipient or an individual/organisation for whose benefit or detriment an action is done (i.e., they are benefited or harmed by the event).

The dice fell kindly for the girl

```
(f / fall
   :theme (d / dice)
   :manner (k / kind)
   :benefactive (g / girl))
```

The role `benefactive` is also used to annotate the addressee or hearer of a communication verb.

The boy said to the girl that the bus crashed

```
(s / say
   :actor (b / boy)
   :benefactive (g / girl)
   :theme (c / crash
             :theme (b / bus)))
```

The boy told the girl that it was raining

```
(t / tell
   :actor (b / boy)
   :benefactive (g / girl)
   :theme (r / rain))
```

The boy ordered the girl to clean her room

```
(o / order
   :actor (b / boy)
   :benefactive (g / girl)
   :theme (c / clean
             :actor g
             :theme (r / room
                       :poss g)))
```

As well as arguments of permission and obligation modals.

The girl permitted the boy to eat a cookie

```
(p / permit
   :actor (g / girl
   :benefactive (b / boy)
   :theme (e / eat
             :actor b
             :theme (c / cookie)))
```

The girl obligated the boy to clean the house

```
(o / obligate
   :actor (g / girl
   :benefactive (b / boy)
   :theme (c / clean
             :actor b
             :theme (h / house)))
```

Notice that a `benefactive` role should not be confused with the notion of a beneficiary as the `benefactive` argument may be negatively affected.

The girl laid a trap for the monkey

```
(l / lay
   :actor (g / girl)
   :theme (t / trap)
   :benefactive (m / monkey))
```

Some verbs (e.g., *receive*) seem like they should have a `benficative` subject. However, for the sake of maintaining consistency with both PropBank and VerbNet, we assign this subject the `actor` role.

The girl received a fine

```
(r / receive
   :actor (g / girl)
   :theme (g2 / gift))
```

A.1.4 Asset

In a bid to reduce verb specific arguments, WISeN makes use of the VerbNet relation `asset`. This is used with predicates which describe exchanges and transactions such as `buy`, `sell`, `offer`, `order`, as well as many others. We use the `asset` relation for any argument which moves in the opposite direction to the `theme` in an exchange.

The girl bought the axe for twenty dollars with a credit card
The girl bought the axe with a credit card for twenty dollars

```
(b / buy
   :actor (g / girl)
   :theme (a / axe)
   :asset (m / monetary-quantity
             :quant 20
             :unit (d / dollar)))
   :instrument (c / card
                 :mod (c2 / credit)))
```

Unlike the VerbNet role, the WISeN role `asset` is not restricted to monetary prices. For instance, verbs like `refund` or `rebate` have a `theme` which is typically a `monetary-quantity` while the `asset` is the thing exchanged in order to receive the money.

The cashier refunded the girl twenty dollars for the axe
The cashier refunded twenty dollars to the girl for the axe

```
(r / refund
   :actor (c / cashier)
   :benefactive (g / girl)
   :theme (m / monetary-quantity
            :quant 20
            :unit (d / dollar))
   :asset (a / axe))
```

A.1.5 Instrument

The `instrument` relation is used for arguments which describe a thing used in carrying out an action. In English, `instruments` are usually introduced by the preposition *with*.

The girl chopped the wood with the axe

```
(c / chop
   :actor (g / girl)
   :theme (w / wood)
   :instrument (a / axe))
```

Notice that, in the following example, `axe` is not an `instrument` of `chop` despite being used in the chopping event because it is not an argument of `chop`.

The girl used the axe to chop the wood

```
(u / use
   :actor (g / girl)
   :theme (a / axe)
   :purpose (c / chop
              :actor g
              :theme (w / wood)))
```

A.1.6 Topic

The `topic` relation is used to annotate the subject-matter of an entity or event.

The professor wrote about math

```
(w / write
   :actor (p / professor)
   :topic (m / math))
```

The professor of math taught at the university
The math professor taught at the university

```
(t / teach
   :actor (p / professor
             :topic (m / math))
   :location (u / university))
```

The employee emailed the president of human resources

```
(e / email
   :actor (e2 / employee)
   :theme (p / president
             :topic (r / resources
                       :mod (h / human))))
```

The problem with deregulation

```
(p / problem
   :topic (d / deregulation))
```

A.1.7 Manner

The `manner` relation is used for arguments which describe the way something happens. It provides an answer to the question: "how was it done?". Notice that, when using `manner`, we drop the *-ly* from the end of the adverb.

The boy sang beautifully

```
(s / sing
   :actor (b / boy)
   :manner (b2 / beautiful))
```

It can also represent the means by which something is done.

The boss decreased spending by shortening hours

```
(d / decrease
   :actor (b / boss)
   :theme (s / spending)
   :manner (s2 / shorten
                :actor b
                :theme (h / hours)))
```

WISeN also uses `manner` to represent arguments which AMR handles with a `medium` role. Note that this use of `manner` differs from `instrument`, as `instrument` relations describe the thing used, whereas this use of `manner` describes a more general means.

The girl talked in French

```
(t / talk
   :actor (g / girl)
   :manner (l / language :wiki "French_language"
               :name (n / name :op1 "French")))
```

The boy told the girl by email

```
(t / tell
   :actor (b / boy)
   :theme (g / girl)
   :manner (e / email))
```

A.1.8 Accompanier

The `accompanier` relation is used for arguments that accompany another in the event.

The nanny walked to town with the newborn

```
(w / walk
   :actor (n / nanny)
   :end (t / town)
   :accompanier (n / newborn))
```

This differs from an `actor` as the `accompanier` may not able to perform the event on its own.

The boy went to school with his backpack

```
(g / go
   :actor (b / boy)
   :end (s / school)
   :accompanier (b2 / backpack
                 :poss b))
```

A.2 Spatial

A.2.1 Location

The role `location` is used to represent constituents which describe where an event took place.

The man died in his house

```
(d / die
   :theme (m / man)
   :location (h / house
             :poss m))
```

The man died near his house

```
(d / die
   :theme (m / man)
   :location (n / near
             :op1 (h / house
                     :poss m)))
```

The man died between his house and the river

```
(d / die
   :theme (m / man)
   :location (b / between
                :op1 (h / house
                        :poss m)
                :op2 (r / river)))
```

The detective arrived at the scene of the crime

```
(a / arrive
   :theme (d / detective)
   :end (s / scene
           :location-of (c / crime)))
```

The `location` role can also be used for some verbal arguments.

The man fit three marshmallows in his mouth

```
(f / fit
   :actor (m / man)
   :theme (m2 / marshmallow
              :quant 3)
   :location (m3 / mouth
                 :part-of m))
```

A.2.2 Direction and Path

The relations `direction` and `path` can represent arguments and modifiers of verbs of movement. Either one of these relations may be present without the other. In the following example we know the `path` of the bouncing, but not the direction.

The ball bounced along the street

```
(b / bounce
   :theme (b / ball)
   :path (a / along
            :op1 (s / street)))
```

Similarly, we might know the `direction` but not the `path`.

The car drove west

```
(d / drive
   :theme (c / car)
   :direction (w / west))
```

Besides the cardinal directions (*north, south, east, west*), other typical directions include *up, down, back, left, right, through, over*, etc.[3]

A `direction` may also appear within a `path` argument.

The soldiers marched east along the road to Moscow

```
(m / march
   :actor (s / soldier)
   :direction (e / east)
   :path (a / along
            :op1 (r / road
                    :direction (c / city
                                  :wiki "Moscow"
                                  :name (n / name
                                           :op1
"Moscow")))))
```

A.2.3 Start and End

The relations `start` and `end` are generally used for changes in location (corresponding to the AMR roles `source` and `destination`) or changes in state. Examples of locational `start` and `end` are given below.

The monkey jumped from tree to tree

```
(j / jump
   :actor (m / monkey)
   :start (t / tree)
   :end (t2 / tree))
```

[3] For the sake of the annotation exercises we will not use the `wiki` role.

He drove west, from Houston to Austin

```
(d / drive
   :actor (h / he)
   :direction (w / west)
   :start (c / city :wiki ‘‘Houston’’
             :name (n / name :op1 ‘‘Houston’’))
   :end (c2 / city :wiki ‘‘Austin,␣Texas’’
            :name (n2 / name :op1 "Austin")))
```

They can also be used for more abstract directional arguments.

They are descended from royalty

```
(d / descend
   :theme (t / they)
   :start (r / royalty))
```

WISeN also uses `start` for initial states or materials in verbs of creation (i.e., the `material` role of VerbNet), and `end` for the thing created (i.e., the `product` role of VerbNet).

She cast the bronze into a statue
She cast a statue out of bronze

```
(c / cast
   :actor (h / he)
   :start (b / bronze)
   :end (s / statue))
```

She made a dress out of her curtains
She made her curtains into a dress

```
(m / make
   :actor (s / she)
   :start (c / curtains
              :poss s))
   :end (d / dress))
```

He folded the paper into a card
He folded a card out of the paper

```
(f / fold
   :actor (s / she)
   :start (p / paper)
   :end (c / card))
```

As well as certain verb specific arguments.

The monkey arranged the bananas from a neat stack into a messy pile

```
(a / arrange
   :actor (m / monkey)
   :theme (b / banana)
   :start (s / stack
             :mod (n / neat)
   :end (p / pile
           :mod (m2 / messy))
```

Annotators should also be careful with locative alternations. These involve a `theme` and an `end`.

He sprayed paint onto the wall
He sprayed the wall with paint

```
(s / spray
   :theme (p / paint)
   :end (w / wall))
```

He loaded hay onto the cart
He loaded the cart with hay

```
(l / load
   :theme (h / hay)
   :end (c / cart))
```

However, the `end` can also appear without the `theme`. So annotators should be particularly careful not to assign the `theme` role to the `end` here.

He sprayed the wall

```
(s / spray
   :end (w / wall))
```

He loaded the cart

```
(l / load
   :end (c / cart))
```

An annotator might also wonder whether we could also annotate a `benefactive` as an `end` in a transfer of possession verbs. For instance, in the following example.

The girl gave a dog to the boy

```
(g / give
   :actor (g / girl)
   :theme (d / dog)
   :benefactive (b / boy))
```

WISeN opts to prioritize `benefactive` above `end`. If the argument could best be described as a "recipient", as in this example, you should use `benefactive`.

Finally, it might be hard to tell the difference between an `end` and a `direction`.

The girl threw the pie at the boy

```
(t / throw
   :actor (g / girl)
   :theme (p / pie)
   :end (b / boy))
```

Typically, a `direction` is a word such as *up, down, left, right, north, south, east, west, over, under, through* etc. or a place like a country or city (See `direction` direction and path).

A.3 Temporal

A.3.1 Time

The `time` relation establishes when an event took place.

The robbery happened yesterday

```
(r / robbery
   :time (y / yesterday)
```

The bridge was built in December

```
(b / build
   :theme (b2 / bridge)
   :time (d / date-entity
            :month 12))
```

It can also be used for relative time.

The woman had just eaten lunch

```
(e / eat
   :actor (w / woman)
   :theme (l / lunch)
   :time (r / recent))
```

In addition, the `time` relation can equate the time of two events.

The woman frowned when the baby cried

```
(f / frown
   :actor (w / woman)
   :time (c / cry)
            :actor (b / baby)))
```

A.3.2 Duration

The `duration` relation describes the amount of time over which an event occurs.

He worked for two hours yesterday

```
(w / work
   :actor (h / he)
   :duration (t / temporal-quantity
                :quant 2
                :unit (h2 / hour))
   :time (y / yesterday))
```

The investigator searched for a long time

```
(s / search
   :actor (i / investigator)
   :duration (l / long)
```

The athlete finished the marathon in two hours

```
(f / finish
   :actor (a / athlete)
   :theme (r / run)
            :actor a
            :theme (m / marathon)
            :duration (t / temporal-quantity
                         :quant 2
                         :unit (h / hour)))
```

A.3.3 Frequency

The `frequency` relation describes how often something occurs.

The phone rang three times

```
(r / ring
   :theme (p / phone)
   :frequency 3)
```

It can also be used to represent quantificational temporal adverbs.

She always eats breakfast

```
(e / eat
   :actor (s / she)
   :theme (b / breakfast)
   :frequency (a / always))
```

A.3.4 Range

The `range` relation is used to describe a period of time over which an event occurs. This is different from `duration`, because it does not measure the length of the event. Rather, it establishes a period of time in which the event occurs.

His first drink in 3 years

```
(d / drink
   :actor (h / he)
   :ord (o / ordinal-entity
           :value 1
           :range (t / temporal-quantity
                     :  quant 3
                     :  unit (y / year))))
```

Notice in the next example that if we had used the `duration` role, the sentence would mean *"it did not snow for 10 years"*, which is compatible with it having snowed for 9 years.

It had not snowed in ten years

```
(s / snow
   :polarity -
   :range (t / temporal-quantity
             :quant 10
             :unit (y / year)))
```

A.4 Causal/Conditional/Concessive

A.4.1 Cause

The `cause` role is typically used for causal adverbial clauses such as *because* clauses. The `cause` role is used to annotate an answer to the question "why did the event happen?".

The wind broke the vase because it was fragile

```
(b / break
   :actor (w / wind)
   :theme (v / vase)
   :cause (f / fragile)
             :theme v))
```

A `cause` is one of two ways of representing the notion of a "reason" in WISeN, (See also `purpose` purpose). In the following sentence, there are two reasons the judge sentenced the man.

The judge sentenced the man for speeding because he looked shifty

```
(s / sentence
   :actor (j / judge)
   :theme (m / man)
   :cause (a / and
             :op1 (s2 / speed
                     :actor m)
             :op2 (s3 / seem
                     :theme (s4 / shifty
                                :theme m))))
```

WISeN also uses the inverse `cause-of` relation to represent some result states.[4]

[4]Notice we use `cause-of` instead of `end` here, since *pieces* is not something which is made out of the vase (i.e. a product). Moreover, it is not a grammatical argument of *break*. Finally, it does not take part in the material/product alternation which is indicative of the `end` relation.
i. He folded the paper into a card / He folded a card out of the paper
ii. He broke the vase into pieces / *He broke pieces out of the vase

The vase broke into pieces

```
(b / break
   :theme (v / vase)
   :cause-of (i / in-pieces
                 :theme v))
```

He painted the house green

```
(p / paint
   :actor (h / he)
   :theme (h2 / house)
   :cause-of (g / green
                 :theme h))
```

The soldiers marched themselves tired

```
(m / march
   :actor (s / soldier)
   :cause-of (t / tired
                 :theme s))
```

A.4.2 Purpose

The role `purpose` is used to annotate an answer to the question "why was the event done?". A `purpose` is one of two ways to represent a "reason" in WISeN, (See also `cause` cause). In contrast to a `cause`, a `purpose` always follows the event.

She works for a living

```
(w / work
   :actor (s / she)
   :purpose (l / living))
```

She works to improve her life

```
(w / work
   :actor (s / she)
   :purpose (i / improve
               :theme (l / life
                          :poss h)))
```

A physical object may also have a `purpose`.

She found a trap for catching monkeys

```
(f / find
   :actor (s / she)
   :theme (t / trap
             :purpose (c / catch
                         :theme (m / monkey))))
```

A.4.3 Condition

The `condition` role is used for introducing an *if*-clause.

We will stay home if it rains

```
(s / stay
   :theme (w / we)
   :location (h / home)
   :condition (r / rain))
```

In combination with `polarity`, it can be used to represent an *unless* clause.[5]

[5]The AMR guidelines incorrectly places the negative polarity directly under the root concept, rather than embedded within the `condition`. Our example shows that this is incorrect. Consider the following sentences.

i. We will win the tournament unless we lose the final game.
ii. We won't win the tournament if we lose the final game.
iii. We will win the tournament if we don't lose the final game.

Both (ii) and (iii) would be true if (i) is true. However, (ii) would be true even if we cannot win the tournament with a draw. But (i) and (iii) would be false. This shows that (i) is closer in meaning to (iii) than (ii).

We will win the tournament unless we lose the final game

```
(w / win
   :actor (w2 / we)
   :theme (t / tournament)
   :condition (l / lose
                 :polarity -
                 :actor (w2 / we)
                 :theme (g / game
                           :mod (f / final))))
```

It can also represent an unconditional *whether or not* clause.

We will go to the park whether it rains or not

```
(g / go
   :actor (w / we)
   :end (p / park)
   :condition (o / or
                 :op1 (r / rain)
                 :op2 (r2 / rain
                          :polarity -)))
```

These clauses are often fronted. In which case, use the inverse `condition-of` (See also `cause` cause).

If it rains, we will stay home

```
(r / rain
   :condition-of (s / stay
                    :theme (w / we)
                    :location (h / home)))
```

A.4.4 Concession

WISeN uses the role `concession` in the same way as AMR. It is used to represent concessive connectives such as *although* and *despite*.

The game continued although it rained
The game continued despite the rain

```
(c / continue
   :theme (g / game)
   :concession (r / rain))
```

These clauses are often fronted, in which case you can use the inverse `concession-of`.

Although it rained, the game continued
Despite the rain, the game continued

```
(r / rain
   :concession-of (c / continue
                     :theme (g / game)))
```

Sometimes *but* is used concessively (see also `comparison` comparison for contrastive uses of *but*).

Trade has developed rapidly but it still has potential

```
(d / develop
   :theme (t / trade)
   :manner (r / rapid)
   :concession-of (h / have
                     :actor t
                     :theme (p / potential)
                     :mod (s / still)))
```

A.5 Mereology and Degrees

A.5.1 Domain and Mod

The roles `domain` and `mod` are inverses. The former is typically used in noun-copula-noun constructions.

They are birds

```
(b / birds
   :domain (t / they))
```

As well as in small clauses.

I consider him a friend

```
(c / consider
   :actor (i / i)
   :theme (f / friend
             :domain (h / he))
```

They are considered traitors

```
(c / consider
   :theme (p / person
             :domain (t / they)
             :actor-of (b / betray)))
```

The role `mod` is typically used for nominal modifiers such as adjectives.

Vice president

```
(p / president
   :mod (v / vice))
```

As well as relative clauses in which the main predicate is a noun (i.e., when you need to use the inverse of `domain`).

The man who is a lawyer

```
(m / man
   :mod (l / lawyer))
```

It is important to note, however, that `mod` is not used for all adjectives. Since the concept `toy` could be a `theme` of the concept `new` (not `domain`) we use the inverse of `theme`, `theme-of`, not `mod`.

The new toy

```
(t / toy
   :theme-of (n / new))
```

Likewise for `weather` and `cold`.

The cold weather

```
(w / weather
   :theme-of (c / cold))
```

Consider also the following more complicated example.

My favorite dog
The dog I favor

```
(d / dog
   :theme-of (f / favor
                :actor (i / i)))
```

A.5.2 Attribute

WISeN introduces the `attribute` role to account for a number of verb specific arguments, as well as providing a more intuitive description for some existing roles. The role `attribute` is used to annotate an argument which answers the question "In what respect does an argument have, or change in, the property described?".

Oftentimes, the `attribute` can appear redundant.

The man is short in stature

```
(s / short
   :theme (m / man)
   :attribute (s2 / stature))
```

The popcorn was free of charge

```
(f / free
   :theme (p / popcorn)
   :attribute (c / charge))
```

But this is not always the case.[6]

The man grew in courage

```
(g / grow
   :theme (m / man)
   :attribute (c / courage))
```

The man is rich in spirit

```
(r / rich
   :theme (m / man)
   :attribute (s / spirit))
```

Silver's advance in price

```
(a / advance
   :theme (s / silver)
   :attribute (p / price))
```

Attributes are commonly introduced by the prepositions *as* and *in*, and they add more specific information about some feature of one of the arguments (typically the `theme`). This includes non-result state secondary predicates.

The woman was accredited as an expert

```
(a / accredited
   :theme (w / woman)
   :attribute (e / expert))
```

The girl was denounced as a fraud

```
(d / denounce
   :theme (g / girl)
   :attribute (f / fraud))
```

[6]For the sentence *the man is rich in spirit*, PropBank would give *the man* the thematic role goal, and *spirit* the thematic role theme.

The girl employed the boy as a cleaner

```
(e / employ
   :actor (g / girl)
   :theme (b / boy)
   :attribute (p / person
                 :actor-of (c / clean)))
```

Lying counts as a sin

```
(c / count
   :theme (l / lie)
   :attribute (s / sin))
```

A.5.3 Quantity

The relation `quantity` is used to annotate numerical amounts.

Three boys passed the exam

```
(p / pass
   :actor (b / boy
             :quant 3)
   :theme (e / exam))
```

Several hundred apples

```
(a / apples
   :quant (s / several
             :op1 100))
```

Four out of five investors lost money

```
(l / lost
   :actor (p / person
             :actor-of (i / invest)
             :quant 4
             :subset-of (p2 / person
                            :actor-of (i2 / invest)
                            :quant 5))
   :theme (m / money))
```

It is also used to specify distance quantities and temporal quantities (See `extent` extent, `duration` duration and `range` range).

A.5.4 Degree

The `degree` role is used to introduce intensifiers like *very*, and *extremely* as well as "downtoners" like *somewhat* and *relatively*.

The girl is very tall

```
(t / tall
   :theme (g / girl)
   :degree (v / very))
```

The girl is too tall

```
(t / tall
   :theme (g / girl)
   :degree (t / too))
```

It is also used in comparatives and superlatives (See also `comparison` comparison).

The girl is the best

```
(g / good
   :theme (g2 / girl)
   :degree (m / most))
```

A.5.5 Extent

The role `extent` is not to be confused with `degree`. This role is often used to quantify a predicate.

The road goes on forever

```
(g / go-on
   :theme (r / road
   :extent (f / forever))
```

The boy grew 3 inches

```
(g / grow
   :theme (b / boy
   :extent (d / distance-quantity
              :unit (i / inches
                       :quant 3)))
```

We will also use this relation to introduce a measure phrase in comparative constructions (See also `comparison` comparison).

A.5.6 Comparison

The annotations in the AMR 3.0 corpus follow the suggestions in [**?**]. As such, we adopt these suggestions for now, modulo the discarding of numbered `ARG`s.[7] Comparatives are represented using a reification of the `degree` relation, `have-degree`. Since WISeN does not use numbered `ARG`s, we introduce a `comparison` relation. The `comparison` role is given to arguments which something is being compared to or contrasted with.

The girl is taller than the boy
The girl is taller than the boy is

```
(h / have-degree
   :theme (g / girl)
   :attribute (t / tall)
   :degree (m / more)
   :comparison (b / boy))
```

[7] WISeN aims to seek potential improvements on this work in the future.

A full list of the relations used are as follows.[8]

`theme`	entity characterized by attribute
`attribute`	attribute (e.g. tall)
`degree`	degree itself (e.g. more/most, less/least, equal)
`comparison`	compared-to
`comparison`	reference to superset
`comparison`	consequence, result of degree

Table A.1: List of arguments for `have-degree`

Below is an example of a superlative with a `comparison` argument.

The girl is the tallest of her friends

```
(h / have-degree
   :theme (g / girl)
   :attribute (t / tall)
   :degree (m / most)
   :comparison (f / friend
                 :poss g))
```

The following is an example of a 'degree consequence' construction.

The girl is too tall to sit comfortably

```
(h / have-degree
   :theme (g / girl)
   :attribute (t / tall)
   :degree (t2 / too)
   :comparison (s / sit
                 :theme g
                 :manner (c / comfort)))
```

Notice that the above sentence would typically be said when the girl is unable to sit comfortably (i.e., the consequence clause is non-veridical). However,

[8] Notice that we collapse three of **?** 's numbered `ARG` roles into one `comparison` role. We do this for several reasons: (i) they are all responsible for introducing a point of comparison, (ii) they never co-occur, (iii) the choice of numbered `ARG` depends entirely on the value of the `degree` role. As such, we may be able to get away with assuming that interpretation of `comparison` simply depends on the value of the `degree` (e.g., `more`, `most`, `too`, etc.).

rather than inserting negation or a modal concept here, [**?**] leave this representation as it is.[9] In later versions, WISeN aims to make improvements in this respect.

Finally, we use `comparison` for certain verbal arguments, such as the second prototypical patient/theme argument assigned by PropBank to the verb *correlate*.

Life expectancy correlates with wages

```
(c / correlate
   :theme (e / expect
             :theme (l / live))
   :comparison (w / wage))
```

Likewise, PropBank assigns the first argument of *similar* an agent role and the second a patient/theme role. However, neither argument can reasonably be called an agent. The addition of `comparison` allows us to rectify this.

The girl is similar to the boy in height

```
(s / similar
   :theme (g / girl)
   :comparison (b / boy)
   :attribute (h / height))
```

We also use comparison for arguments of `contrast` and contrastive connectives such as *but* (following the annotation of contrastive *but* in AMR 3.0).

[9] They note that sentences such as *the man was too drunk to drive* do not always entail that the man didn't drive.

The boy likes it, but the girl does not.

```
(l / like
   :actor (b / boy)
   :theme (i / it)
   :theme-of (c / contrast
                :comparison (l2 / like
                                :actor (g / girl)
                                :theme i
                                :polarity -)))
```

A.5.7 Possession

The `poss` relation is used to represent ownership or possession.

He loved his children

```
(l / love
   :actor (h / he)
   :benefactive (c / children
                   :poss h))
```

Note that `poss` is different from `part-of`, as it shows ownership not the relationship between two parts of one thing.

The sailor's boat

```
(b / boat
   :poss (s / sailor))
```

The boat's sail

```
(s / sail
   :part-of (b / boat))
```

A.5.8 Part-of and Consist-of

We use `consist-of` to represent the substance which an instance of a concept is composed from.

The gold watch

```
(w / watch
   :consist-of (g / gold))
```

We can also use it to cover some verb specific roles such as that of `compose`.[10]

The team is composed of players

```
(c / compose
   :theme (t / team)
   :consist-of (p / player))
```

This can be read ‘the composition of the team consists of players’.

The `part-of` relation is used when representing a part of an entity.

The engine of the car
The car’s engine

```
(e / engine
   :part-of (c / car))
```

The boy’s leg

```
(l / leg
   :part-of (b / boy))
```

The south of France

```
(s / south
   :part-of (c / country :wiki "France"
               :name (n / name :op1 "France")))
```

[10]This receives the vn-role ‘material’ in the PropBank frame.

A.5.9 Subevent

The `subevent` relation is used to describe the larger event of which the event in question is a part. It is often introduced with the phrase *in which*.

A massive bombardment in which 300 missiles rained on the capital

```
(b / bombard
   :mod (m / massive)
   :subevent (r / rain
                :theme (m / missiles
                          :quant 300)
                :location (c / capital)
                :direction (d / down))))
```

It contextualizes the event as part of an overarching event.

The speakers left on the final day of the conference

```
(l / leave
   :actor (s / speakers)
   :time (d / day
            :mod (f / final)
            :subevent (c / conference)))
```

A.6 Operators

The WISeN roles described in this section are adopted wholesale from the AMR guidelines. Annotators with experience converting text into AMR can safely skip this section.

A.6.1 Op

As in AMR, `opx` roles are used in conjunctions and disjunctions.

The boy and the girl swam

```
(s / swim
   :actor (a / and
             :op1 (b / boy)
             :op2 (g / girl))
```

As well as in spatial and temporal arguments.

The boy sang 10 minutes ago

```
(s / sing
   :actor (b / boy)
   :time (b2 / before
             :op1 (n / now)
             :quant (t / temporal-quantity
                       :unit (m / minutes)
                       :quant 10)))
```

And for named entities.

The Titanic

```
(s / ship
   :wiki "RMS_Titanic"
   :name (n / name)
            :op1 "Titanic"))
```

For more uses of `opx`, refer to the AMR guidelines.

A.6.2 Polarity

The `polarity` relation is used to evaluate the logical truth value of the statement and can be used to negate sentences. This relation is a binary value.

The boy doesn't go

```
(g / go
   :actor (b / boy)
   :polarity -)
```

This role negates the predicate under which it is immediately nested. Consider the following example in contrast to the first.

It is not the boy who goes

```
(g / go
   :actor (b / boy)
             :polarity -)
```

A.6.3 Polite

The `polite` role is used to annotate politeness markers. This role has a binary value.

We'd ask you to please leave

```
(a / ask
   :actor (w / we)
   :benefactive (y / you)
   :theme (l / leave
             :actor y
             :polite +))
```

A.6.4 Mode

The `mode` role describes the mood of the sentence and the intentions of the speakers. It can mark an imperative.

Let's go!

```
(g / go
   :actor (w / we)
   :mode imperative)
```

Wait here!

```
(w / wait
   :actor (y / you)
   :location (h / here)
   :mode imperative)
```

Or an expressive.

Wow!

```
(w / wow
   :mode expressive)
```

A.6.5 Example

The `example` role introduces something which is an example of a concept

The family vacations in resort spots like the beach

```
(v / vacation
   :actor (f / family)
   :location (s / spots
                :mod (r / resort)
                :example (b / beach)))
```

I like music such as country and rock

```
(l / like
   :actor (i / i)
   :theme (m / music
             :example (a / and
                        :op1 (c / country)
                        :op1 (r / rock))))
```

A.6.6 Name

The `name` role provides a concept's name.

The family's dog Snoopy barked

```
(b / bark
   :actor (d / dog
             :poss (f / family)
             :name (n / name
                      :op1 "Snoopy")))
```

A.6.7 Age

The `age` role provides an entity's age.

The 38 year old man injured his leg

```
(i / injure
   :actor (m / man
             :age (t / temporal-quantity
                     :quant 38
                     :unit (y / year)))
   :theme (l / leg
             :part-of m))
```

A.6.8 Value and Ord

The role `value` is used for specifying the numerical value of an entity.

Ninety-nine percent
99%

```
(p / percentage-entity
   :value 99)
```

While `ord` is used for ordinal numbers (i.e., 1st, 2nd, 3rd, etc.)

The second planet

```
(p / planet
   :ord (o / ordinal-entity
           :value 2))
```

A.6.9 Unit

The `unit` relation is used, often with `quantity`, to denote the measurement of a quantity.

She had planned her wedding for ten years

```
(p / plan
   :actor (s / she)
   :theme (w / wedding)
   :duration (t / temporal-quantity
                 :quantity 10
                 :unit (y / year)))
```

Units also don't have to be scientifically measured units.

a dozen bottles of water

```
(w / water
   :quantity (d / dozen)
   :unit (b / bottle))
```

We also must be explicit about what we are measuring when we use units. In the below example, without the `weight-quantity` predicate the meaning representation would be under specified.

The couple bought 4 pounds of rice

```
(b / buy
   :actor (c / couple)
   :theme (r / rice
             :quantity (w / weight-quantity
                          :quantity 4
                          :unit (p / pound)))
```

Similarly, we use `x-quantity` for other measurements such as volume for mass nouns.

The couple bought 2 gallons of milk

```
(b / buy
   :actor (c / couple)
   :theme (m / milk
             :quantity (v / volume-quantity
                          :quantity 2
                          :unit (g / gallon)))
```

A.6.10 List

The `list` relation is used to enumerate a list of items.

She believed she lived in the best city- one, everyone was friendly; two, the weather was perfect; and three, the food was delicious

```
(m / multi-sentence
   :snt1 (b / believe
            :actor (s / she)
            :theme (r / reside
                      :theme s
                      :location (c / city
                                   :mod (b / best))))
   :snt2 (f / friendly
            :list 1
            :actor-of (e / everyone))
   :snt3 (p / perfect
            :list 2
            :theme-of (w / weather))
   :snt4 (d / delicious
            :list 3
            :theme-of (f / food)))
```

A.7 Questions

WISeN uses the question tag `wisen-question` to denote questions. For yes/no questions, `wisen-question` is used in conjunction with the `polarity` relation to show that the truth value is in question.

Did the boy eat lunch?

```
(e / eat
   :actor (b / boy)
   :theme (l / lunch)
   :polarity (w / wisen-question))
```

Does the teacher read a lot?

```
(r / read
   :actor (p / person
             :actor-of (t / teach))
   :frequency (f / frequent)
   :polarity (w / wisen-question))
```

For *wh*-questions such as those containing *who*, *what*, *when*, *where*, *why*, and *how*, `wisen-question` is used in the *wh*-item's argument position (e.g., *the boy ate what?*).

What did the boy eat?

```
(e / eat
   :actor (b / boy)
   :theme (w / wisen-question))
```

How fast did the athlete run?

```
(r / run
   :actor (a / athlete)
   :manner (f / fast
              :degree (w / wisen-question)))
```

Whose toy did the girl find?

```
(f / find
   :actor (g / girl)
   :theme (t / toy
             :poss (w / wisen-question)))
```

Why did the baby cry?

```
(c / cry
   :actor (b / baby)
   :cause (w / wisen-question))
```

For choice questions, we use `wisen-choice` to denote options.

Do you want tea or coffee?

```
(w / want
   :actor (y / you)
   :theme (w2 / wisen-choice
              :op1 (t / tea)
              :op2 (c / coffee)))
```

Did the teacher walk or did she drive to school?

```
(s / school
   :end-of (w / wisen-choice
              :op1 (w2 / walk
                       :actor (g / girl))
              :op2 (d / drive
                      :actor t))))
```

Did the man win or lose the lottery?

```
(m / man
   :actor-of (w / wisen-choice
                :op1 (w2 / win
                         :theme (l / lottery))
                :op2 (l2 / lose
                         :theme l)))
```

A.8 Relative Clauses

Relative clauses are represented with inverse roles.

The boy who wore red sang at the concert

```
(s / sing
   :actor (b / boy
             :actor-of (w / wear
                          :theme (r / red)))
   :location (c / concert))
```

The main predicate in this sentence is *sing* which therefore forms the root of our annotation. The predicate *wear red* is then introduced with the inverse relation `actor-of`.

The man saw the executive that moved into the large office

```
(s / see
   :actor (m / man)
   :theme (e / executive
             :actor-of (m / move
                          :end (o / office
                                  :mod (l / large)))))
```

Note that the information about the executive moving into a large office is used to identify the person that the man hates. In this sentence, the man saw the executive. In contrast, the following sentence does not involve a relative clause.

The man saw that the executive that moved into the large office

```
(s / see
   :actor (m / man)
   :theme (m / move
             :actor (e / executive)
             :end (o / office
                     :mod (l / large))))
```

For this sentence to be true, the man need not directly see the executive. It is sufficient that he sees evidence that the executive is the new occupant of the large office.

A.9 Have-rel-role and have-org-role

WISeN follows AMR in using special predicate to attribute certain roles to people. For instance, a person who stand in a certain professional or personal relation to another.

she is my doctor

```
(h / have-rel-role
   :actor (s / she)
   :theme (i / i)
   :attribute (d / doctor))
```

`actor`	person who has role
`theme`	with whom
`attribute`	the relation

Table A.2: List of arguments for `have-rel-role`

My girlfriend swims

```
(s / swim
   :actor (p / person
             :actor-of (h / have-rel-role
                          :theme (i / i)
                          :attribute (g / girlfriend))))
```

Other examples of `have-rel-role` include: *father, sister, husband, grandson, godfather, stepdaughter, brother-in-law, friend, boyfriend, buddy, enemy, landlord, tenant* etc.

We use a similar structure for `have-org-role`.

`actor`	person who has role
`theme`	organization
`attribute`	the role

Table A.3: List of arguments for `have-org-role`

She is the company president

```
(h / have-org-role
   :actor (s / she)
   :theme (c / company)
   :attribute (p / president))
```