# Building a Framework to Generate Semantic Representations

## Final Presentation

University of Bonn

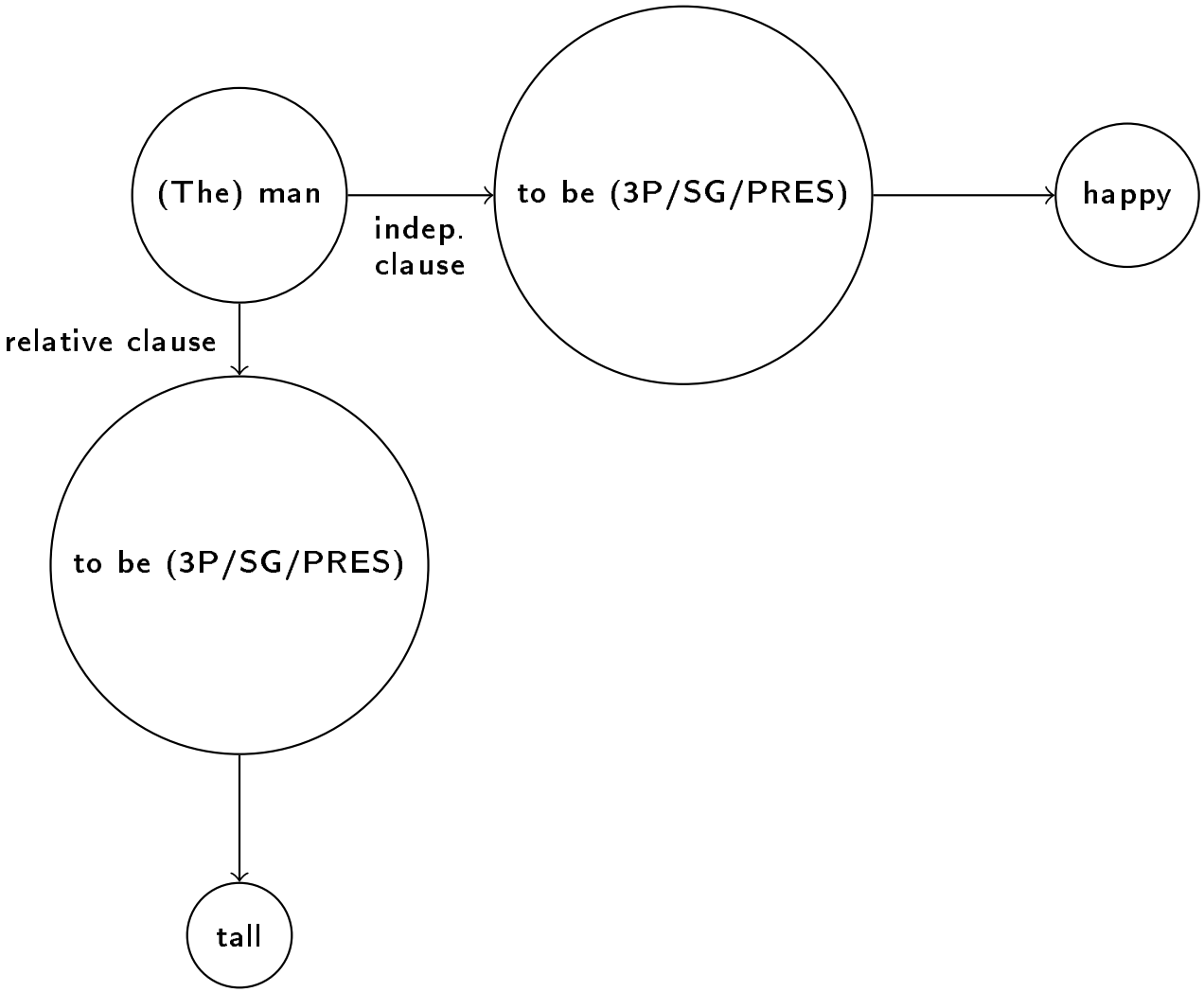
07.08.2018

# Recap

- preparation for a bachelor thesis related to machine translation
- build a graph-based meaning representation (MR)
  - as 'interlingua' for machine translation
  - to allow more sophisticated search in texts
- build a basic framework
  - to parse german sentences into the MR
  - to generate german sentences from MR graphs

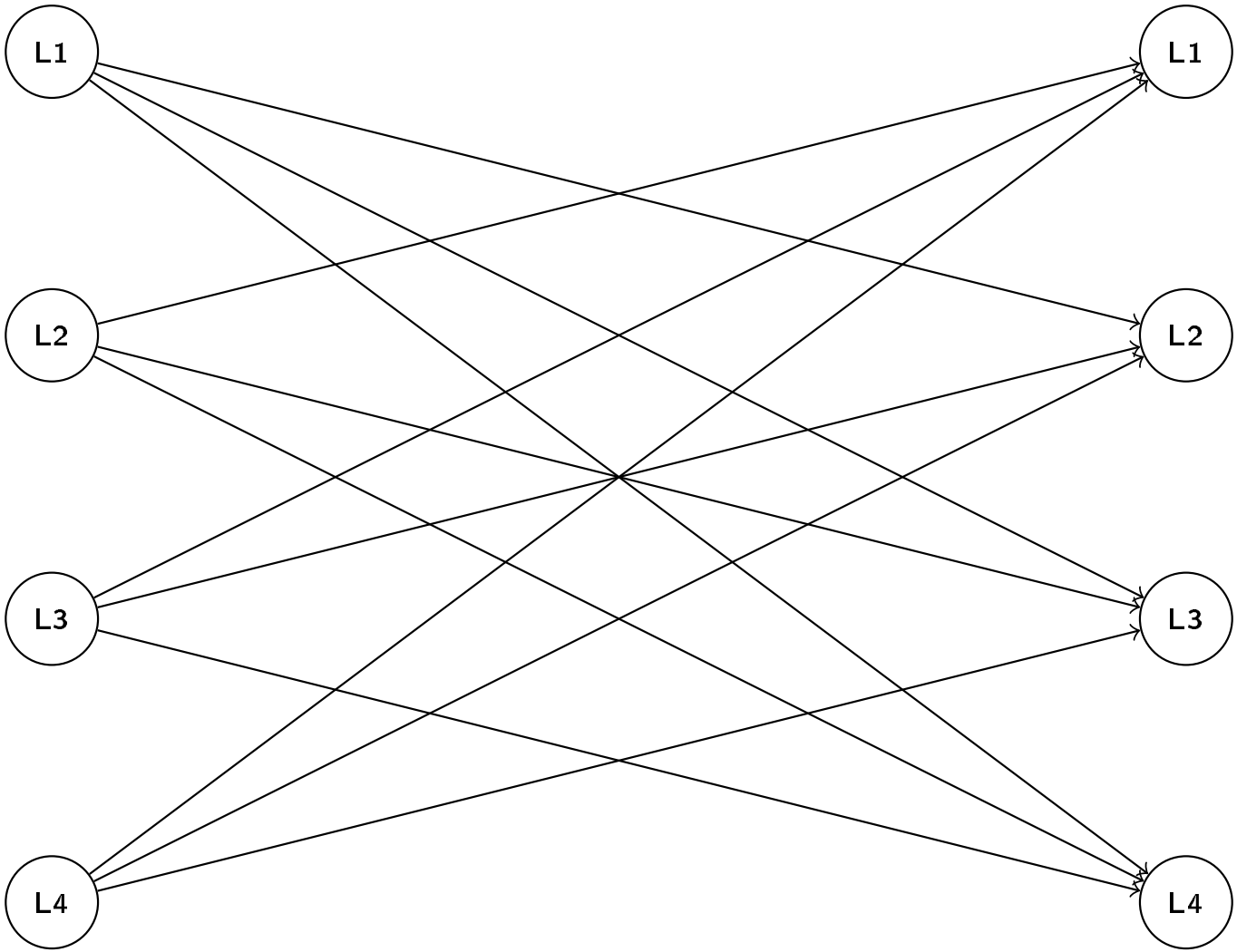
# Example

(1) The man who is tall is happy.



# Standard Approach: Machine Translation

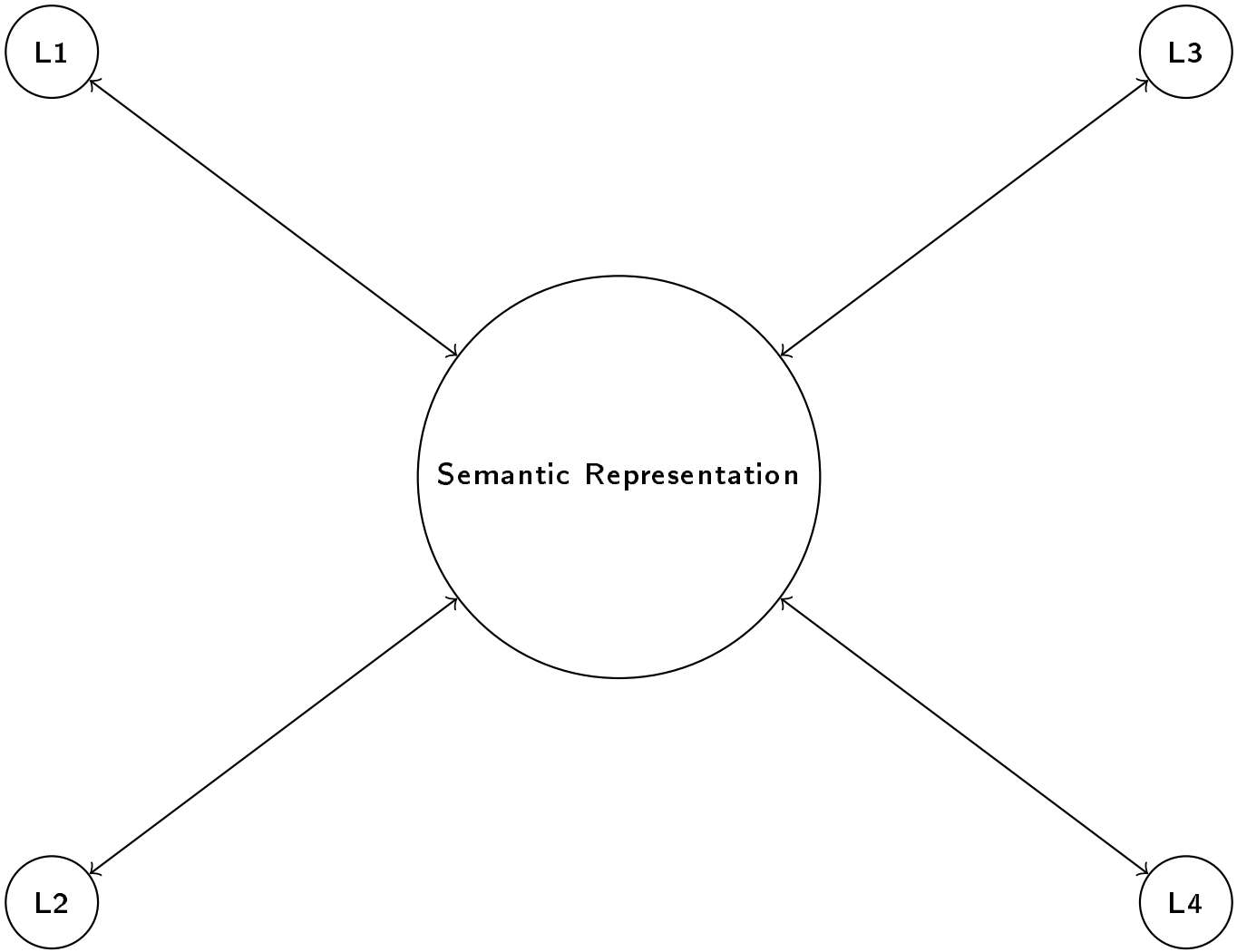
Source/Target Language Pairs




- for $n$ languages one has $n(n-1)$ source/target language pairs

# Interlingual Approach: Machine Translation

Source/Target Language Pairs




- for $n$ languages one has $2n$ source/target language pairs

# Lemmatizer/Stemmer

## Building a Semantic Representation

- one of the first step is to identify the given words in the sentence
  - identify their lemma (dictionary form)
  - identify the form if it is an inflected form
- stemmer
  - input: lemma, form
  - output: inflected form
- lemmatizer
  - input: inflected form
  - output: lemma, form

```
word: verstandst
lemma: verstehen
form: indicative imperfect active singular second-person
```

# Lemmatizer/Stemmer

## Implementation

- build a generic framework for inflected languages
  - makes it possible to implement inflection for certain language
  - possibility to add irregular inflected forms
- used it to implement inflection for German verbs, nouns and adjectives
  - can be used as a stemmer
- used a lookup table for lemmatization
  - build with a dictionary of words and their inflection type using the stemmer

# POS Tagger

## Building a Semantic Representation

- part of speech (POS)-tagging is necessary
- identifies category of the words in given sentence ("grammatical tagging")

```
Wir: pronoun
haben: verb
noch: adverb
keinen: pronoun
Hunger: noun
```

# Parsing Wiktionary

- online dictionary build by the community
- runs on MediaWiki platform, the Wikipedia-software
- I parsed the german part of the german version
- '...wiktionary is a formatting-disaster, and was not build to be computer-readable' - quote from 'spencercooly' on stackoverflow
  - however, I was not the first who tried it
  - IWNLP, parser in C#

# Parsing Wiktionary

## Formatting-Disaster

```
| {{{2. Singular Indikativ Präsens Aktiv|{{#if:
{{{Präsens|{{{Präsens Aktiv|{{{Hauptsatzkonjugation|
{{{unpersönlich|}}}}}}}}}}}}| -| {{#switch: {{{3}}} |
e=du {{{1}}}{{{2}}}{{{3}}}{{{4}}}st{{#if: {{{Teil 1|}}}|
&#32;{{{Teil 1}}}{{#if: {{{Teil 2|}}}|&#32;{{{Teil 2}}}|
}}|}} | m={{#switch: {{{2}}} | b | c | ch | d
| f | g | j | k | p | s | t | v | w | x | z |
ß=du {{{1}}}{{{2}}}{{{3}}}est{{#if: {{{Teil 1|}}}|
&#32;{{{Teil 1}}}{{#if: {{{Teil 2|}}}|&#32;{{{Teil 2}}}|}}
|}}| #default=du {{{1}}}{{{2}}}{{{3}}}st{{#if: {{{Teil 1|
}}}|&#32;{{{Teil 1}}}{{#if: {{{Teil 2|}}}|&#32;{{{Teil 2
}}}|}}|}}}} | n={{#switch: {{{2}}} | b | c | ch | d | f
| g | j | k | m | p | s | t | v | w | x | z | ß=du {{{1}}}
{{{2}}}{{{3}}}est{{#if: {{{Teil 1|}}}|&#32;{{{Teil 1}}}
{{#if: {{{Teil 2|}}}|&#32;{{{Teil 2}}}|}}|}} | #default=du
{{{1}}}{{{2}}}{{{3}}}st ...
```

# Parsing Wiktionary

## Formatting-Disaster

- just extracted the basic information for inflection
- used the implemented stemmer for inflection
- still a mess, but manageable

```
{{Deutsch Verb regelmäßig|m|a|ch|e|n|gemacht|zp=zp3
|vp=vp3}}

{{Deutsch Verb unregelmäßig|2=woll|3=wollte|4=wollt|
5=gewollt|6=will|8=n|10=wollen|vp=ja|zp=ja|gerund=ja}}

{{Deklinationsseite Adjektiv|Positiv-Stamm=gut
|Komparativ-Stamm=besser|Superlativ-Stamm=best}}
```

# Parsing Wiktionary

- some words are still not parsed correctly
  - especially those where the MediaWiki-syntax is violated due to mistakes of contributors
- vast majority of words are parsed correctly
  - $\sim$ 70000 nouns
  - $\sim$ 11000 adjectives
  - $\sim$ 9000 verbs
  - $\sim$ 1500 articles, adverbs, conjunctions, subjunctions, prepositions and pronouns

# Tentative Roadmap

- write a lemmatizer/stemmer for german
  - the NLTK 'Snowball Stemmer' is horrible
  - spaCy is better, but still insufficient
- train a POS tagger/syntax parser using the lemmatization tool on the TIGER Corpus
- build semantic representation graphs from the syntax trees
  - will probably work rule based
- build sentences from semantic representation graphs
  - will probably work rule based as well
  - I will focus on 'simple sentences' first and try to get as far as possible

# What I Really did

- write a lemmatizer/stemmer for german
  - the NLTK 'Snowball Stemmer' is horrible
  - spaCy is better, but still insufficient
- ~~train~~ build a POS tagger~~/syntax parser using the lemmatization tool on the TIGER Corpus~~ that uses the parsed wiktionary
- build semantic representation graphs from the syntax trees
  - will probably work rule based
- build sentences from semantic representation graphs
  - will probably work rule based as well
  - I will focus on 'simple sentences' first and try to get as far as possible

# Summary: Results

- framework for inflected languages
  - can be used to implement other languages
- inflection of German nouns, verbs and adjectives
- parsed the German wiktionary
- framework allows easy extension of the word database
- solid foundation for development of semantic representation

# Outlook

## Next Steps: Further Work on Database

- the current program parses the data from the wiktionary dump
  - time-consuming
- save words, POS tags and necessary inflection parameters in XML file or a SQL database
- a lot of additional useful parameters can be parsed from wiktionary by extending the parser
- the database can be extended
  - furthermore, a statistical/ML method should be added to handle words not given in the database
  - on the other hand it should be sufficient to build representations for most basic sentences
  - therefore not urgent

# Outlook

## Next Steps: From Single Words to Groups of Words

- till now the focus was on single words
- for single words the forms are often still ambiguous
  - for nouns the form can be derived by their corresponding article
  - for verbs one often has a combination of a participle past/infinitive and a auxiliary
- identify the related words to determine the form
- in much cases this should be relatively easy with the database and framework

# Outlook

Next Steps: Identify Dependent and Independent Clauses

- identify all dependent and independent clauses
- recap: the semantic representation should indicate which parts are expressed in independent clauses/dependent clauses
  - one can first build the semantic representation for the particular clauses
  - connect them afterwards
- German dependent clauses are separated by conjunctions and/or commas
- German independent clauses are separated by subjunctions and commas

# Questions?

# References


- Schubert, Lenhart: Semantic Representation. In: Proceedings of the Twenty-Ninth (AAAI) Conference on Artificial Intelligence, p. 4132-4139 (2015)
- Jurafsky and Martin: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. Upper Saddle River, New Jersey: Prentice Hall, 2000. Print (Chapter on Machine Translation: p.799-831)
- Liebeck and Conrad: IWNLP: Inverse Wiktionary for Natural Language Processing, Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Association for Computational Linguistics, 2015, p. 414-418