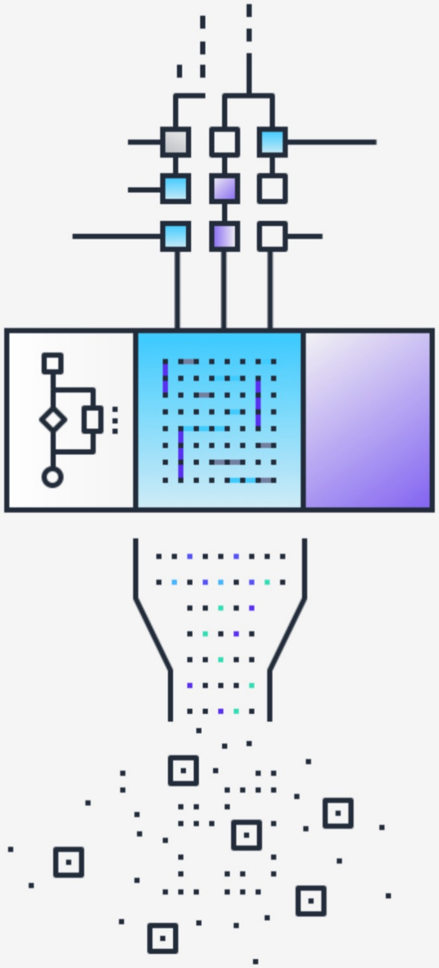
Interpretable Natural Language Segmentation and Generation Using Link Grammar

VIGNAV RAMESH
R&D Intern @ SingularityNET,
Contributor @ Aigents

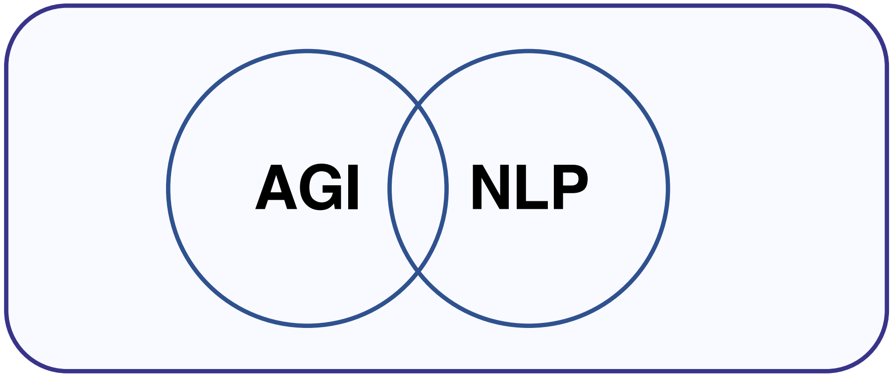


Part I

INTRO: AGI, ILP, ULL, QA, & LINK GRAMMAR

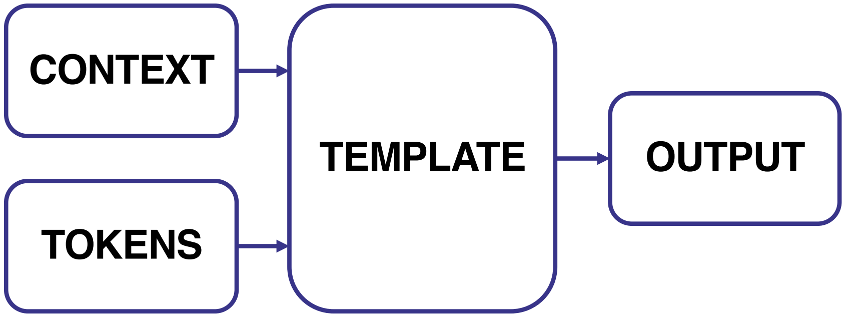
The Big Picture

GENERAL CONVERSATIONAL INTELLIGENCE



AGI

NLP



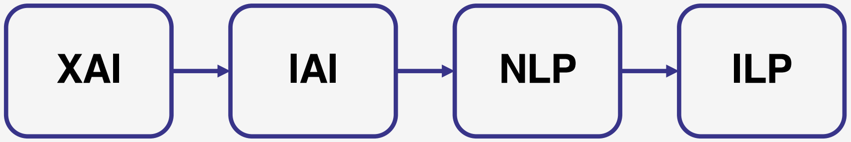
CONTEXT

TOKENS

TEMPLATE

OUTPUT

INTERPRETABLE LANGUAGE PROCESSING

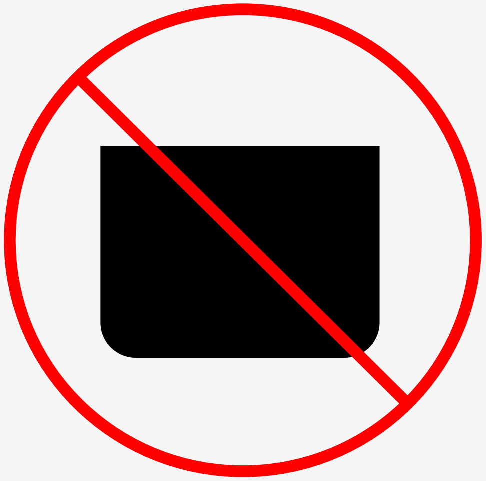


XAI

IAI

NLP

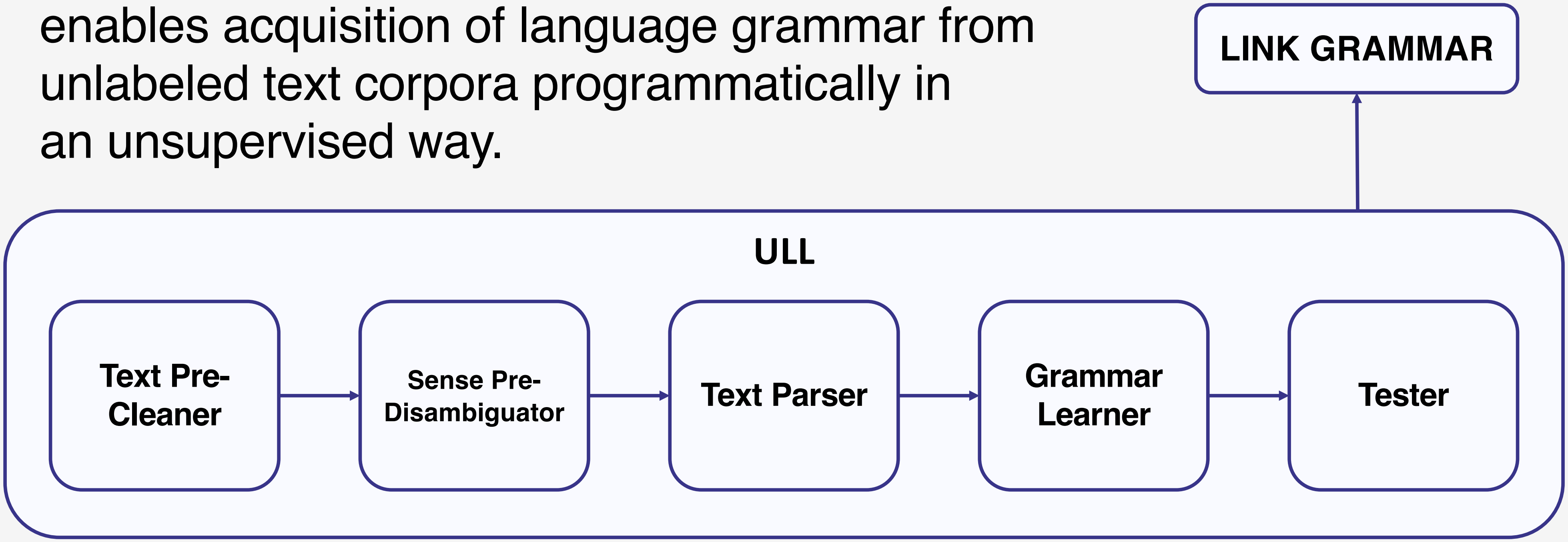
ILP



Unsupervised Language Learning

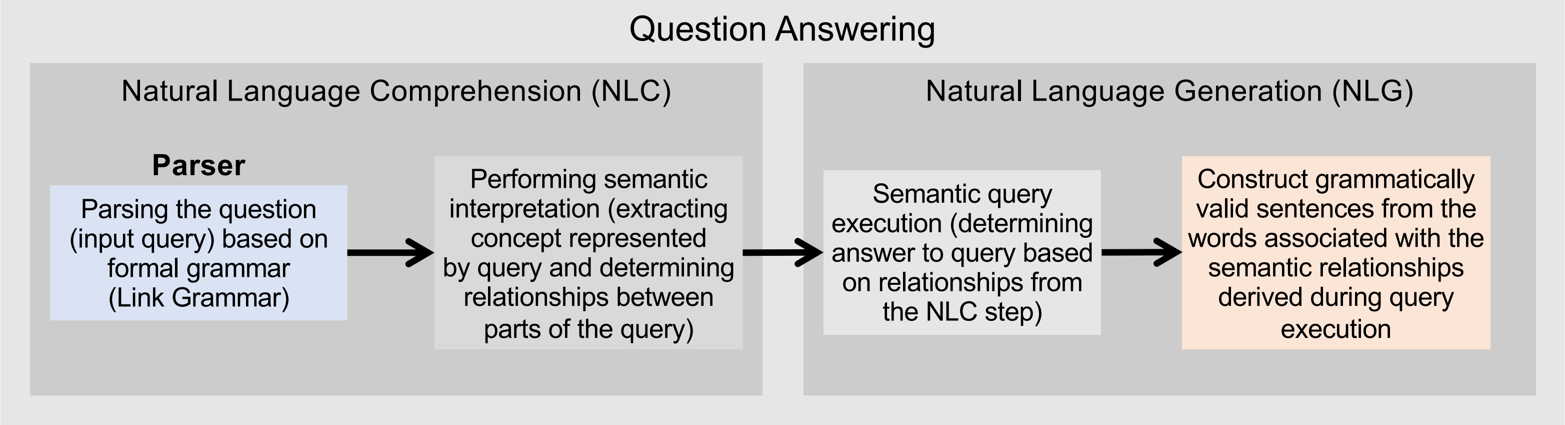
ULL…

enables acquisition of language grammar from unlabeled text corpora programmatically in an unsupervised way.



Question Answering

Pipeline



Question Answering

Natural Language Comprehension (NLC)

Parser

Parsing the question (input query) based on formal grammar (Link Grammar)

Performing semantic interpretation (extracting concept represented by query and determining relationships between parts of the query)

Natural Language Generation (NLG)

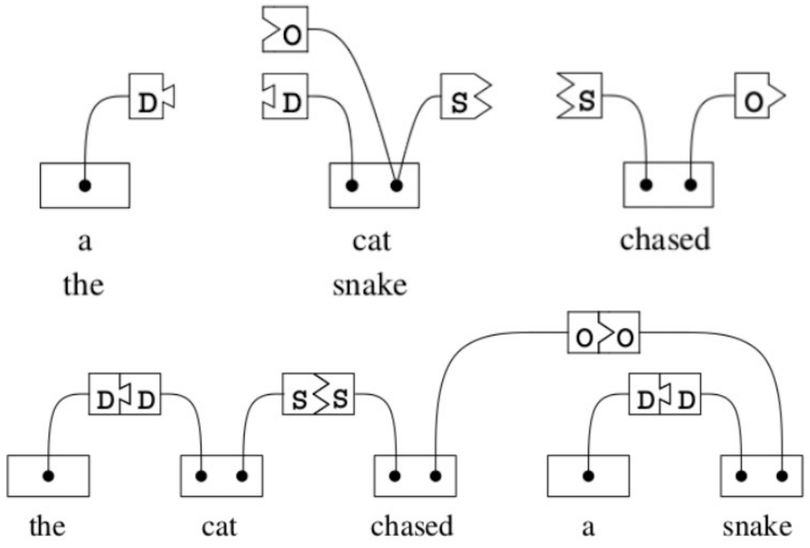
Semantic query execution (determining answer to query based on relationships from the NLC step)

Construct grammatically valid sentences from the words associated with the semantic relationships derived during query execution

Link Grammar

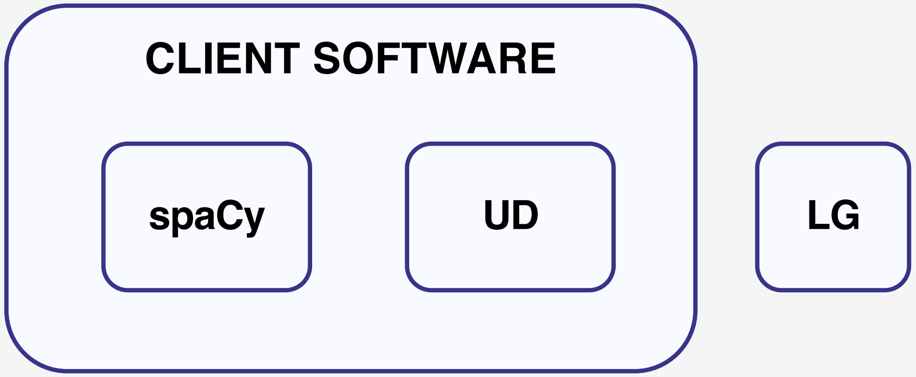
WHAT IS LINK GRAMMAR?[1]

a the: D+;
cat snake: D– & (S+ or O–);
chased: S– & O+;



[1] Sleator, D., Temperley, D.: Parsing English with a Link Grammar. In: Proceedings of the Third International Workshop on Parsing Technologies, pp. 277–292. Association for Computational Linguistics, Netherlands (1993)

WHY LINK GRAMMAR?



- ○ spaCy[2], UD[3] – head-dependent, integrated into end-user programs
- ✓ LG can be updated **without having to modify client code**
- ✓ **First native Java LG support**
- ✓ **Human-readable, editable – adds to interpretability NLG/NLS, contributes to GCI**

[2] https://spacy.io/, [3] https://universaldependencies.org/

Part II

DEEP DIVE INTO NLS

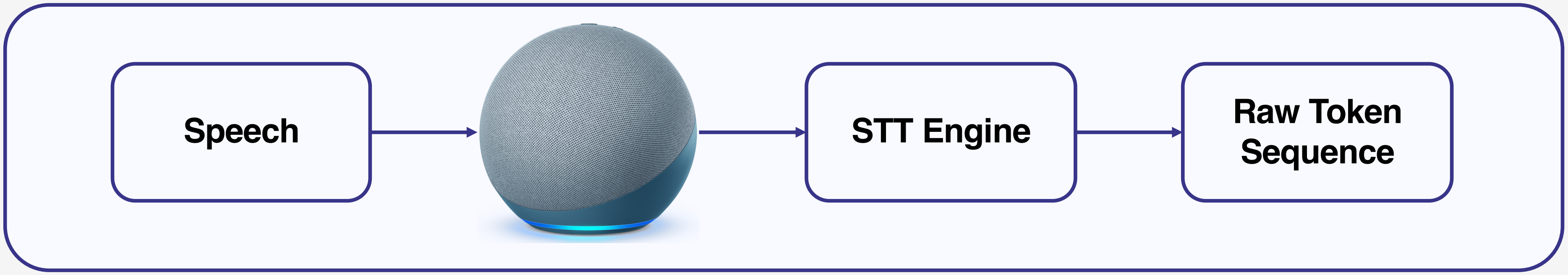
What is NLS?

- Dividing text into meaningful units
- Sub-problem: sentence segmentation

green vines attached to the trunk of the tree had wound themselves toward the top of the canopy ants used the vine as their private highway, avoiding all the creases and crags of the bark, to freely move at top speed from top to bottom or bottom to top depending on their current chore at least this was the way it was supposed to be something had damaged the vine overnight halfway up the tree leaving a gap in the once pristine ant highway

Sources of Unsegmented Text

- Speech-to-text (STT) recognition engines

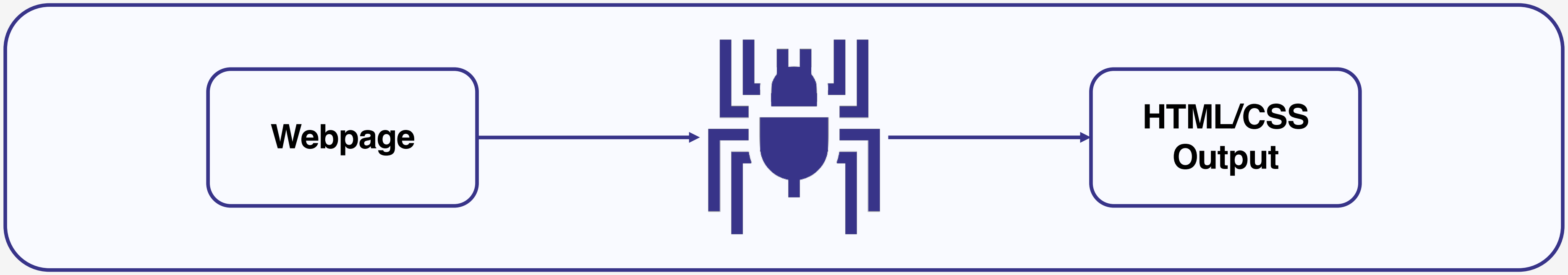


Speech

STT Engine

Raw Token Sequence

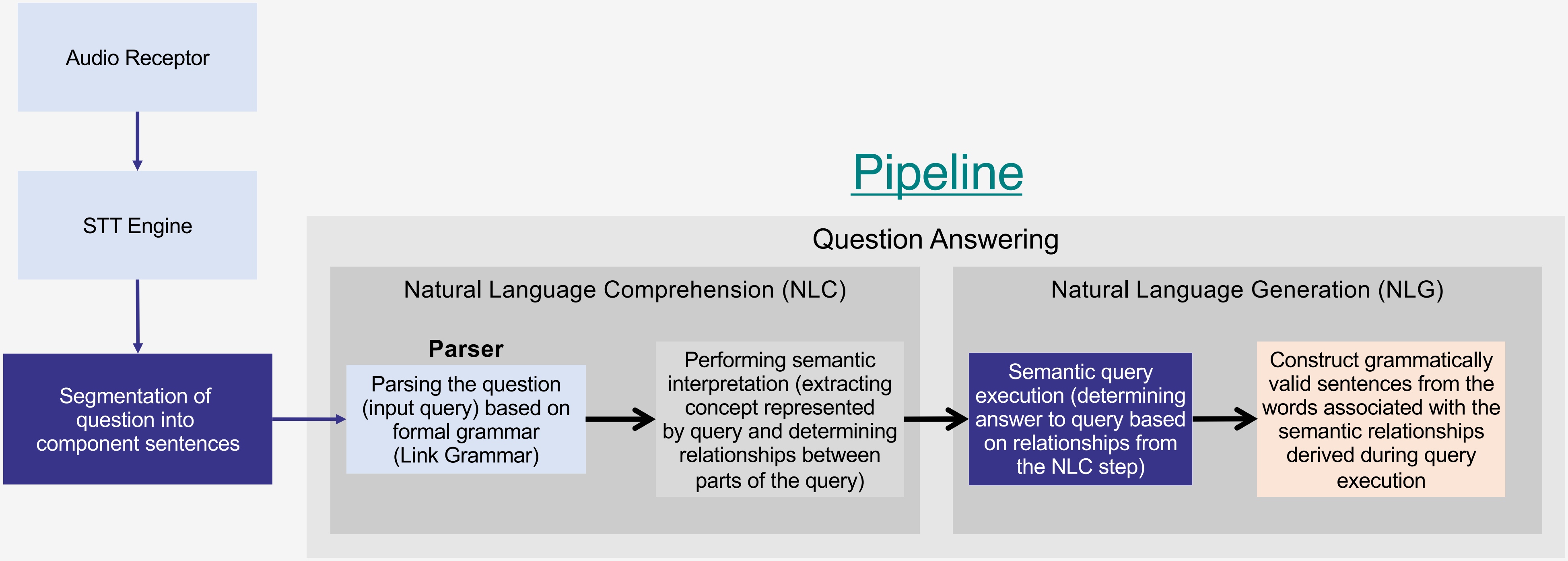
- Crawled web pages



Webpage

HTML/CSS Output

NLS in QA



Audio Receptor

STT Engine

Segmentation of question into component sentences

Pipeline

Question Answering

Natural Language Comprehension (NLC)

Parser

Parsing the question (input query) based on formal grammar (Link Grammar)

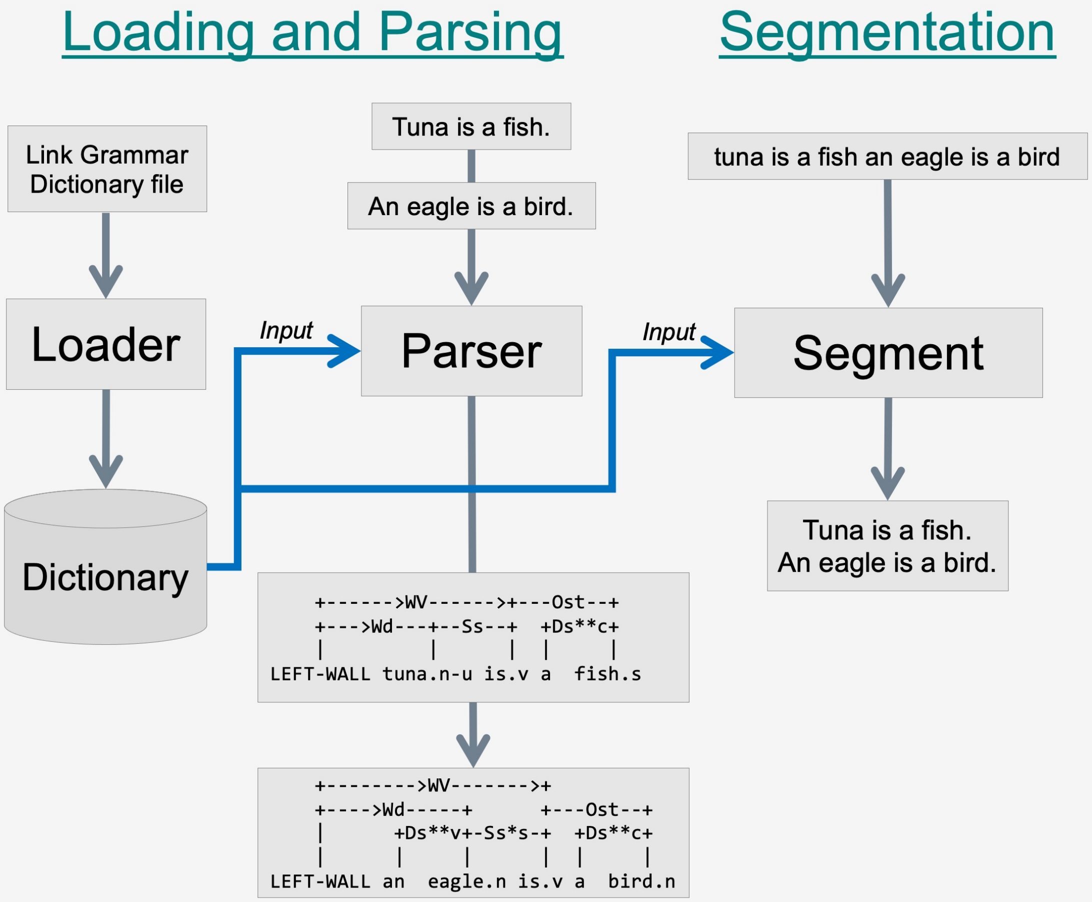
Performing semantic interpretation (extracting concept represented by query and determining relationships between parts of the query)

Natural Language Generation (NLG)

Semantic query execution (determining answer to query based on relationships from the NLC step)

Construct grammatically valid sentences from the words associated with the semantic relationships derived during query execution

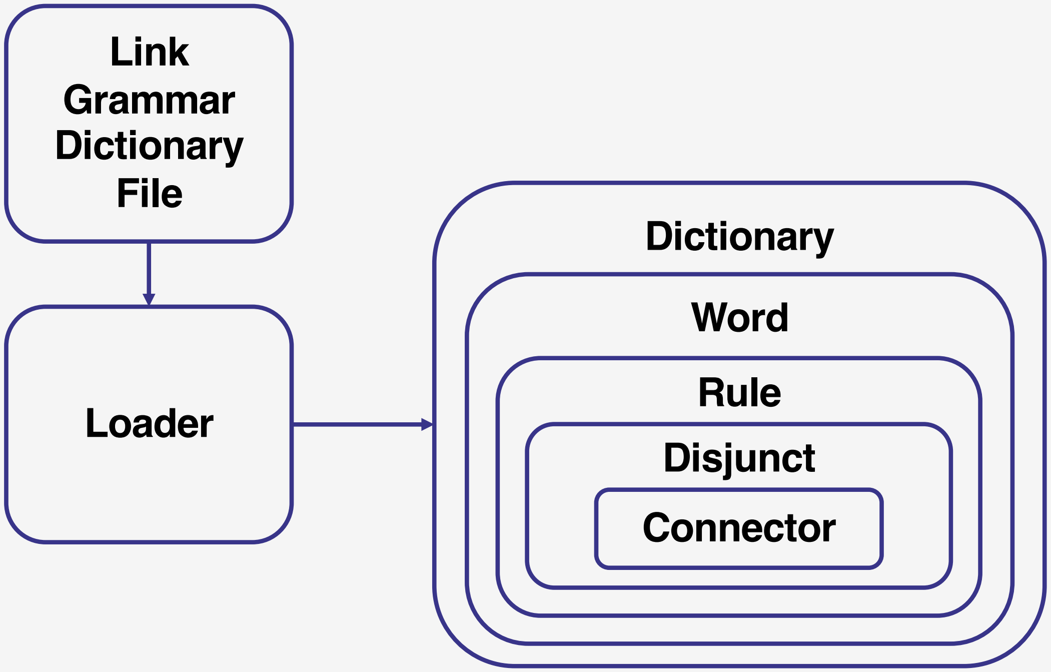
Loader-Segment NLS Architecture



```
    +------>WV------>+---Ost--+
    +--->Wd---+--Ss--+  +Ds**c+
    |         |      |  |     |
LEFT-WALL tuna.n-u is.v a  fish.s
```

```
    +-------->WV------->+
    +---->Wd-----+      +---Ost--+
    |      +Ds**v+-Ss*s-+  +Ds**c+
    |      |     |      |  |     |
LEFT-WALL an  eagle.n is.v a  bird.n
```

Loader



Link Grammar Dictionary File

Loader

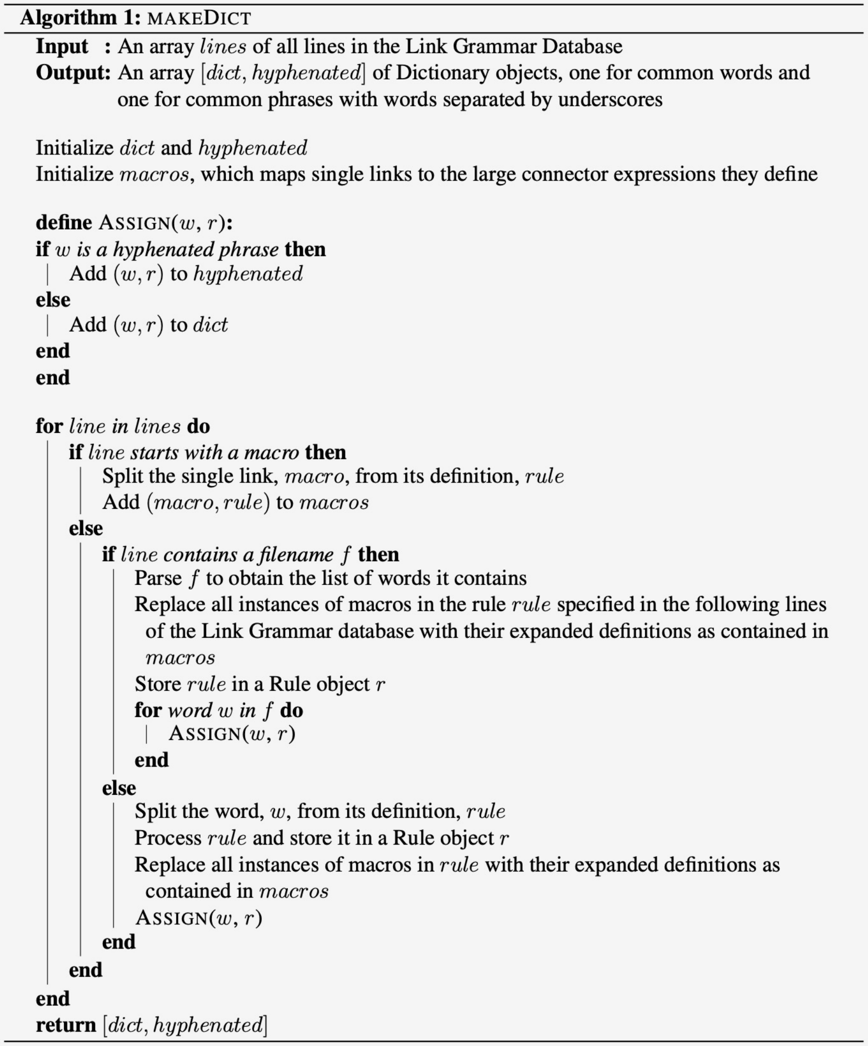
Dictionary

Word

Rule

Disjunct

Connector



```
Algorithm 1: MAKEDICT
Input  : An array lines of all lines in the Link Grammar Database
Output: An array [dict, hyphenated] of Dictionary objects, one for common words and
        one for common phrases with words separated by underscores

Initialize dict and hyphenated
Initialize macros, which maps single links to the large connector expressions they define

define ASSIGN($w$, $r$):
if $w$ is a hyphenated phrase then
    Add $(w, r)$ to hyphenated
else
    Add $(w, r)$ to dict
end
end

for line in lines do
    if line starts with a macro then
        Split the single link, macro, from its definition, rule
        Add (macro, rule) to macros
    else
        if line contains a filename $f$ then
            Parse $f$ to obtain the list of words it contains
            Replace all instances of macros in the rule rule specified in the following lines
              of the Link Grammar database with their expanded definitions as contained in
              macros
            Store rule in a Rule object $r$
            for word $w$ in $f$ do
                ASSIGN($w$, $r$)
            end
        else
            Split the word, $w$, from its definition, rule
            Process rule and store it in a Rule object $r$
            Replace all instances of macros in rule with their expanded definitions as
              contained in macros
            ASSIGN($w$, $r$)
        end
    end
end
return [dict, hyphenated]
```

SEGMENT

Algorithm 2: SEGMENT

```
Input  : An array $tokens$ of words and commas extracted from the input text by
         PROCESSSENTENCES
Output: A list of sentences obtained by segmenting $tokens$ into grammatically and
         morphologically valid arrays of tokens

Start a counter $idx$, representing the index of the current token in $tokens$
Initialize an empty list $ret$, which will eventually contain the sentences that SEGMENT
  will return

while $idx < length(tokens)$ do
    for $i$ in $[idx, length(tokens)]$ do
        Create array $arr$ containing the subset of $tokens$ from indices $idx$ to $i$
        if ISVALID($arr$) and CHECK($tokens[i+1], tokens[i+2]$) then
            $threshold = n$ (default value of 2)
            Add $tokens[i+1], tokens[i+2]$ ... $tokens[i+n]$ to $arr$
            if ISVALID($arr$) and CHECK($tokens[i+n+1], tokens[i+n+2]$) then
                Construct a sentence from $arr$, i.e. create a string with the tokens in $arr$
                  separated by spaces and add appropriate punctuation
                Add the sentence to $ret$
                $idx \leftarrow i + n + 1$
            else
                Construct a sentence from the original value of $arr$
                Add the sentence to $ret$
                $idx \leftarrow i + 1$
            end
        end
    end
end
return $ret$
```

Algorithm 3: CONNECTS

```
Input  : A pair of strings $left$ and $right$, representing the two words to potentially be
         connected
Output: A boolean value indicating whether $left$ and $right$ can be connected via valid
         Link Grammar rules

Obtain $leftList$, the list of rules corresponding with $left$ (i.e. the rule when $left$ is a verb,
  the rule when $left$ is a gerund, etc.), from the global Dictionary variables $dict$ and
  $hyphenated$
Obtain $rightList$ in a similar manner

for $leftRule$ in $leftList$ do
    for $rightRule$ in $rightList$ do
        Split $leftRule$ and $rightRule$ into lists of Disjuncts $ld$ and $rd$
        for $l$ in $ld$ do
            for $r$ in $rd$ do
                Replace all instances of '$-$' in $l$ with '$+$' and vice versa
                if $l = r$ then
                    return true
                else
                    continue
                end
            end
        end
    end
end
return false
```

Results

“Small World” Corpus NLS Results

Metric	Result
Ground Truth (POC-English Corpus) [4]	
Total number of sentences	88
Average sentence length	5.51136
NLS Algorithm Results	
Total number of sentences	87
Average sentence length	5.57471
Overall Statistics	
Runtime	57 sec.
Number of sentences matching exactly	78
Number of sentence boundaries accurately identified	85/87
Accuracy of boundary identification	0.97701

Gutenberg Corpus NLS Results

Metric	Result
Ground Truth (Gutenberg Corpus) [5]	
Total number of sentences	10
Average sentence length	13.2
NLS Algorithm Results	
Total number of sentences	11
Average sentence length	13.2
Overall Statistics	
Runtime	14 sec.
Number of sentences matching exactly	7
Number of sentence boundaries accurately identified	7/9
Accuracy of boundary identification	0.77778

[4] http://langlearn.singularitynet.io/data/poc-english/, [5] http://langlearn.singularitynet.io/data/cleaned/English/Gutenberg

Comparison with Prior Work

- **Three alternatives**: Syntok[6], PragmaticNet[7], DeepSegment[8]

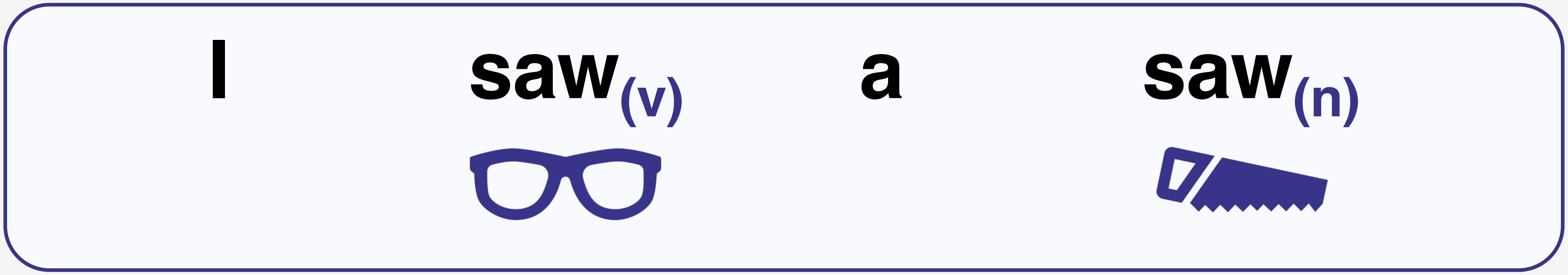
- **Syntok and PragmaticNet**
  - "Terminal markers"
  - Punctuation, quotations, and parentheticals
  - Zero boundaries identified

- **DeepSegment**
  - BiLSTMs
  - CRF-based supervision
  - 1-2 boundaries identified

[6] https://github.com/fnl/syntok
[7] https://www.tm-town.com/natural-language-processing
[8] https://github.com/notAI-tech/deepsegment

The Grammatical Ambiguity Problem

- **Problem**

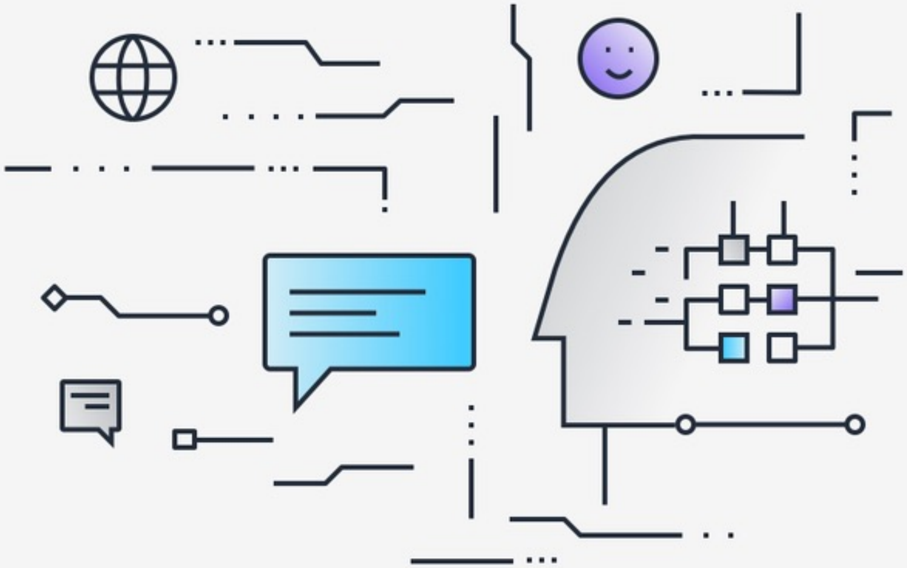


I saw$_{(v)}$ a saw$_{(n)}$

- **Solution: Semantic (word sense) disambiguation**
    - What “sense” or definition of a word is activated by that word’s use in a particular context?

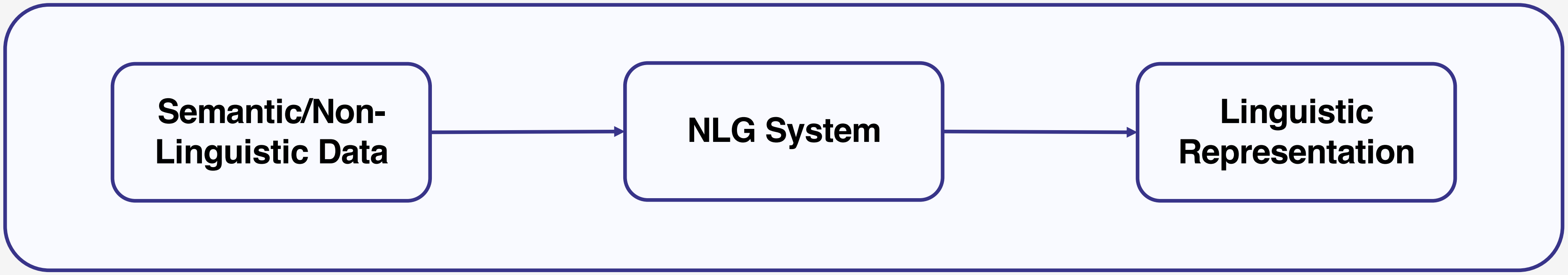
Part III

DEEP DIVE INTO NLG



What is NLG?

- Holistic NLG

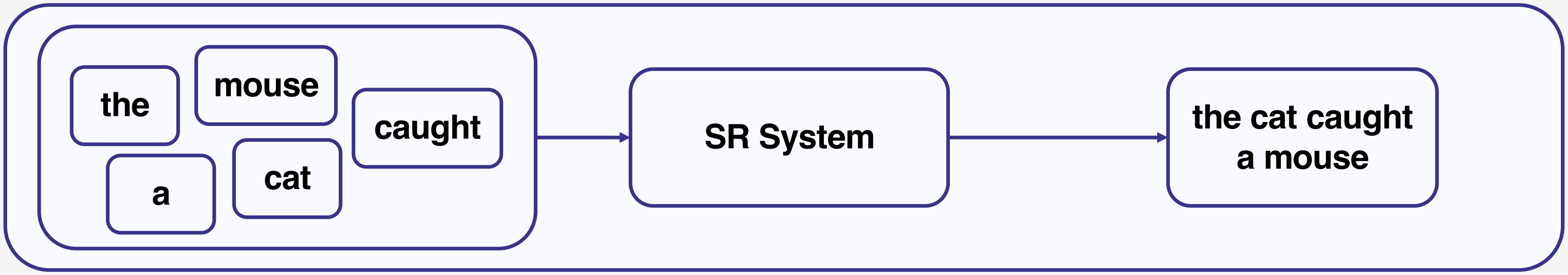


Semantic/Non-Linguistic Data

NLG System

Linguistic Representation

- Surface Realization



the

mouse

caught

a

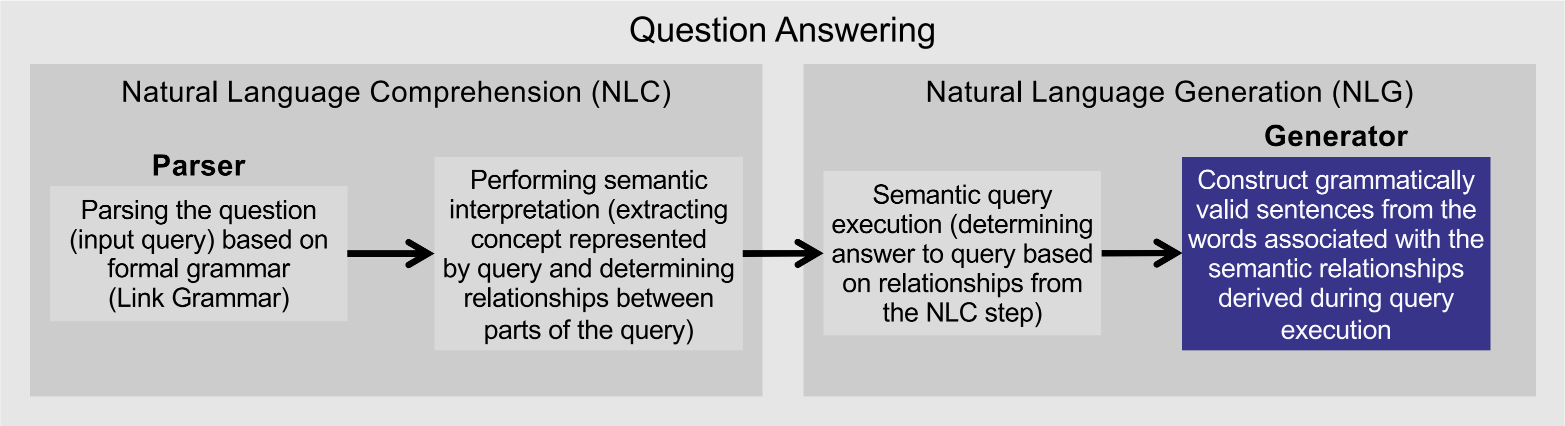
cat

SR System

the cat caught a mouse

NLG in QA

Pipeline



Question Answering

Natural Language Comprehension (NLC)

Parser

Parsing the question (input query) based on formal grammar (Link Grammar)

Performing semantic interpretation (extracting concept represented by query and determining relationships between parts of the query)

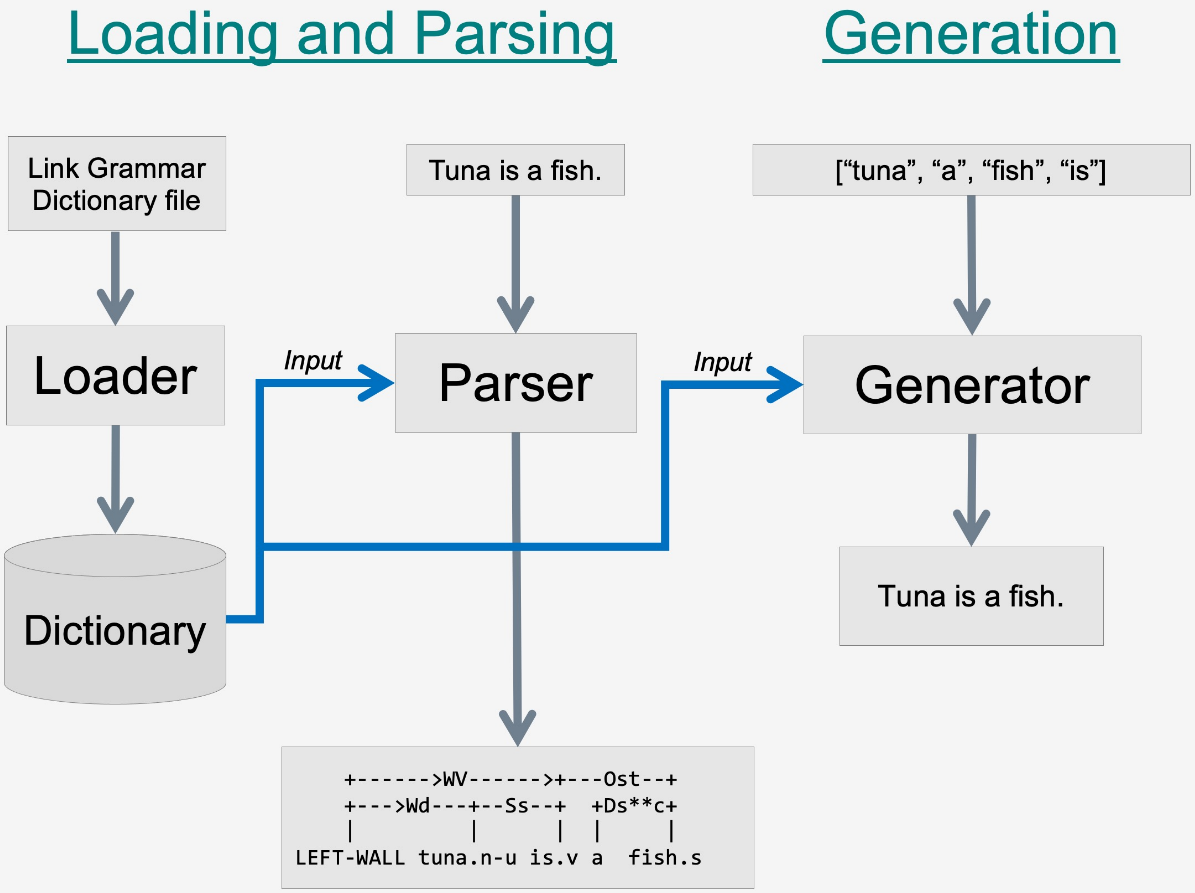
Natural Language Generation (NLG)

Semantic query execution (determining answer to query based on relationships from the NLC step)

Generator

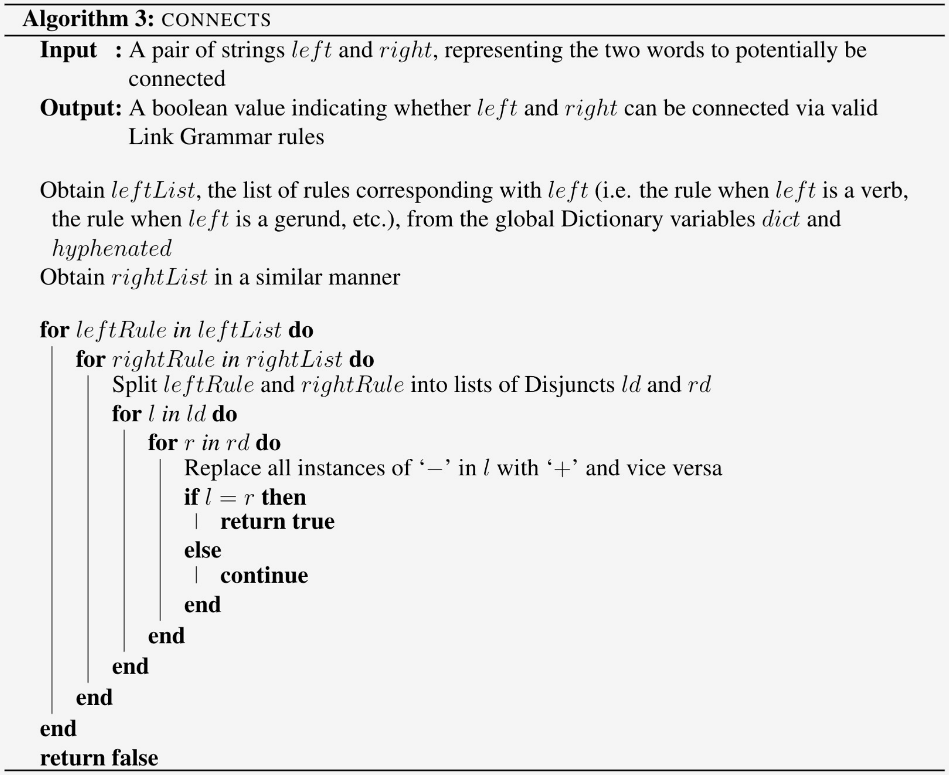
Construct grammatically valid sentences from the words associated with the semantic relationships derived during query execution

Loader-Generator NLG Architecture



```
    +------>WV------>+---Ost--+
    +--->Wd---+--Ss--+  +Ds**c+
    |         |      |  |     |
LEFT-WALL tuna.n-u is.v a  fish.s
```

Generator

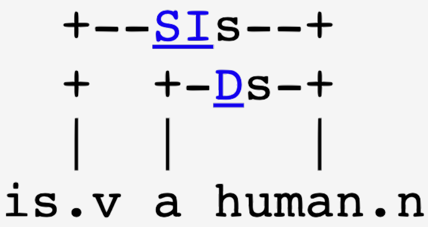


Algorithm 3: CONNECTS

Input : A pair of strings $left$ and $right$, representing the two words to potentially be connected

Output: A boolean value indicating whether $left$ and $right$ can be connected via valid Link Grammar rules

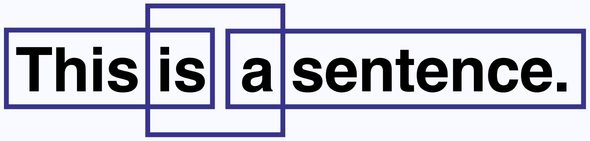
```
Obtain $leftList$, the list of rules corresponding with $left$ (i.e. the rule when $left$ is a verb,
  the rule when $left$ is a gerund, etc.), from the global Dictionary variables $dict$ and
  $hyphenated$
Obtain $rightList$ in a similar manner

for $leftRule$ in $leftList$ do
    for $rightRule$ in $rightList$ do
        Split $leftRule$ and $rightRule$ into lists of Disjuncts $ld$ and $rd$
        for $l$ in $ld$ do
            for $r$ in $rd$ do
                Replace all instances of '$-$' in $l$ with '$+$' and vice versa
                if $l = r$ then
                    return true
                else
                    continue
                end
            end
        end
    end
end
return false
```



```
  +--SIs--+
  +  +-Ds-+
  |  |    |
is.v a human.n
```

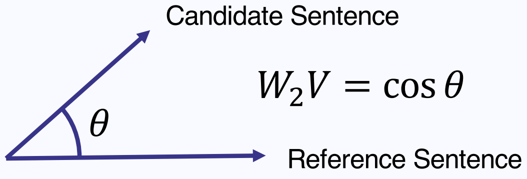
Metrics

BLEU (Bigram)



This is a sentence.

Word2Vec Cosine Similarity



$$W_2V = \cos\theta$$

WER

$$\frac{S + D + I}{N}$$

TER

$$\frac{E}{R}$$

Results

“Small World” Corpus, ULL Grammar

Metric	Result
Architecture-Specific Metrics	
Single correct generated sentence	62/92
Multiple sentences with one correct	30/92
Multiple sentences with none correct	0/92
No generated sentences	0/92
Too many results (>25 candidates)	0/92
Accuracy	1.000
Overall Statistics	
Average BLEU (Bigram)	1.000
Average Word2Vec Cosine Similarity	0.988
Average WER	0.246
Average TER	0.082

“Small World” Corpus, ULL Grammar

Metric	Result
Architecture-Specific Metrics	
Single correct generated sentence	8/92
Multiple sentences with one correct	57/92
Multiple sentences with none correct	0/92
No generated sentences	0/92
Too many results (>25 candidates)	27/92
Accuracy	0.707
Overall Statistics	
Average BLEU (Bigram)	0.999
Average Word2Vec Cosine Similarity	0.900
Average WER	3.713
Average TER	0.395

Results (cont.) + Comparison with Prior Work

Gutenberg Corpus, Link Grammar

Metric	Result
Architecture-Specific Metrics	
Single correct generated sentence	1/54
Multiple sentences with one correct	53/54
Multiple sentences with none correct	0/54
No generated sentences	0/54
Too many results (>25 candidates)	0/54
Accuracy	1.000
Overall Statistics	
Average BLEU (Bigram)	0.652
Average Word2Vec Cosine Similarity	0.746
Average WER	5.976
Average TER	1.738

Baseline Results

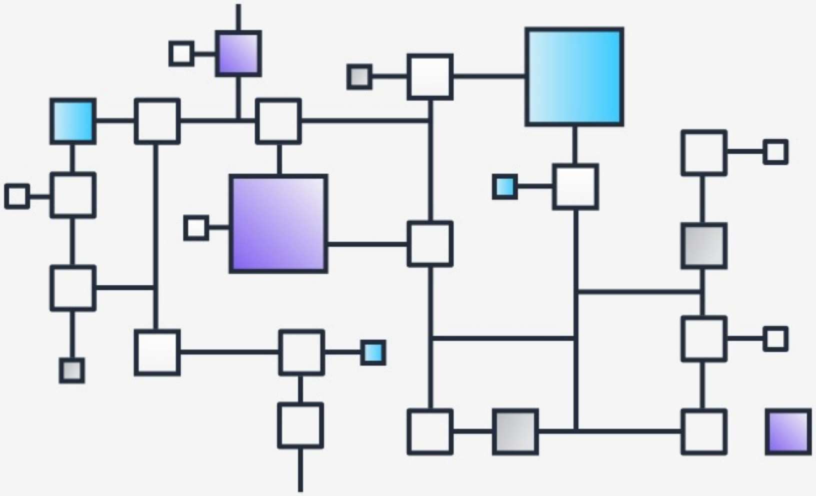
Metric	Result
Architecture-Specific Metrics	
Average BLEU (Bigram)	0.747
Average Word2Vec Cosine Similarity	0.722
Average WER	3.114
Average TER	0.505
Overall Statistics	
Average BLEU (Bigram)	0.325
Average Word2Vec Cosine Similarity	0.401
Average WER	11.62
Average TER	1.988

Baseline Model: State-of-the-art Transformer[9] architecture tasked with sentence reconstruction and subject to BERT-type modifications

[9] A. Vaswani, et al., "Attention Is All You Need," arXiv:1706.03762 [cs.CL], December 2017

Part IV

CONCLUSION



Applications

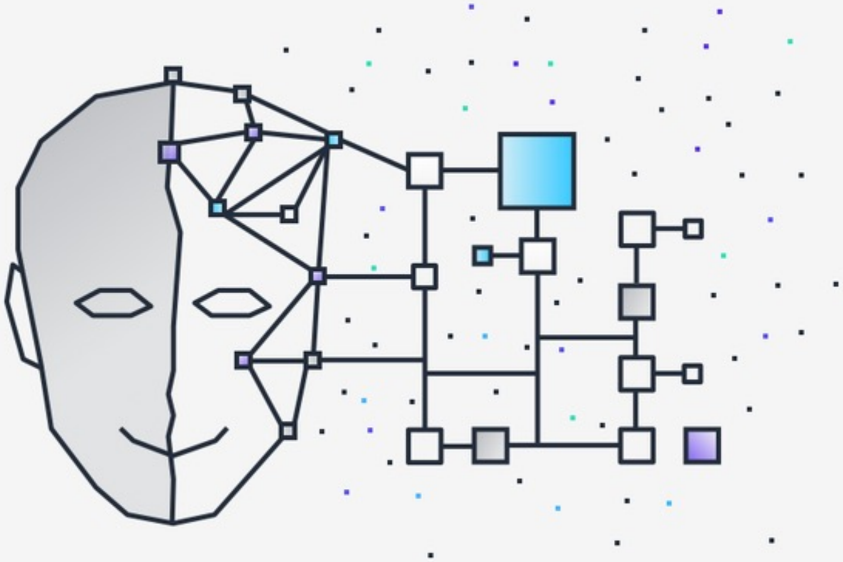
- **NLS**
  - Semantic query execution in QA
    - Aigents Social Media Intelligence Platform
  - Text simplification
  - Any NLP algorithms that operate at the sentential level
    - Automatic summarization, entity extraction, etc.
- **NLG**
  - Sentence generation in QA
    - Replacing Aigents “pidgin” English
  - Virtual assistant AI technologies

Current & Future Work

- Implementing grammatical and semantic disambiguation
- Adding support for languages besides English
  - Russian – requires heavy morphology usage

Part IV

CODE USAGE & DEMOS



Loader

1. To execute a series of unit tests, run:

```
cd src
javac main/java/org/aigents/nlp/lg/*.java
javac main/java/org/aigents/nlp/gen/*.java
javac test/java/org/aigents/nlp/gen/*.java
java test.java.org.aigents.nlp.gen.TestSegment
```

2. To output the rule and disjuncts associated with a given word, run:

```
cd src
javac main/java/org/aigents/nlp/lg/*.java
java main.java.org.aigents.nlp.lg.Loader en/4.0.dict board
```

Natural Language Segmentation

1. To test on the POC-English corpus, run:

```
cd src
javac main/java/org/aigents/nlp/lg/*.java
javac main/java/org/aigents/nlp/gen/*.java
java main.java.org.aigents.nlp.gen.Segment en/4.0.dict poc_english.txt
```

2. To test on the Gutenberg corpus, run:

```
cd src
javac main/java/org/aigents/nlp/lg/*.java
javac main/java/org/aigents/nlp/gen/*.java
java main.java.org.aigents.nlp.gen.Segment en/4.0.dict gutenberg544.txt
```

3. To test on custom text, run:

```
cd src
javac main/java/org/aigents/nlp/lg/*.java
javac main/java/org/aigents/nlp/gen/*.java
java main.java.org.aigents.nlp.gen.Segment en/4.0.dict tuna is a fish
    eagle is a bird dog is a mammal
```

Natural Language Generation

1. To test on the POC-English corpus, run:

```
cd src
javac main/java/org/aigents/nlp/lg/*.java
javac main/java/org/aigents/nlp/gen/*.java
java main.java.org.aigents.nlp.gen.Generator en/4.0.dict poc_english.txt
```

2. To test on the Gutenberg corpus, run:

```
cd src javac main/java/org/aigents/nlp/lg/*.java
javac main/java/org/aigents/nlp/gen/*.java
java main.java.org.aigents.nlp.gen.Generator en/4.0.dict gutenberg544.txt
```

3. To test on custom text, run:

```
cd src
javac main/java/org/aigents/nlp/lg/*.java
javac main/java/org/aigents/nlp/gen/*.java
java main.java.org.aigents.nlp.gen.Generator en/4.0.dict food Cake a is now
```

Walking Through GitHub

https://github.com/aigents/aigents-java-nlp

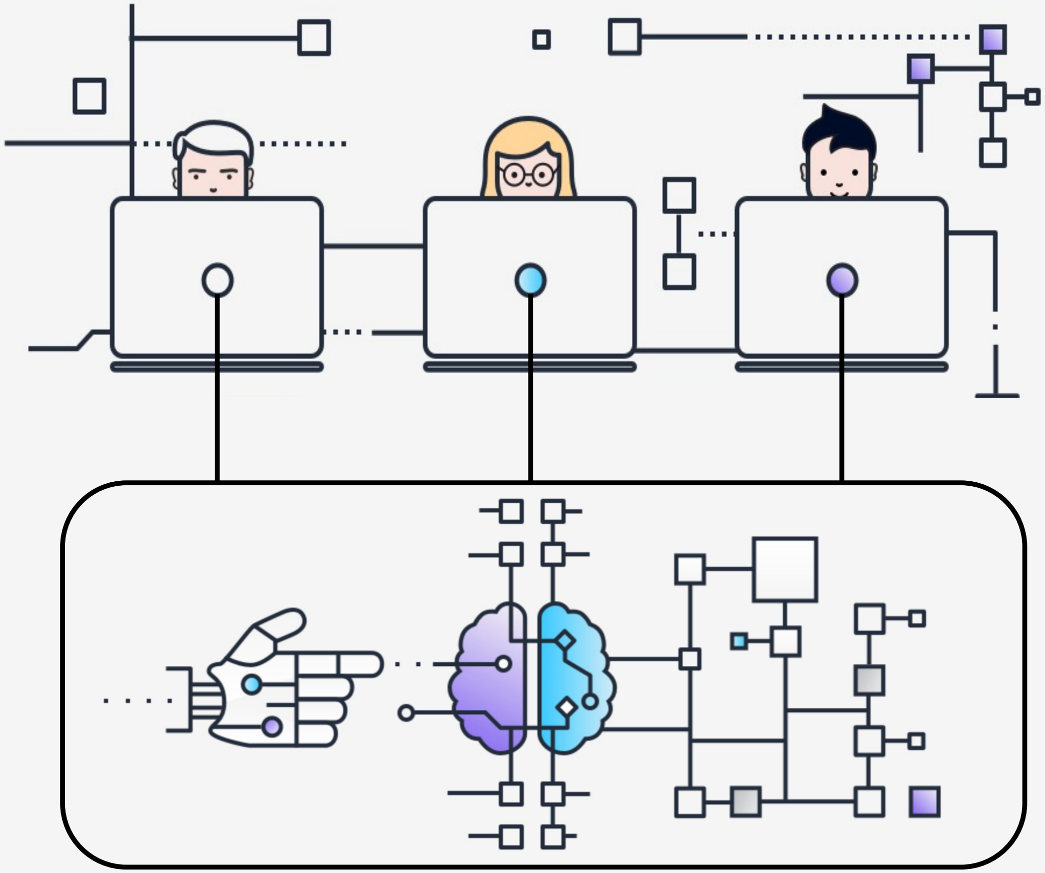
Questions?

Code

NLG Paper

NLS Paper

Contact



ART: https://blog.singularitynet.io/research/home