AI in NLP and Code Generation

Tianyi Zhang
Nov 28, 2022

CS 57700: Natural Language Processing
CS 59300: Human-AI Interaction
Tianyi Zhang
Assistant Professor of Computer Science
Interactive Intelligent Systems Lab
LWSN 3154H • tianyi@purdue.edu • https://tianyi-zhang.github.io/

Research Interests: HCI, SE, AI

In IIS, we develop interactive intelligent systems to:

• **improve the productivity of programmers and domain experts**
  o Interactive program synthesis and program repair
  o Knowledge acquisition from Stack Overflow
  o Interactive web scraping and data labeling for data scientists

• **improve the robustness, transparency, and fairness of ML**
  o Declarative testing and traffic scene generation for autonomous driving
  o Interactive model debugging for RNNs and transformers
  o Interactive visual analytics for detecting biases and Out-of-Distribution (OOD) errors
Natural Language Processing

Human-Human Communication

Human-Machine Communication
NLP is a branch of artificial intelligence that concerns with the interactions between computers and human languages, in particular how to enable computers analyze, understand, and generate human languages in both written and spoken contexts.
Welcome to

Eliza is a mock Rogerian psychotherapist.
The original program was described by Joseph Weizenbaum in 1966.
This implementation by Norbert Landsteiner 2005.

ELIZA: Is something troubling you?
YOU: Men are all alike.
ELIZA: What is the connection, do you suppose?
YOU: They're always bugging us about something or other.
ELIZA: Can you think of a specific example?
YOU: Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here?
YOU: He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU: It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy?
YOU:
Lunar (1971)

Chemical analysis of rock samples from the moon

Natural Language Query

- Give me all analyses for Hydrogen in Sample 10046.
- In which samples has apatite been identified?
- What are the plag analyses for breccias?
- What is the average age of the basalts?
- …
Apple Knowledge Navigator (1987)
NLP after 2010

Customer Service
Online

Customer Service

👋 Hello!

What brings you here today?

💰 Support questions

💰 Sales questions

Siri
GitHub Copilot: Your AI Pair Programmer

```python
import datetime

def parse_expenses(expenses_string):
    """Parse the list of expenses and return the list of triples (date, value, currency).
    Ignore lines starting with #.
    Parse the date using datetime.
    Example expenses_string:
        2016-01-02 -34.01 USD
        2016-01-03 2.59 DKK
        2016-01-03 -2.72 EUR
    """
    expenses = []
    for line in expenses_string.splitlines():
        if line.startswith("#"): continue
        date, value, currency = line.split(" ")
        expenses.append((datetime.datetime.strptime(date, "%Y-%m-%d"),
                         float(value),
                         currency))
    return expenses
```

OpenAI

Microsoft

GitHub
Galactica: A Language Model for Science
NLP History

• Symbolic NLP (1950s – early 1990s)
  • Grammars, rules, ontologies, knowledge base, etc.

• Statistical NLP (1990s – 2010s)
  • Learn from a text corpus
  • Hidden markov models, probabilistic grammars, TF-IDF, LDA, SVM, etc.

• Neural NLP (2010s – present)
  • Deep neural networks and representation learning
Tasks and Applications in NLP

**Understanding**

- Sentiment analysis
- Speech recognition
- Topic modeling
- Text classification
- Natural language inference
- Semantic parsing
- Spam detection
- Named entity recognition
- Relation extraction
- etc.

**Generation**

- Chatbots
- Question answering
- Text summarization
- Image captioning
- Machine translation
- Natural language interfaces
- Code generation
- Text completion
- Creative writing
- etc.
Fundamental Methods in NLP

• Text preprocessing
  • Lowercase, tokenization, stop words removal, stemming, lemmatization

<table>
<thead>
<tr>
<th>Original Word</th>
<th>After Stemming</th>
</tr>
</thead>
<tbody>
<tr>
<td>program</td>
<td>program</td>
</tr>
<tr>
<td>programs</td>
<td>program</td>
</tr>
<tr>
<td>programmed</td>
<td>program</td>
</tr>
<tr>
<td>programming</td>
<td>program</td>
</tr>
</tbody>
</table>

Chopping off suffixes based on rules

<table>
<thead>
<tr>
<th>Original Word</th>
<th>After Lemmatization</th>
</tr>
</thead>
<tbody>
<tr>
<td>is</td>
<td>be</td>
</tr>
<tr>
<td>are</td>
<td>be</td>
</tr>
<tr>
<td>better</td>
<td>good</td>
</tr>
<tr>
<td>programming</td>
<td>program</td>
</tr>
</tbody>
</table>

Reducing each word to its base form
Fundamental Methods in NLP

• Part of Speech (POS) Tagging

I like programming in Python.

Pronoun Verb Verb Preposition Noun

• Applications of POS tagging
  • Named entity recognition, sentiment analysis, question answering, etc.
Fundamental Methods in NLP

- Dependency Parsing

**nssubj**: nominal subject  
**dobj**: direct object  
**det**: determiner  
**nmod**: nominal modifier  
**case**: prepositions, postpositions, and other case markers

I prefer the morning flight through Denver
Fundamental Methods in NLP

• Dependency Parsing

I prefer the morning flight through Denver

prefer
I
flight
the
morning
Denver

through

Dependency Tree

Phrase-Structure Tree
Fundamental Methods in NLP

• Text preprocessing
  • Lowercase, tokenization, stop words removal, stemming, lemmatization
• Part of Speech (POS) tagging
• Dependency parsing
• Vectorization

Commonly used in symbolic NLP
Commonly used in statistical and neural NLP
Vectorization Methods

• Bag of Words
  • Build a dictionary from a corpus and convert a sentence to an array of 0 and 1

  $S1$: *Without music life would be a mistake*

  $S2$: *Radiohead are a great music band*
Vectorization Methods

• TF-IDF
  • Term frequency: how likely to find a word in the corpus?
  • Inverse document frequency: how unique is a word in the corpus?

\[ w_{x,y} = tf_{x,y} \times \log \left( \frac{N}{df_x} \right) \]

tf_{x,y}: frequency of x in y

df_x: number of documents containing x

N: total number of documents
Vectorization Methods

- Word embedding
  - Map words to a high-dimensional space where similar words are close to each other
Vectorization Methods

- Learning word embeddings

Vectorization Methods

• Language models (or contextualized word embeddings)
  • Build a vector for each word conditioned on its context
NLP Libraries and Pre-trained Models

- spaCy
- NLTK
- Stanford CoreNLP
- TextBlob
- Gensim
- Hugging Face
Semantic Parsing

• The task of converting a NL utterance to a logical form, e.g., SQL
Semantic Parsing

• Slot-filling systems  
  • Rule-based intent detection  
  • Template-based code generation

• Neural machine translation (Seq2Seq)  
  • Encoder-decoder model architecture  
  • Attention-based, e.g., transformers!

Shallow semantic parsing

Deep semantic parsing
Grammar-based Semantic Parsing

- Combinatory Categorical Grammar (CCG)
  - A lexicon and a set of grammar rules

A simple lexicon

<table>
<thead>
<tr>
<th></th>
<th>syntactic type</th>
<th>semantic type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utah</td>
<td>$NP : \text{utah}$</td>
<td></td>
</tr>
<tr>
<td>Idaho</td>
<td>$NP : \text{idaho}$</td>
<td></td>
</tr>
<tr>
<td>borders</td>
<td>$(S\backslash NP)/NP : \lambda x.\lambda y.\text{borders}(y, x)$</td>
<td></td>
</tr>
</tbody>
</table>

A derivation tree of “Utah borders Idaho”

Simple functional application rules

$A/B : \text{f} \quad B : \text{g} \implies A : \text{f}(g)$

$B : \text{g} \quad A\backslash B : \text{f} \implies A : \text{f}(g)$

Azaria et al. *Instructable Intelligent Personal Agent*. AAAI 2016.
## CCG Semantic Parsing

- **Combinatory Categorical Grammar (CCG)**
  - A lexicon and a set of grammar rules

<table>
<thead>
<tr>
<th>Word</th>
<th>Syntactic Category</th>
<th>Logical Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>set</td>
<td>((S/PP.StringV)/MutableField)</td>
<td>((\lambda x\ y\ (\text{setFieldFromString}\ x\ y)))</td>
</tr>
<tr>
<td>to</td>
<td>PP.StringV/StringV</td>
<td>((\lambda x\ x))</td>
</tr>
<tr>
<td>subject</td>
<td>FieldName</td>
<td>subject</td>
</tr>
<tr>
<td>send</td>
<td>S/InstanceName</td>
<td>((\lambda x\ (\text{send}\ x)))</td>
</tr>
<tr>
<td>email</td>
<td>InstanceName</td>
<td>email</td>
</tr>
<tr>
<td>set</td>
<td>((S/PP.FieldVal)/MutableField)</td>
<td>((\lambda x\ y\ (\text{setFieldFromFieldVal}\ x\ y)))</td>
</tr>
<tr>
<td>to</td>
<td>PP.FieldVal/FieldVal</td>
<td>((\lambda x\ x))</td>
</tr>
</tbody>
</table>

Azaria et al. *Instructable Intelligent Personal Agent*. AAAI 2016.
PCCG Semantic Parsing

- Extend CCG with a probabilistic model $P(L, T|S)$
  - A conditional distribution over possible $(L, T)$ pairs for a given sentence $S$
- Parameterized by $\theta$

$$P(L, T|S; \bar{\theta}) = \frac{e^{f(L, T, S) \cdot \bar{\theta}}}{\sum_{(L,T)} e^{f(L, T, S) \cdot \bar{\theta}}}$$

$$\arg\max_{L} P(L|S; \bar{\theta}) = \arg\max_{L} \sum_{T} P(L, T|S; \bar{\theta})$$

- Handle ambiguity in natural language

Azaria et al. *Instructable Intelligent Personal Agent.* AAAI 2016.
Neural-based Semantic Parsing

• Encoder: understand the meaning of the input sentence
• Decoder: generate the corresponding logic form
Relation-Aware SQL Generation

• Encode database schemas via a relation-aware transformer

Relation-Aware SQL Generation

- Tree-structured decoder
- Guided by SQL grammar
- Predict a derivation rule at a time, not a token