Sensala: A Dynamic Semantics System for Natural Language Processing

Daniyar Itegulov
Australian National University
ITMO University
daniyar.itegulov@anu.edu.au

Ekaterina Lebedeva
Australian National University
ekaterina.lebedeva@anu.edu.au

Bruno Woltzenlogel Paleo
bruno.wp@gmail.com

Abstract

Here we describe SENSALA, an open source framework for the semantic interpretation of natural language that provides the logical meaning of a given text. The framework's theory is based on a lambda calculus with exception handling and uses contexts, continuations, events and dependent types to handle a wide range of complex linguistic phenomena, such as donkey anaphora, verb phrase anaphora, propositional anaphora, presuppositions and implicatures.

Title and Abstract in Russian

Sensala: Система динамической семантики для обработки естественного языка

В данной статье описывается Sensala – программная система семантической интерпретации естественного языка с открытым исходным кодом, позволяющая получить логический смысл текста. Теоретическим фундаментом для системы послужило лямбда исчисление с обработкой исключений, а использование контекстов, продолжений, событий и зависимых типов позволяет системе интерпретировать широкий спектр таких лингвистических явлений, как «ослиная» анафора, глагольная анафора, пропозициональная анафора, пресуппозиция и импликатура.

1 Introduction

Attempts towards a modern logic-based semantics for natural language can be traced back at least to Montague (1974). He provided a framework for interpreting a fragment of the English language using lambda calculus, giving birth to a new branch of natural language processing, with roots in formal logic. Although Montague's formalisation of the English language is rather limited and serves more as a proof-of-concept, his work was already sufficiently comprehensive to represent quantification and capture the nature of ambiguity.

In the following 40 years Montague's approach was further developed and extended with new techniques for handling various natural language phenomena. Recently, de Groote (2006) showed how to use continuations and contexts to handle dynamic phenomena while still retaining standard mathematical logic constructions (first-order logic on top of a simply typed lambda calculus à la Church (1940)). The lambda calculus of de Groote's framework was extended by Lebedeva (2012) with an exception raising and handling mechanism, which allowed cross-sentential anaphora and presupposition triggers to be formalized. Itegulov and Lebedeva (2018) further combined it with event semantics and dependent type semantics (Bekki, 2014) to represent verb phrase anaphora and propositional anaphora.

SENSALA is based on these recent theoretical advances, which are partly summarized in sections 2 and 3. The linguistic phenomena handled by SENSALA are discussed in section 4 and its architecture is described in section 5. SENSALA has been deployed and can be used through the web interface available at http://sensala.cecs.anu.edu.au.

Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations, pages 123–127
Santa Fe, New Mexico, USA, August 20-26, 2018.

2 Dynamic Semantics

SENSALA implements the dynamic semantics framework introduced by de Groote (2006) and extended by Lebedeva (2012). The theory is built upon three atomic types: ι, the type of *individuals* (a.k.a. *entities*), o, the type of propositions, and γ, the type of left contexts. The right context is represented as a *continuation* of type $\gamma \to o$. A semantical interpretation of a single sentence has type $\gamma \to (\gamma \to o) \to o$.

de Groote (2006) focused on the representation of anaphora assuming that it has already been solved by some oracle operators, such as $sel_{he}, sel_{she}, sel_{it}$. These operators extract an entity from a left context passed to them and hence have type $\gamma \to \iota$. de Groote and Lebedeva (2010) proposed to view the context as a finite list of entities together with their properties. If c is such a list and $(a, man(a))$ is a new pair, then $(a, male(a)) :: c$ the new list obtained by pre-pending the pair to the list. Consider the following example from (de Groote, 2006):

John loves Mary. He smiles at her. (1)

These two sentences can be individually interpreted as formulae in (2)[1]. Each of these interpretations can be constructed compositionally by interpreting lexical components of the respective sentences. For example, proper name *John* has interpretation $\lambda\psi c.\psi(\mathbf{j}, (\mathbf{j}, m(\mathbf{j})) :: c)$ and pronoun *he* has interpretation $\lambda\psi c.\psi(sel_{he}(c), c)$, where ψ are continuations of type $\iota \to \gamma \to o$.

$$\begin{aligned} &\lambda c\phi.\textbf{love}(\mathbf{j}, \mathbf{m}) \wedge \phi((\mathbf{m}, f(\mathbf{m})) :: (\mathbf{j}, m(\mathbf{j})) :: c) \\ &\lambda c\phi.\textbf{smile}(sel_{he}(c), sel_{she}(c)) \wedge \phi(c) \end{aligned} \tag{2}$$

The sequential composition of interpretations in (2) leads to the following normal form:

$$\begin{aligned} &\lambda c\phi.\textbf{love}(\mathbf{j}, \mathbf{m}) \wedge \textbf{smiles}(sel_{he}((\mathbf{m}, f(\mathbf{m})) :: (\mathbf{j}, m(\mathbf{j})) :: c), sel_{her}((\mathbf{m}, f(\mathbf{m})) :: (\mathbf{j}, m(\mathbf{j})) :: c)) \\ &\quad \wedge \phi((\mathbf{m}, f(\mathbf{m})) :: (\mathbf{j}, m(\mathbf{j})) :: c) \end{aligned}$$

After meta-interpretation of the sel-operators, we obtain the following interpretation of discourse (1):

$$\lambda c\phi.\textbf{love}(\mathbf{j}, \mathbf{m}) \wedge \textbf{smiles}(\mathbf{j}, \mathbf{m}) \wedge \phi((\mathbf{m}, f(\mathbf{m})) :: (\mathbf{j}, m(\mathbf{j})) :: c)$$

3 Event Semantics

Event semantics was first described by Davidson (1967) and then extended by Parsons (1990). The resulting neo-Davidsonian event semantics introduces a new atomic type for events e and a few thematic predicates for describing properties of events (e.g. $agent$, $patient$). Consider, for example, the sentence and its interpretation according to neo-Davidsonian event semantics in (3), where predicates $agent$ and $patient$ indicate the event's participants, while $yesterday$ indicates when the event happened:

$$\begin{aligned} &\textit{John met Mary yesterday}. \\ &\exists e^{e}.met(e) \wedge agent(e, \mathbf{j}) \wedge patient(e, \mathbf{m}) \wedge yesterday(e) \end{aligned} \tag{3}$$

4 Linguistic Phenomena handled by Sensala

Pronominal anaphora are phenomena in which the interpretation of a pronoun depends on an antecedent expression in the left context. Currently, SENSALA can interpret most English personal pronouns. For example, the pronoun "*he*" is interpreted into a selection of an entity with the property $\lambda x.man(x)$ from the left context and the pronoun "*it*" is interpreted into a selection of an entity with the property $\lambda x.\neg person(x)$. Then, after extracting the hypernym relationship, as discussed in section 5.3, SENSALA interprets discourse (4) as (5).

John owns a dog. He loves it. (4)

$$\begin{aligned} &\exists d^{en}.dog(d) \wedge \exists e^{ev}.owns(e) \wedge agent(e, j) \wedge patient(e, d) \\ &\wedge \exists e'^{ev}.loves(e') \wedge agent(e', j) \wedge patient(e', d) \end{aligned} \tag{5}$$

[1] **j** stands for the entity *John* and **m** stands for the entity *Mary*. The predicates f and m represent being female and male.

Propositional anaphora are another type of anaphora, where an anaphoric clause is used to refer to a whole proposition (e.g., a sentence). SENSALA interprets the demonstrative pronoun "that" in, for example, (6) into selection of an event from the left context. Thus, SENSALA interprets (6) as (7):

John loves Mary. I heard that from Bob. (6)

$$\begin{aligned}&\exists e^{ev}.loves(e) \wedge agent(e, j) \wedge patient(e, m) \wedge \exists e'^{ev}.heard(e') \\ &\wedge agent(e', speaker) \wedge patient(e', e) \wedge from(e', b)\end{aligned} \tag{7}$$

Verb phrase anaphora involve omissions of a full-fledged verb phrase when the ellipsed part can be implicitly derived from the context. The interpretation of verb phrase anaphora is more challenging than the interpretation of propositional and pronominal anaphora: an anaphoric clause in a verb phrase anaphora usually talks about a new event that inherits some properties of another event. For example, the second sentence in (8) talks about an event that inherits the property of being a "leaving" event while also changing the property of being performed by John to the property of being performed by Mary. SENSALA interprets (8) as (9):

John left. Mary did too. (8)

$$\exists e^{ev}.left(e) \wedge agent(e, j) \wedge \exists e'^{ev}.left(e') \wedge agent(e, m) \tag{9}$$

Donkey anaphora may occur when the syntactic structure of a sentence does not conform to its meaning. The classical example of donkey anaphora is (10), which SENSALA interprets as (11) using techniques in line with the approach described by de Groote (2006).

Every farmer who owns a donkey beats it. (10)

$$\begin{aligned}&\forall f^{en}.farmer(f) \rightarrow \forall d^{en}.donkey(d) \rightarrow \\ &\forall e^{ev}.owns(e) \wedge agent(e, f) \wedge patient(e, d) \rightarrow \exists e'^{ev}.beats(e') \wedge agent(e', f) \wedge patient(e', d)\end{aligned} \tag{11}$$

Implicature is something conveyed in a discourse but not explicitly stated by the discourse. Currently, SENSALA supports deductive implicatures, whose implicit meaning can be derived using classical logic inference rules, but not abductive implicatures, which would require non-monotonic logics. A deductive implicature can be observed in (12), where the implicature is the logically deduced fact "John owns a donkey".

Every farmer owns a donkey. John is a farmer. (12)

SENSALA uses the automated theorem prover SCAVENGER (Itegulov et al., 2017) to derive new knowledge from the discourse's interpretation.

5 Software Architecture and Implementation

The architecture of SENSALA has been designed in accordance with software engineering, functional programming and object-oriented programming principles such as immutability, modularity and referential transparency. The adherence to these principles has been facilitated by the use of the hybrid programming language Scala. The source code is available in GitLab at `https://gitlab.com/aossie/Sensala` under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

SENSALA has five main modules:

- `core` module contains all basic data structures (e.g. lambda terms, types and left context) and all natural language syntax trees with their interpretation functions.
- `parser` module contains a transformer from a text to its natural language syntax tree.
- `wordnet` module contains an interaction with the WordNet database for extracting relationships between words (e.g. hypernym, synonym).
- `cli` module contains a simple way to interact with SENSALA from the command line.

- `web` module contains a web server with a user interface (UI) and an application program interface (API) for interacting with SENSALA.

Figure 1 shows SENSALA's execution pipeline, with stages and corresponding modules. The following sections describe three main modules of SENSALA.

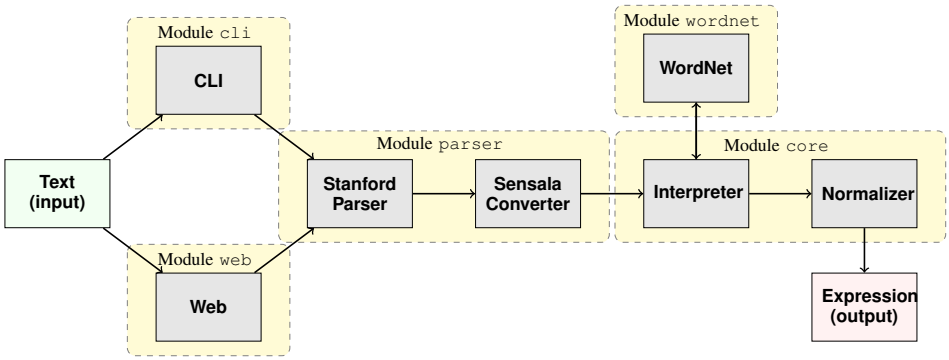

Figure 1: SENSALA execution pipeline

5.1 Core

Every single phrase interpreted in SENSALA is represented by one of the *natural language classes*. The origin trait for all natural language constructions is the `NL` trait. Second-level traits are English language parts of speech. SENSALA currently supports the interpretation of common nouns, proper nouns, definite and indefinite articles (represented by `QuantNounPhrase` class), pronouns, transitive and intransitive verbs, adjectives, adverbs and some wh phrases. `Discourse` is a class representing a sequential combination of sentences, while a sentence is a noun phrase accompanied by a verb phrase.

5.2 Parser

SENSALA uses the Stanford Parser (Klein and Manning, 2002) to retrieve a Penn-tagged tree from raw text. As the Stanford Parser's output trees differ from the classes described in section 5.1. SENSALA implements a `DiscourseParser` to convert Stanford Parser trees into SENSALA syntax trees.

5.3 WordNet

WordNet (Fellbaum, 1998) is used in SENSALA to extract hypernym relationships (a.k.a. is-a relationships) between common nouns in text. SENSALA uses JWNL library to interface with the WordNet database. The library provides a way to extract hypernym relationship trees from the database. For example, the tree for the word "*farmer*" contains hypernyms "*creator*",' '*person*" and "*organism*"; and the tree for the word "*donkey*" contains hypernyms "*ass*", "*mammal*" and "*animal*".

After retrieving all hypernyms of "*farmer*" and "*donkey*", SENSALA interprets the discourse "*A farmer owns a donkey. He loves it.*" successfully. The entity farmer has the property of being a person (according to the WordNet hypernym tree), which is required by "*he*"; and the entity donkey has a property of being an animal, which is one of the satisfying properties for the pronoun "*it*".

6 Conclusion

SENSALA is a new open source logic-based system for formal semantics of natural language. Although it is still at an early stage of development, SENSALA can already handle various complex linguistic phenomena such as some pronominal anaphora, propositional anaphora, verb phrase anaphora, donkey anaphora, presuppositions and implicatures. It currently supports subsets of English and German. Planned future work includes support of other natural and domain-specific controlled natural languages.

Given that SENSALA is being developed with rigorous software engineering principles in mind and with the ambition of being more than just a prototype, and given the scarcity of tools for formal semantics, we hope SENSALA will become a widely used and useful tool in this research field.

References

Daisuke Bekki. 2014. Representing anaphora with dependent types. In *Logical Aspects of Computational Linguistics*, pages 14–29. Springer Berlin Heidelberg.

Alonzo Church. 1940. A formulation of the simple theory of types. *The Journal of Symbolic Logic*, 5:56–68.

Donald Davidson. 1967. The logical form of action sentences. In *The Logic of Decision and Action*. University of Pittsburgh Press.

Philippe de Groote and Ekaterina Lebedeva. 2010. Presupposition accommodation as exception handling. In *Proceedings of the SIGDIAL 2010 Conference*, Tokyo, Japan, September. Association for Computational Linguistics.

Philippe de Groote. 2006. Towards a montagovian account of dynamics. In *Semantics and Linguistic Theory XVI*.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge.

Daniyar Itegulov and Ekaterina Lebedeva. 2018. Handling verb phrase anaphora with dependent types and events. In *Logic, Language, Information, and Computation – WoLLIC 2018*.

Daniyar Itegulov, John Slaney, and Bruno Woltzenlogel Paleo. 2017. Scavenger 0.1: A theorem prover based on conflict resolution. In *Automated Deduction – CADE 26*, pages 344–356. Springer International Publishing.

Dan Klein and Christopher D. Manning. 2002. Fast exact inference with a factored model for natural language parsing. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, pages 3–10. MIT Press.

Ekaterina Lebedeva. 2012. *Expression de la dynamique du discours à l'aide de continuations*. Ph.D. thesis, Université de Lorraine.

Richard Montague. 1974. *Formal Philosophy; Selected Papers of Richard Montague*. New Haven: Yale University Press.

Terence Parsons. 1990. *Events in the Semantics of English: A Study in Subatomic Semantics*. MIT Press.