МОДУЛЬНАЯ КОНЦЕПЦИЯ ОБОБЩЕННОГО НЕЙРОСЕТЕВОГО СЕМАНТИЧЕСКОГО АНАЛИЗАТОРА

Д.Е. Понявин

Белорусский государственный университет, г. Минск;
pndmev@gmail.com;
науч. рук. – А.А. Буславский, ст. преп.

Семантический анализ естественно-языковых текстов – неотъемлемая часть коммуникации. Основная сложность заключается в отсутствии строгой формализации принципов функционирования естественных языков в силу их эволюционного развития. Автоматизация данного процесса крайне важна для построения эффективных коммуникативных интерфейсов между человеком и компьютером, используемых, в частности, при реализации различного рода ассистентов: Яндекс Алиса, Google Assistant и др. Инструментом автоматизации могут выступать искусственные нейронные сети, на базе которых разработана уникальная архитектурная концепция семантического анализатора, обладающая свойствами расширяемости, адаптируемости и универсальной применимости при решении задач семантического анализа текстов. Предложенная концепция реализована на практике в рамках библиотеки NNSAtool, доступной на сервисе PyPI.

Ключевые слова: нейросетевой семантический анализ, обобщенный семантический анализатор, модульный семантический анализатор, библиотека NNSAtool.

ФОРМАЛИЗАЦИЯ ПРОЦЕССА СЕМАНТИЧЕСКОГО АНАЛИЗА

Семантический анализ текста – это процесс получения *семантического представления* (смыслового описания) исходного текста. При реализации автоматизированного процесса семантического анализа результирующее представление должно иметь формальный вид – последовательность C *дескрипторов* текста (элементарных частей описания). Соответственно, $|C|$ – количество дескрипторов в семантическом описании. Обозначим, множество всех возможных дескрипторов как C^0.

Различают два типа задач семантического анализа текстов [1]: классификация текста в целом и классификация отдельных составляющих текста. Элементарной составляющей текста является *токен*, который может быть представлен в виде целого слова или его части: слога или нескольких слогов. Таким образом, исходный текст может быть представлен в виде упорядоченной последовательности токенов. Данная последовательность есть *токенизированное представление* текста.

Здесь и далее понятия текста и его токенизированного представления отождествляются и обозначаются как T. Упорядоченное множество всех возможных токенов обозначается как T^0. Отметим, что $|T|$ – общее количество токенов текста T.

В контексте токенизированного представления возможны два типа задач семантического анализа:

1. Задачи *агрегации семантики*: получение совокупного семантического представления текста T – номера класса, которому принадлежит текст T по семантическому признаку ($|C| = 1$);
2. Задачи *распределения семантики*: получение семантического представления текста T, состоящего из последовательности дескрипторов для каждого его токена – номеров классов, которым принадлежат токены по семантическому признаку ($|C| = |T|$).

Примечание: В случае необходимости классификации определенной части текста вводится специальный класс для токенов, не подлежащих классификации, что позволяет перейти к случаю 2.

Традиционно при решении задач области обработки естественных языков (NLP – Natural Language Processing) при помощи нейронных сетей используется слой вложения (Embedding) или иной модуль, реализующий подобное преобразование [2]. Преобразование вложения F_{embed} ставит в соответствие последовательности целых чисел D последовательность векторов E пространства заданной размерности. Последовательность D – *первичное представление* текста T, получаемое в результате входного преобразования F_{in}, на вход которого подается токенизированное представление T, где каждому токену ставится в соответствие его порядковый номер в упорядоченном множестве T^0.

Упомянутая ранее векторная последовательность E – *вторичное представление* текста T – может находиться в одном из трех состояний:

- Распределенное: $E = E_D$, где $|E_D| = |T|$;
- Агрегированное: $E = E_A$, где $|E_A| = 1$;
- Смешанное: $E = E_M$, где $E_M = (E_D, E_A)$.

Как было сказано ранее, при решении задач агрегации имеем $|C| = 1$, а при решении задач распределения – $|C| = |T|$. Так, при использовании не соответствующих состояний вторичного представления E (E_D для задач агрегации, E_A – для задач распределения, E_M – для любых задач), требуется применение дополнительного распределяющего (F_{distr}) или агрегирующего (F_{aggrg}) преобразования. На выходе F_{distr} и F_{aggrg} формируется векторная последовательность H – *скрытое семантическое представление* текста T – длины $|T|$ и 1 соответственно.

Последовательность номеров классов C – *внешнее семантическое представление* – является результатом выходного преобразования F_{out}, позволяющее получить для каждого вектора входной последовательности соответствующе целое число – номер класса. Традиционно размерность векторов совпадает с количеством возможных классов $|C^0|$, а позиция наибольшей

векторной компоненты соответствует номеру класса [1]. Например, результирующая векторная последовательность может состоять из вероятностных векторов. В любом случае преобразование F_{out} реализуется через функцию взятия аргумента максимума по вектору.

С целью согласованности размерности векторов последовательностей E и H с заданной размерностью входных векторов преобразований F_{distr} (или F_{aggrg}) и F_{out} соответственно вводятся дополнительные преобразования транзитного типа F_{trans}, F_{intra} и F_{incon}, для которых имеем:

$$F_{trans}: \{H\} \to \{P\},\ F_{intra}: \{E_D\} \to \{E'_D\},\ F_{incon}: \{E_A\} \to \{E'_A\}$$

Так, преобразование транзита F_{trans} ставит в соответствие векторной последовательности H последовательность P векторов размерности $|C^0|$ (*внутреннее семантическое представление*), причем $|H| = |P|$ – основное требование к преобразованиям транзитного типа. На выходе преобразований внутреннего транзита F_{intra} и внутренней конверсии F_{incon} формируется *третичное представление* текста T распределенного (E'_D, где $|E_D| = |E'_D|$) и агрегированного (E'_A, где $|E_A| = |E'_A|$) типа соответственно. В случае использования смешанного вторичного представления E_M оно предварительно разбивается на два представления: распределенное (E_D) и агрегированное (E_A), к каждому из которых может быть применено соответствующее преобразование транзитного типа.

ОПИСАНИЕ МОДУЛЬНОЙ КОНЦЕПЦИИ

Данная архитектурная концепция основывается на результатах описанной выше формализации процесса семантического анализа. Так, каждому приведенному преобразованию соответствует модуль, который его реализует. В таблице 1 показано данное соответствие.

Таблица 1

Соответствие преобразований и реализующих модулей

Преобразование	Модуль	Описание (англ.)	Описание (рус.)
F_{in}	PREPR	Pre-processing	Предобработка
F_{embed}	EMBED	Embedding	Вложение
F_{intra}	INTRA	Internal transition	Внутренний транзит
F_{incon}	INCON	Internal conversion	Внутренняя конверсия
F_{distr}	MIDDL	Middle	Средний
F_{aggrg}			
F_{trans}	TRANS	Transition	Транзит
F_{out}	PSTPR	Post-processing	Постобработка

Примечание: в силу общности интерфейса взаимодействия с внешними модулями для потенциальных модулей DISTR и AGGRG, реализующих преобразования F_{distr} и F_{aggrg}, введено понятие среднего модуля MIDDL, экземплярами которого они являются.

В рамках модульной концепции обобщенного нейросетевого семантического анализатора стоит выделять два уровня:

- Статический – фиксированное расположение модулей
- Динамический – переменное расположение модулей

Так, для модулей статического уровня расположение изначально определяется архитектурой обобщенного семантического анализатора. В свою очередь для модулей динамического уровня расположение необходимо задать на этапе проектирования конкретного семантического анализатора. Все модули, упомянутые ранее, относятся к статическому уровню, архитектура которого представлена на рисунке 1.

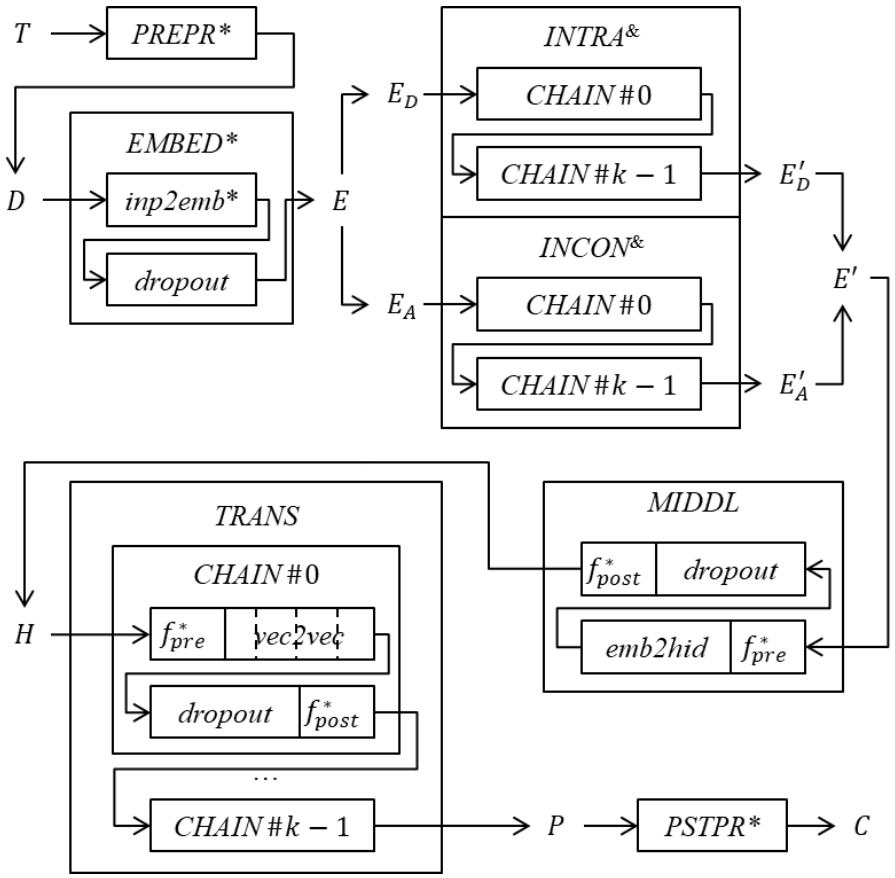

Рис. 1. Схема статического уровня модульного семантического анализатора

Примечание:

- обязательные модули и иные компоненты отмечены символом «звездочка» (*); модули, отмеченные символом «амперсанд» (&), могут быть непустыми только при наличии непустого модуля MIDDL;

- *inp2emb*, *emb2hid* – нейросетевые модули, *dropout* – слой исключения (Dropout), *vec2vec* – последовательность согласованных однотипных нейросетевых модулей, f_{pre} и f_{post} – преобразования локальной пред- и постобработки.

Примером модуля, располагаемого на динамическом уровне, является модуль CNNCT (connection module – модуль соединения). Данный модуль является объединением двух нейросетевых концепций, используемых для обеспечения большей гибкости в глубоких нейросетевых архитектурах: соединений быстрого доступа (skip connections, shortcuts) [3] и механизма внимания (Attention) [4].

Точное расположение модуля CNNCT определяется при создании конкретного семантического анализатора парой модулей: отправитель (*sender*) и приемник (*receiver*). Модуль CNNCT обеспечивает соединение между отправителем и приемником параллельно другим модулям, находящимся между ними на статическом уровне. Посредством данного модуля могут быть переданы данные полностью или частично по принципу соединения быстрого доступа, а также вектор весовых коэффициентов, вычисленный на основе текущего контекста, или результирующий вектор, полученный из векторной последовательности в соответствии с вычисленными весами по принципу механизма внимания.

Представленная архитектура имеет ряд потенциальных точек расширения системы, таких как настраиваемые локальные преобразования пред- и постобработки, возможность использования собственных нейросетевых компонент в рамках соответствующих модулей. На основе предложенной модульной концепции обобщенного нейросетевого семантического анализатора реализована библиотека NNSAtool, доступная на сервисе PyPI. Подробнее об универсальной практической применимости и вариантах адаптации предложенной архитектурной концепции сказано в тексте дипломной работы [5].

Библиографические ссылки

1. A Deep Architecture for Sentiment Analysis of News Articles / D. Nguyen [et al.]; Ho Chi Minh City University of Technology. – Ho Chi Minh, 2019. – 12 p.
2. Almeida, F. Word Embeddings: A Survey / F. Almeida, G. Xexéo; Federal University of Rio de Janeiro. – Rio de Janeiro, 2019. – 10 p.
3. Skip Connections Matter: On the Transferability of Adversarial Examples Generated with ResNets / D. Wu [et al.]; The 8th International Conference on Learning Representations (ICLR 2020). – Addis Ababa, 2020. – 15 p.
4. Attention Is All You Need / A. Vaswani [et al.]; The 31st Conference on Neural Information Processing Systems (NIPS 2017). – Long Beach, 2017. – 11 p.
5. Понявин, Д.Е. Решение задач семантического анализа текстов при помощи искусственных нейронных сетей / Д.Е. Понявин; Белорусский государственный университет. – Минск, 2021. – 63 стр.