

# Семантические принципы разработки текстовых систем искусственного интеллекта

к.т.н. В. С. Блюм, к.т.н. В. М. Космачёв, к.т.н. О. М. Поляков

Санкт-Петербургский государственный университет аэрокосмического приборостроения

Санкт-Петербург, Россия

vladblum7@gmail.com, kvm@aanet.ru, polyakovoleg@yandex.ru

**Аннотация.** Целью статьи является определение теоретических и технологических основ вычисления семантики текстов. Доказано, что любое предложение, имеющее безусловное правое отрицание, не имеет смысла. Рассмотрены принципы построения основных современных генераторов «осмысленных» текстов, последовательных систем искусственного интеллекта и способы их применения. Рассмотрены структура и алгоритмы рекуррентных нейронных сетей и универсального нейросетевого вычислительного механизма Transformer. Новизна подхода состоит в том, что в основу вычисления семантики генерируемого текста положена большая языковая модель.

**Ключевые слова:** семантика, языковая модель, искусственный интеллект.

## ВВЕДЕНИЕ

Синтез лингвистической модели является одной из центральных проблем обработки естественного языка (Natural Language Processing, NLP) [1] — общего направления искусственного интеллекта (ИИ) и математической лингвистики.

Одной из основных теорий лингвистики, используемых в синтезе лингвистической модели искусственного интеллекта, является теория формальных грамматик, определяющая правила построения предложений и языковых конструкций, а также правила определения значения слов и действий внутри предложения [2].

Для синтеза лингвистической модели искусственного интеллекта используются такие методы компьютерных наук, как алгоритмы машинного обучения и нейронные сети [3]. Эти методы позволяют обучать модель распознаванию и анализу языковых конструкций и текстов.

В целом синтез лингвистической модели искусственного интеллекта основывается на изучении естественных языков и применении теории компьютерных наук для создания интеллектуальных систем, способных анализировать и обрабатывать текстовую информацию на естественном языке.

## СЕМАНТИЧЕСКИЕ АСПЕКТЫ И ПРОБЛЕМЫ ПОСЛЕДНИХ ВЕРСИЙ ИИ

Для того, чтобы понять возможности и недостатки систем ИИ типа большой языковой модели GPT 4 (Generative Pre-trained Transformer 4), необходимо разобраться с общим вопросом формирования семантики.

Заметим, что в качестве объекта обучения в таких системах выступает рекуррентная нейросеть. Это означает, что прогноз в таких моделях производится не только на основе поступающих извне параметров, но и на основе предыдущего состояния сети. Это происходит независимо

от числа используемых параметров, выделенных системой вниманием.

Итак, пусть в нашем распоряжении есть некоторый естественный язык и  $\mathcal{L}$  — множество предложений в этом языке. Хотя в GPT 4 прогнозу подвергаются слова, предположение о характере множества  $\mathcal{L}$  не является существенным, поскольку при этом сохраняется последовательно-рекуррентная структура интерпретации текста.

Под предложением мы понимаем последовательность слов языка, которая может быть интерпретирована (отражена в модели). Это означает существование некоторого оператора интерпретации (осмысления)  $\Psi$ , который перерабатывает предложения в семантику или смысл. Мы фактически называем предложением такие конечные последовательности слов языка, которые входят в область определения оператора  $\Psi$ .

Оператор интерпретации  $\Psi$  интерпретирует предложение  $s \in \mathcal{L}$  с учетом уже накопленной к моменту интерпретации  $s$  семантики  $C$  (предшествующие события или предложения могут изменять или уточнять интерпретацию  $s$ ) и формирует новый смысл  $C'$ , включающий и смысл предложения  $s$ . Итак, «текущая» семантика после осмысления предложения  $s$  определяется как  $C' = \Psi(C, s)$ .

В частности, при отдельной интерпретации предложения без предварительных смыслов получаем  $C' = \Psi(\emptyset, s)$ . Под отдельной интерпретацией предложения  $s$  здесь понимается такая интерпретация, которой не предшествует некоторая семантика в отношении предложения  $s$ . Например, когда мы приступаем к чтению СМС-сообщения с неизвестного номера телефона, мы имеем нулевую начальную семантику.

К этому моменту мы не знаем, как работает эта интерпретация и из чего состоит семантика, поэтому  $C'$  пока обозначает не множество, а некую семантическую структуру, которая получается после интерпретации  $s$ . В частности,  $\emptyset$  обозначает здесь не пустое множество, а пустую семантическую структуру, которая соответствует отсутствию смысла. Разумеется, у каждого человека или ИИ в конкретный момент времени свой оператор интерпретации. Он зависит от модели мира, от настроения, желаний и т. п.

Например, интерпретация одного и того же текста человеком в состоянии сильного испуга может существенно отличаться от интерпретации в благодушном расположении духа. Но все перечисленные факторы на конкретный момент интерпретации фиксированы, изменяется только накапливаемый смысл, так что на момент интерпретации предложения у каждого интеллекта действует какой-то конкретный оператор  $\Psi$ .

Мы перевели проблему множественности факторов, влияющих на формирование смысла, на оператор  $\Psi$ . Например, формируемый смысл зависит от действующей на момент интерпретации модели мира  $M$ , что отмечено в [1]: «Мы не можем судить о том, что человек хочет сказать, пока мы не знаем его взглядов, в то же время мы не можем судить, каковы взгляды человека, пока мы не знаем того, что он говорит». Тот факт, что мы перенесли эту зависимость на  $\Psi$ , означает: мы полагаем, что за время интерпретации текста модель мира не меняется. Это означает, что в мозгу человека или в модели ИИ имеется долговременная память (результат обучения), в которой хранится модель мира (для GPT 4 — модель языка), и оперативная память, в которой накапливаются текущие смыслы.

Кроме того, существуют временные периоды, когда накопленные смыслы пересматриваются, чтобы скорректировать модель мира. Например, у человека это может происходить во время сна, а у ИИ — в период обучения. После обучения ИИ имеет обновленную модель мира  $M'$ , что соответствует обновленному оператору  $\Psi'$ .

Иногда возникают моменты, когда человеку как бы «открывают глаза» на происходящее. Другими словами, он получает такие важные смыслы, что возникает необходимость срочно скорректировать модель мира, после чего интерпретация им того же текста станет совершенно другой. Но это бывает редко, и мы не будем рассматривать случай изменения оператора  $\Psi$  в процессе интерпретации.

Описанный выше механизм вполне соответствует «герменевтическому кругу», суть которого сформулировал один из родоначальников герменевтики Фридрих Шлейермахер (Friedrich Schleiermacher) [1]: «...как целое понимается из отдельного, но и отдельное может быть понято только из целого, имеет такую важность для данного искусства и столь неоспоримо, что уже первые же операции невозможно проделать без применения его, да и огромное число герменевтических правил в большей или в меньшей степени основывается на нем...». Другими словами, получаемые «частные смыслы» через оператор интерпретации зависят от модели мира, которая изменяется этими смыслами и изменяет оператор интерпретации, что приводит к изменению получаемых смыслов от тех же текстов и так далее по кругу.

На множестве предложений языка  $\mathcal{L}$  определена операция конкатенации (приписывания —  $\bullet$ ), которая к одному предложению приписывает другое так, что в результате получается некоторый текст. Под текстом  $s = s_1 \bullet \dots \bullet s_n$  будем понимать конечную последовательность предложений из  $\mathcal{L}$ .

Будем считать, что интерпретация текста  $s = s_1 \bullet \dots \bullet s_n$  происходит следующим образом. Сначала интерпретируется первое предложение  $C_1 = \Psi(\emptyset, s_1)$ . Предполагается, что предварительного смысла в отношении текста перед началом интерпретации нет. На основе интерпретации первого предложения интерпретируется второе  $C_2 = \Psi(C_1, s_2)$ , на основе интерпретации первых двух интерпретируется третье предложение и так далее до  $C_n$ .

Здесь необходимо сделать одно замечание. Несмотря на то, что мы определили текст как последовательность предложений, все же правильнее было бы понимать под текстом абзац. Однако неопределенность, связанная с этим семантическим понятием, не позволяет дать его строгое определение, не сводящееся к тавтологии типа «абзац — отрезок

письменной речи между двумя красными строками», что фактически означает: абзац — это то, что выделено как абзац. Обычно интерпретация происходит по абзацам, которые в устной речи выделяются более длительными паузами. Разбиение устной речи на абзацы хорошо видно, когда человек говорит под перевод, делая паузу и как бы приглашая переводчика приступить к переводу абзаца.

Если предложение является единицей интерпретации, то абзац — единицей завершенной мысли. Завершение абзаца обычно означает, что говорящий дал на его взгляд достаточно сведений, чтобы слушающий мог завершить интерпретацию, задать недостающие для интерпретации вопросы и осуществить логический вывод.

Для теоретической семантики представляется крайне важным понимание оснований (правил), по которым говорящий определяет конец абзаца. В частности, для экспертных систем это сигнал к началу вывода, а точнее — полноценного вывода. Здесь уместно также напомнить о различиях между синхронным и последовательным переводами. Разница между ними заключается именно в отсутствии для синхронного переводчика абзацев как завершенных мыслей.

Как бы ни выглядела семантическая структура  $C$ , для естественных языков выполняются две аксиомы.

**Первая аксиома** утверждает, что в языке существует пустое предложение  $e$ , приписывание которого к любому тексту справа и слева не меняет интерпретации текста. Это означает, что  $e$  само имеет пустую семантику ( $\Psi(C, e) = C$  для любого  $C$ ) и не изменяет семантику любого текста:  $\Psi(C, e \bullet s) = \Psi(C, s \bullet e) = \Psi(C, s)$ .

Например, если после текста (перед текстом) идет чистая бумага, потому что текст закончился (не начался), то семантика текста не изменяется. Это семантическое правило в языке отображается как  $s \bullet e = e \bullet s = s$ . В языке может существовать только одно пустое предложение в семантическом смысле, поскольку, если бы их было более одного (например,  $e$  и  $e'$ ), то  $e = e \bullet e' = e'$ . Отметим, что по определению пустое предложение интерпретируемо и имеет нулевой смысл (соответствует нулевой семантической структуре). Именно поэтому Word выделяет как ошибку появление в тексте лишних пробелов (которые не выполняют функцию разделения слов) и предлагает удалять эти пустые предложения.

**Вторая аксиома** (аксиома идемпотентности) утверждает, что повторение одного и того же текста  $s$  (предложения) не изменяет семантики текста:  $\Psi(C, s) = C' = \Psi(C', s)$  для любой семантики  $C$ . В языке это свойство интерпретации отражается равенством  $s = s \bullet s$ . Например, мы пропускаем повторно пропечатанный текст, поскольку он не несет дополнительной информации. Недаром Word выделяет как ошибку дублирование слова (предложения) в тексте.

На первый взгляд аксиома идемпотентности противоречит пословице «повторение — мать учения». Однако эта пословица означает, что результаты интерпретации впоследствии могут изменить модель и за счет этого скорректировать оператор интерпретации, так что в дальнейшем результаты интерпретации того же текста могут измениться (герменевтический круг). Однако в силу сделанного ранее замечания, на этапе интерпретации текста мы полагаем оператор  $\Psi$  неизменным. Это означает, что *внимание*

воспринимающего текст человека или ИИ в процессе интерпретации остается неизменным.

**Определение 1.** Будем говорить, что предложение  $s$  имеет правое отрицание  $s^{-1}$ , если интерпретация текста  $s \bullet s^{-1}$  при нулевом начальном смысле порождает пустой смысл («Я пойду в магазин. Я не пойду в магазин»). Другими словами, если  $C = \Psi(\emptyset, s)$ , то  $\Psi(C, s^{-1}) = \emptyset$ .

В языке обычно мы получаем отрицание предложения, когда ставим «не» перед глаголом. Семантическое свойство правого отрицания в языке выражается правилом  $s \bullet s^{-1} = e$ , где  $s$  — предложение и начальный смысл — нулевой.

**Определение 2.** Будем говорить, что предложение  $s$  имеет безусловное правое отрицание, если для любого смысла  $C$  имеют место равенства: если  $\Psi(C, s) = C'$ , то  $\Psi(C', s^{-1}) = C$ . Другими словами,  $\Psi(C, s \bullet s^{-1}) = C$ .

Это семантическое свойство предложения означает, что операция  $\bullet$  ассоциативна для текста  $s \bullet s^{-1}$ , то есть для любого текста  $t$  выполняется равенство  $(t \bullet s) \bullet s^{-1} = t \bullet (s \bullet s^{-1})$ . Разумеется, если операция  $\bullet$  в целом ассоциативна, то она порождает языки с безусловными правыми отрицаниями.

**Пример 1.** В естественных языках свойство безусловности правого отрицания в общем случае не выполняется. Так, в известном фильме «Берегись автомобиля» следователь Максим Подберезовиков, выступая в качестве свидетеля дежний Деточкина, заявляет: «Он, конечно, виноват, но он... не виноват». Эта фраза не вызывает у слушателя ощущения нулевого смысла, поскольку кроме самого фильма, формирующего определенную семантику, Подберезовиков перед этой фразой произносит предложения, которые создают семантику, сохраняющую непустую семантику этой фразы, а именно: он наделяет положительную часть фразы смыслом незаконности действий Деточкина, а отрицательную — смыслом справедливости его действий. В итоге смысл справедливости не может нейтрализовать смысл незаконности и наоборот.

Кроме того, из определения 2 не следует, что у каждого предложения языка должно быть отрицание. Например, у предложения «ну, вот» нет отрицания. В этом также одно из отличий отрицания определения 2 от отрицания в логике.

Безусловность правого отрицания обеспечивает закон двойного отрицания в языке. В самом деле,  $(s^{-1})^{-1} = e \bullet (s^{-1})^{-1} = (s \bullet s^{-1}) \bullet (s^{-1})^{-1} = s \bullet (s^{-1} \bullet (s^{-1})^{-1}) = s \bullet e = s$ . Если предположить, что логика была абстрагирована из языка, то, скорее всего, прототипом операции дизъюнкции была операция приписывания, а прототипом логического отрицания — операция безусловного отрицания. Как известно, операций дизъюнкции и отрицания достаточно для выражения всех функций алгебры логики. Вместе с тем, имеет место следующая теорема.

**Теорема 1.** Любое предложение, имеющее безусловное правое отрицание, не имеет смысла.

**Доказательство.** Пусть предложение  $s$  имеет безусловное правое отрицание  $s^{-1}$ , тогда  $s = s \bullet e = s \bullet (s \bullet s^{-1}) = (s \bullet s) \bullet s^{-1} = s \bullet s^{-1} = e$ .

**Следствие.** Если текст состоит из предложений, имеющих безусловные правые отрицания, то семантика первого предложения равна  $\Psi(C, s) = \Psi(C, e) = C$ .

То же получим для второго и последующего предложения текста. В частности, для  $C = \emptyset$  получаем пустую интер-

претацию текста. Если какие-либо предложения языка имеют безусловные правые отрицания, то верно следующее утверждение: в любом языке любой текст, составленный из предложений с безусловными правыми отрицаниями, не имеет смысла.

Поскольку предложения естественных языков имеют смысл, то безусловные правые отрицания в языке отсутствуют, а с ними не выполняется и правило двойного отрицания в том смысле, что два отрицания эквивалентны положительному утверждению.

Например, известно [4], что «не» с подлежащего или дополнения можно выносить на глагол, так что из предложения «я не пойду не в кино» следует «я не не пойду в кино». Из первого предложения следует, что человек может пойти в кино или куда-то не пойти, но, в соответствии со вторым предложением, при допущении двойного отрицания один из вариантов отпадает.

Проверим это в диалоге с виртуальным собеседником Chat GPT 4 (ИИ).

**Человек.** Что означает фраза «Я не пойду не в кино»?

**GPT 4.** Эта фраза имеет двойное отрицание, что означает, что человек пойдет в кино.

Итак, ИИ допускает двойное отрицание и, значит, в силу теоремы 1 может создавать пустую семантику даже там, где ее нет.

Возьмем теперь два предложения, которые формально отрицают друг друга. Как ИИ оценит семантику в этом случае?

**Человек.** Какая из фраз истинна: «У Кутузова был один глаз», «У Кутузова не было одного глаза»?

**GPT 4.** Как ИИ, я могу подтвердить, что обе фразы описывают отсутствие одного глаза у Кутузова. Разница только в формулировке: одна фраза утверждает, что у него был один глаз, а другая — что не было одного глаза. Но обе выражают одно и то же.

Итак, в случае с одинарным отрицанием GPT 4 справился отлично. И последняя проверка — на скрытые отрицания.

**Человек.** Что означает фраза «В моей практике не было такого, чтобы такого не было»?

**GPT 4.** Эта фраза означает, что в опыте говорящего не было случаев, которые бы полностью исключали возможность определенного события или явления. Другими словами, говорящий утверждает, что все возможно в его практике.

И мы опять видим, что GPT 4 при нескольких отрицаниях семантически «плывет». Это имеет прямое отношение к его способу формирования смыслов.

Теорема 1 очень важна в понимании семантики языка. Семантика «склеивает» предложения в единое целое. В этом ее важнейшая функция. Даже самая незначительная попытка разорвать (упростить) эту связь приводит к ликвидации семантики, а значит, и возможности интерпретации языка в некоторой модели. Действительно, наличие безусловного правого отрицания всего лишь позволяет проинтерпретировать текст  $s \bullet s^{-1}$  отдельно, независимо от контекста. Сохраняется порядок предложений в тексте, его последовательная интерпретация и т. д. И все же это разрушает семантику текста. В самом деле, пусть предложение  $s$  имеет семантику  $C$ . Аксиома идемпотентности

утверждает, что предложение  $s*s$  имеет такую же семантику  $S$ . То есть семантики предложений как бы накладываются друг на друга («сцепляются») и, если они совпадают, дают тот же результат. Но тогда приписка предложения-отрицания  $s^{-1}$  должна уничтожить семантику  $S$  независимо от того, как она была сформирована: через  $s$  или  $s*s$ . В итоге мы получаем нулевую семантику, которая соответствует пустому предложению. Но та же самая последовательность, вычисленная как  $s*(s*s^{-1})$ , порождает семантику  $S$ , а это значит, что семантика  $S$  нулевая. Иначе нам пришлось бы признать, что либо у одного и того же предложения могут быть разные семантики (при одинаковом  $\Psi$ ), либо семантики не «накладываются» друг на друга. То есть это бы означало, что семантика не «склеивает» смыслы предложений. Кроме того, это означает, что приписка  $s^{-1}$  формально воздействует не только на ближайшее слева предложение  $s$ , но и на предшествующее предложение  $s$ . Другими словами, приписывание отрицательных предложений может иметь эффект распространения влияния на интерпретацию предшествующего текста, а не только ближайшего предложения, что является одной из причин множества параметров в GPT 4, которые учитывают отдаленный уже прошедший текст.

Все сказанное ранее в полной мере относится к ассоциативным языкам, а предположение об ассоциативности сплошь и рядом встречается в лингвистике, хотя такие языки не могут иметь никакой интерпретации, если конечно в этих языках существует отрицание предложений.

Завершая семантический обзор для языковых моделей отметим еще два обстоятельства. Поскольку GPT 4 фактически не имеет модели мира, а имеет модель языка, которая является промежуточным, обслуживающим элементом для модели мира, у него отсутствует семантика, связанная с разным мышлением [5]. Поясним это на примере.

**Человек.** Что находится севернее Северного полюса?

**GPT 4.** Северный полюс является самой северной точкой на Земле, поэтому севернее него — это уже вода: Арктический океан с островами и архипелагами, такими как Шпицберген, Северная Земля, Земля Франца-Иосифа, Новая Земля и др.

По этой же причине GPT 4, как правило, плохо отгадывает загадки.

**Человек.** Отгадай загадку: без окон, без дверей — полна горница людей.

**GPT 4.** Попробуйте задать вопрос снова.

В дальнейшем при различных вариантах формулировки этого вопроса был получен тот же ответ.

Второе обстоятельство связано с коммуникативной семантикой, под которой понимаются смыслы, связанные со взаимодействием различных моделей у разных людей для реализации совместной деятельности. Поясним это на примере следующего диалога:

— Ну так как там, что там? Давай уже... Я тут уже вообще не знаю...

— Ну, понимаешь, там не очень... Там такое дело, что даже не знаю теперь...

— Что, совсем-совсем никак? Да?

— Ну, в общем, честно говоря, да!

— Ну и чего ж теперь? А? Вот ведь!

— Правда, знаешь, там может быть один еще вариант, но это если только Петр Михайлович сам...

— А! Ну так и ничего! Пускай Петр Михайлович... Не вопрос... Нет. А что такого? Серьезно. Это как-то даже и лучше будет.

В приведенном диалоге у двух собеседников есть модели обсуждаемой ситуации, и эти модели практически не отличаются. Первый собеседник интересуется, произошло ли событие, которое они прогнозировали исходя из своих моделей. Его интерес эмоциональный, потому что для него значимо достижение цели развития этой ситуации. По этой причине он недоволен затягиванием диалога вторым собеседником. Далее собеседники обсуждают коррекцию технологии достижения целевого состояния. Заметим, что здесь не обсуждается ни ситуация, ни ее признаки, ни модель, ни целевое состояние. Обсуждается неизвестная нам (но известная собеседникам) технология достижения цели, которая в какой-то момент реализации дала сбой из-за расхождения модельных прогнозов и реальной действительности.

Поскольку GPT 4 не имеет собственной модели мира, а только модель языка, он «понимает» коммуникативную семантику лишь в языковом смысле.

**Человек.** Ну ты даешь! Что, теперь вообще никак? И даже как раньше нельзя?

**GPT 4.** Извините, я не совсем понимаю ваш вопрос. Можете уточнить, о чем именно вы спрашиваете?

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ ИИ ДЛЯ ИЗВЛЕЧЕНИЯ СМЫСЛОВ

К числу современных инструментов для извлечения смыслов из текстов (вычисления семантики предложений) следует отнести последовательные системы искусственного интеллекта (ПСИИ) — это системы, в которых некоторые действия следуют за другими, и результат вычислений предыдущего шага служит входом для следующего.

Обобщенная схема применения ПСИИ включает:

1. Разбиение задачи на последовательность подзадач.

2. Использование специализированных алгоритмов (методов) для решения каждой подзадачи.

3. Организацию последовательности действий таким образом, чтобы результат выполнения каждой подзадачи стал входом для следующей.

4. Непрерывное тестирование и улучшение системы на основе полученных результатов.

Для построения эффективной ПСИИ важно правильно выбрать алгоритмы и методы решения каждой подзадачи, чтобы обеспечить наилучшие результаты при минимальных затратах ресурсов.

ПСИИ могут применяться для различных целей, таких как обработка естественного языка, распознавание образов, компьютерное зрение, управление роботами, принятие решений в режиме реального времени и т. д. Результатом успешной работы ПСИИ является повышение эффективности и точности выполнения задач, автоматизация процессов и увеличение производительности.

Разработка ПСИИ может включать следующие шаги:

1. Определение целей. Перед началом работы над последовательной системой ИИ необходимо четко определить цели, которых эта система должна достигнуть.

2. Подбор датасетов. Для обучения последовательной системы ИИ необходимо отобрать соответствующие датасеты с помощью анализа данных, связанных с основными задачами системы.

3. Выделение признаков. После выбора соответствующего датасета требуется определить самые значимые признаки для обучения системы.

4. Обучение модели. На этом этапе данные используются для обучения системы на основе алгоритмов машинного обучения, таких как нейронные сети, метод опорных векторов и решающие деревья.

5. Тестирование. После завершения обучения системы необходимо проверить ее работоспособность на датасетах, которые не были использованы в процессе обучения.

6. Анализ результатов. На этом этапе анализируются результаты тестирования, и, при необходимости, система дорабатывается и оптимизируется.

7. Применение. После завершения всех предыдущих этапов система готова к применению в практических задачах.

Одной из возможных реализаций ПСИИ является рекуррентная архитектура ChatGPT, состоящая из комбинации двух архитектур: Transformer и рекуррентных нейронных сетей (Recurrent Neural Network, RNN).

В отличие от классических рекуррентных нейронных сетей, которые могут столкнуться с проблемой затухающе-

го градиента и забывать далекие зависимости, ChatGPT использует «длинную краткосрочную память» (Long Short-Term Memory, LSTM), которая позволяет модели запоминать длинные зависимости и глубокие контекстные связи между словами.

Таким образом, рекуррентная архитектура ChatGPT сочетает в себе преимущества Transformer и LSTM, что позволяет модели изучать контекст, создавать более точные представления текста и генерировать более интуитивные ответы на основе понимания контекста.

#### РЕКУРРЕНТНЫЕ НЕЙРОННЫЕ СЕТИ

Рекуррентные нейронные сети являются классом искусственных нейронных сетей, которые состоят из повторяющихся блоков и используются для анализа последовательностей данных, где входы могут быть произвольной длины [6, 7]. В отличие от обычных нейронных сетей, которые могут работать только с фиксированным количеством входных данных, RNN могут обрабатывать и моделировать последовательности переменной длины, например текст или звуковые сигналы (рис. 1).

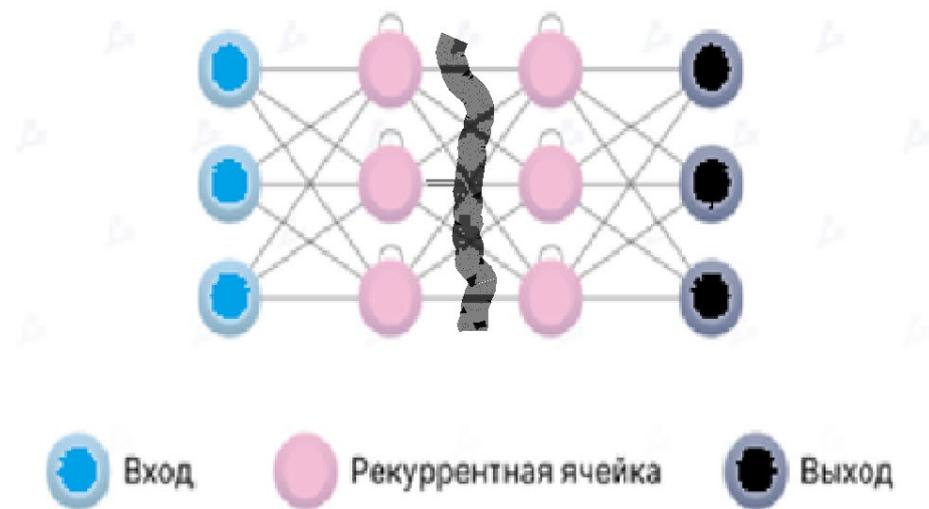


Рис. 1. Рекуррентная нейронная сеть

В отличие от других типов нейронных сетей, RNN обладает возможностью запоминания прошлой информации и использования ее при анализе последующих данных. Они обрабатывают входные данные порциями, сохраняют и передают внутреннее состояние (память) между последовательностями, что позволяет учитывать историю при принятии решений в текущем моменте.

Основными компонентами RNN являются скрытые слои, которые принимают на вход данные от предыдущего шага и вычисляют новое скрытое состояние (память) для передачи на следующий шаг. При этом веса каждого слоя являются общими для всей последовательности данных. Другими словами, каждый слой не просто обрабатывает данные, но и совместно с ранее просмотренными слоями использует их, чтобы создать более осмысленный анализ и вывод.

RNN успешно применяются в разных задачах: в обработке текста (например, в задачах машинного перевода), распознавании речи, создании рекомендательных систем, генерации текстов и т. д. Однако у RNN есть свои проблемы, такие как «затухающий градиент», при котором

влияние более ранних данных на конечный результат исчезает с каждым последующим шагом. Для решения этих и других проблем рекуррентных нейронных сетей были разработаны более совершенные и эффективные модели, такие как LSTM и GRU (Gated Recurrent Unit).

#### ТРАНСФОРМЕР — УНИВЕРСАЛЬНЫЙ ВЫЧИСЛИТЕЛЬНЫЙ МЕХАНИЗМ

Трансформер (Transformer) — это название архитектуры нейросети, предложенной исследователями Google в 2017 году.

Transformer используется для извлечения признаков из текстовых данных и создания представления контекста. Он работает путем многократной обработки последовательности входных данных, называемой энкодером, и создания скрытых представлений для каждого элемента последовательности [8].

Transformer отличается от RNN тем, что он не использует рекуррентность для моделирования последовательностей (рис. 2).

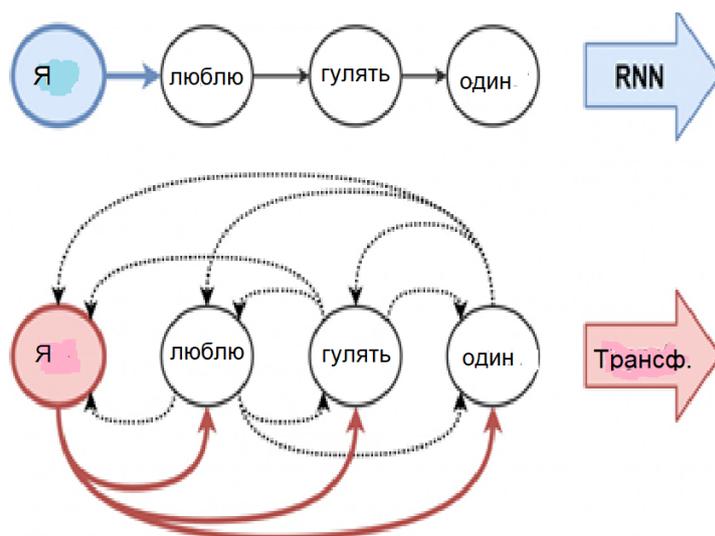


Рис. 2. Отличие Трансформера от рекуррентной нейронной сети

Transformer использует механизм внимания (attention mechanism, attention model) для вычисления весов, которые моделируют отношения между элементами последовательности, и позволяют модели более эффективно использовать контекст. Он состоит из двух основных компонентов — энкодера и декодера.

Универсальный вычислительный механизм Трансформера просто описать: он принимает на вход один набор последовательностей (данных) и выдает на выходе тоже набор последовательностей, но уже другой, преобразованный по некоему алгоритму [9].

Так как текст, картинки и звук можно представить в виде последовательностей чисел, то с помощью Трансформера можно решать практически любые задачи.

Главная особенность Трансформера заключается в его удобстве и гибкости: он состоит из простых модулей-блоков, которые очень легко масштабировать. Если старые, дотрансформерные языковые модели начинали требовать слишком много ресурсов, когда их пытались заставить работать быстро и много слов за один раз, то нейросети-трансформеры справляются с этой задачей гораздо экономнее.

Нейронная сеть Transformer состоит из нескольких блоков, каждый из которых имеет свою структуру. Общая структура нейронной сети может быть описана следующим образом:

1. Входной слой, который принимает входные данные и векторизует их.
2. Предобработка, включающая в себя нормализацию и преобразование данных.
3. Энкодер, который разбивает входные данные на несколько последовательностей (например, токенов) и преобразует каждую последовательность в вектор фиксированной длины.
4. Декодер, который генерирует выходные данные на основе векторов, созданных энкодером.
5. Выходной слой, который генерирует финальный ответ.

Блоки энкодера и декодера в нейронной сети Transformer включают в себя слои многоканальной свертки, слои само-внимания и полносвязные слои. Все эти

блоки работают вместе для достижения высокой точности в задачах обработки текста и генерации языка.

#### АЛГОРИТМ ОБУЧЕНИЯ ТРАНСФОРМЕРА

Алгоритм обучения Трансформера включает в себя следующие шаги [10]:

1. Подготовка данных. На этом шаге данные должны быть представлены в виде последовательности токенов, каждый из которых имеет свой уникальный идентификатор.
2. Построение модели. Трансформер состоит из нескольких слоев, каждый из которых содержит множество механизмов внимания. Модель может быть построена с использованием открытых библиотек машинного обучения, таких как TensorFlow или PyTorch.
3. Инициализация модели. Веса модели инициализируются случайными значениями.
4. Определение функции потерь. Функция потерь определяет, насколько хорошо модель предсказывает правильный ответ. Наиболее распространенными функциями потерь для задач обработки естественного языка являются перекрестная энтропия и среднеквадратичная ошибка.
5. Обучение модели. Модель обучается на тренировочных данных с помощью метода обратного распространения ошибки. Во время обучения модель пытается минимизировать функцию потерь с помощью градиентного спуска.
6. Оценка модели. После обучения модель оценивается на тестовых данных, чтобы определить ее точность и эффективность.
7. Тюнинг модели. Если модель недостаточно эффективна, ее можно доработать, изменяя гиперпараметры.
8. Использование модели. После обучения модель может быть использована для предсказания ответов на новых данных.

Трансформерный блок и токен являются разными элементами в машинном обучении.

Трансформерный блок — это алгоритм, который преобразует входные данные в выходные, обычно с помощью нейронной сети. Он может использоваться, например, для решения задачи классификации или регрессии.

Токен — это отдельный элемент входных данных, который может быть вектором, числом или символом. Он может быть использован для представления слов, символов или фраз в тексте.

Таким образом, трансформенный блок используется для обработки всей входной последовательности, в то время как токены используются для представления отдельных элементов входных данных.

#### ЗАКЛЮЧЕНИЕ

Модель GPT — одна из наиболее эффективных нейронных сетей для вычисления смыслов. Модель обучена на огромных объемах текстовых данных.

Эффект возникновения «осмысленных» текстов в данной модели, очевидно, связан с преодолением известного порога в количестве параметров нейронной сети. Этот эффект требует дальнейших исследований и доказательств.

Отметим некоторые ключевые перспективы развития модели GPT:

1. Увеличение размера модели. Последние версии модели GPT имеют уже достаточно большое количество (миллиарды) параметров, но можно ожидать что они будут еще больше в будущем.

2. Увеличение контекста. Существующие версии модели GPT используют контекст порядка нескольких сотен слов, однако в будущем, вероятно, контекст будет расширяться.

3. Улучшение архитектурных возможностей. Последовательная подача данных между слоями модели типа GPT ограничивает работу сети с длинными последовательностями данных. Будущее развитие модели GPT потребует продвинутых архитектурных решений, например гибридных моделей.

В целом возможности для развития модели GPT в будущем огромны и будут включать в себя техническое совершенствование, изучение дополнительных интересных возможностей и применение в разных задачах.

В дальнейшем ожидается, что развитие модели GPT приведет к более точной генерации текстов, нейронные сети GPT смогут стать более компактными и легкими в работе, произойдет увеличение размера обучаемой модели GPT, развитие модели GPT будет значимо связано с поиском и развитием новых подходов к прикладным задачам.

#### ЛИТЕРАТУРА

1. Шлейермахер, Ф. Д. Э. Академические речи 1829 года / пер. с нем. Е. М. Ананьевой // *Метафизические исследования*. — Санкт-Петербург: Алетейя, 1997. — Вып. 3. — С. 242–260; Вып. 4. — С. 147–163.

2. Худяков, А. А. Теоретическая грамматика английского языка: Учебное пособие. — Москва: Академия, 2005. — 256 с. — (Высшее профессиональное образование).

3. Внимание — все, что вам нужно: как работает attention в нейросетях // *Системный Блок*. — 2020. — 24 июня. URL: <http://sysblok.ru/knowhow/vnimanie-vse-cto-vam-nuzhno-kak-rabotaet-attention-v-nejrosetjah> (дата обращения 16.05.2023).

4. Polyakov, O. M. Linguistic Data Model for Natural Languages and Artificial Intelligence. Part 7. Internal Logic 2 // *Дискурс*. 2022. Т. 8, № 2. С. 98–112.

DOI: 10.32603/2412-8562-2022-8-2-98-112.

5. Блюм, В. С. Диалог о диалоге с виртуальным собеседником / В. С. Блюм, О. М. Поляков // *Актуальные проблемы экономики и управления*. 2022. № 1 (33). С. 81–88.

6. A Novel Connectionist System for Unconstrained Handwriting Recognition / A. Graves, M. Liwicki, S. Fernández, [et al.] // *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2009. Vol. 31, Is. 5. Pp. 855–868. DOI: 10.1109/TPAMI.2008.137.

7. Recurrent Neural Network Tutorial, Part 4 — Implementing a GRU and LSTM RNN with Python and Theano // *Denny's Blog*. — 2015. — 27 October. URL: <http://dennybritz.com/posts/wildml/recurrent-neural-networks-tutorial-part-4> (дата обращения 18.05.2023).

8. Брагин, А. Большие языковые модели как новый инструмент в научной работе // *Хабр*. — 2023. — 03 апреля. URL: <http://habr.com/ru/articles/726606> (дата обращения 22.05.2023).

9. Трансформер // *Викиконспекты*. — Обновлено 04.09.2022.

URL: <http://neerc.ifmo.ru/wiki/index.php?title=Трансформер> (дата обращения 14.10.2023).

10. Алгоритмы обучения нейронной сети: наиболее распространенные варианты // *GeekBrains*. — 2022. — 11 января. URL: <http://gb.ru/blog/algorithmy-obucheniya-nejronnoj-seti> (дата обращения 21.10.2023).

# Semantic Principles of Text Development Artificial Intelligence Systems

PhD V. S. Blyum, PhD V. M. Kosmachev, PhD O. M. Polyakov  
Saint Petersburg State University of Aerospace Instrumentation  
Saint Petersburg, Russia  
vladblum7@gmail.com, kvm@aanet.ru, polyakovoleg@yandex.ru

**Abstract.** The purpose of the article is to determine the theoretical and technological foundations for calculating the semantics of texts. It has been proven that any sentence that has an unconditional right negation does not make sense. The principles of constructing the main modern generators of «meaningful» texts, sequential artificial intelligence systems, and methods of their application are considered. The structure and algorithms of recurrent neural networks and the universal neural network computing mechanism Transformer are considered. The novelty of the approach lies in the fact that the calculation of the semantics of the generated text is based on a large language model.

**Keywords:** semantics, language model, artificial intelligence.

## REFERENCES

1. Schleiermacher F. D. E. Academic speeches of 1829 [Akademicheskie rechi 1829 goda], *Metaphysical Research [Metafizicheskie issledovaniya]*. St. Petersburg, Aletheia Publishing House, 1997. Is. 3, Pp. 242–260; Is. 4, Pp. 147–163.
2. Khudyakov A. A. Theoretical grammar of the English language: Study guide [Teoreticheskaya grammatika angliyskogo yazyka: Uchebnoe posobie]. Moscow, Academia Publishing House, 2005, 256 p.
3. Attention Is All You Need: How Attention Works in Neural Networks [Vnimanie — vse, chto vam nuzhno: kak rabotaet attention v neyrosetyakh], *System Unit [Sistemnyy Blok]*. Published online at June 24, 2020. Available at: <http://sysblok.ru/knowhow/vnimanie-vse-chto-vam-nuzhno-kak-rabotaet-attention-v-nejrosetjah> (accessed 16 May 2023).
4. Polyakov O. M. Linguistic Data Model for Natural Languages and Artificial Intelligence. Part 7. Internal Logic 2, *Discourse [Diskurs]*, 2022, Vol. 8, No. 2, Pp. 98–112. DOI: 10.32603/2412-8562-2022-8-2-98-112.

5. Blyum V. S., Polyakov O. M. Dialogue About Dialogue with a Virtual Speaker [Dialog o dialoge s virtualnym sobesednikom], *Actual Problems of Economics and Management [Aktualnye problemy ekonomiki i upravleniya]*, 2022, No. 1 (33), Pp. 81–88.

6. Graves A., Liwicki, M., Fernández S., et al. A Novel Connectionist System for Unconstrained Handwriting R, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009, Vol. 31, Is. 5, Pp. 855–868. DOI: 10.1109/TPAMI.2008.137.

7. Recurrent Neural Network Tutorial, Part 4 — Implementing a GRU and LSTM RNN with Python and Theano, *Denny's Blog*. Published online at October 27, 2015. Available at: <http://dennybritz.com/posts/wildml/recurrent-neural-networks-tutorial-part-4> (accessed 18 May 2023).

8. Bragin A. Large Language Models as a New Tool in Scientific Work [Bolshie yazykovye modeli kak novyy instrument v nauchnoy rabote], *Habr [Khabr]*. Published online at April 03, 2023. Available at: <http://habr.com/ru/articles/726606> (accessed 22 May 2023).

9. Transformer, *Wikinotes [Vikikonspekty]*. Last update at September 04, 2022. Available at: <http://neerc.ifmo.ru/wiki/index.php?title=Трансформер> (accessed 14 Oct 2023).

10. Neural Network Learning Algorithms: The Most Common Options [Algoritmy obucheniya neyronnoy seti: naibolee rasprostrannyye varianty], *GeekBrains*. Last update at January 11, 2022. Available at: <http://gb.ru/blog/algoritmy-obucheniya-nejronnoj-seti> (accessed 21 Oct 2023).