1 Знакомство с большими языковыми моделями

В этой главе

- ✓ Концепции, лежащие в основе LLM.
- ✓ Архитектура трансформера, на базе которой созданы LLM.
- ✓ План создания LLM с нуля.

Большие языковые модели, такие как ChatGPT от OpenAI, представляют собой модели глубоких нейронных сетей, которые были разработаны за последние несколько лет. Они открыли новую эру в обработке естественного языка (natural language processing, NLP). До появления LLM традиционные методы хорошо подходили для решения задач категоризации, таких как классификация спама в электронной почте и распознавание простых закономерностей, которые можно было обнаружить с помощью разработанных вручную правил или простых моделей машинного обучения. Однако эти методы, как правило, не помогали в случае с языковыми задачами, требующими сложных навыков понимания текста, такими как анализ подробных инструкций, проведение контекстного анализа и создание логически связного текста. Например, предыдущие поколения языковых моделей не могли написать электронное письмо, взяв за основу список ключевых слов, — задача, которая для современных LLM является тривиальной.

Большие языковые модели обладают замечательными способностями понимать, генерировать и интерпретировать человеческий язык. Однако важно уточнить: под словами «языковые модели “понимают”» имеется в виду, что они могут

обрабатывать и генерировать текст так, чтобы он казался связным и контекстуально правильным с точки зрения человека, а не то, что они обладают сознанием или пониманием, подобным человеческому.

Благодаря достижениям в области глубокого обучения, которое является частью машинного обучения и искусственного интеллекта, ориентированного на нейронные сети, большие языковые модели обучаются на огромных объемах текстовых данных. Это масштабное обучение позволяет LLM улавливать более глубокий контекст и лучше (по сравнению с предыдущими подходами) разбираться в тонкостях человеческого языка. В результате модели достигли значительных результатов по широкому спектру задач NLP, таких как перевод текстов, анализ тональности текста, ответы на вопросы и многое другое.

Еще одно важное различие между современными LLM и более ранними моделями NLP заключается в том, что последние обычно разрабатывались для решения конкретных задач: категоризации текста, перевода с одного языка на другой и т. д., и преуспели в своих узких областях применения. LLM же демонстрируют более широкий спектр возможностей в различных задачах NLP.

Успех больших языковых моделей можно объяснить архитектурой трансформера, которая лежит в основе многих LLM, и огромными объемами данных, на которых обучаются модели, что позволяет им улавливать множество лингвистических нюансов, контекстов и закономерностей, которые было бы сложно запрограммировать вручную.

Этот переход к внедрению моделей, основанных на архитектуре трансформера, и использование больших выборок данных для обучения LLM коренным образом изменили обработку естественного языка, предоставив более эффективные инструменты, способные понимать человеческий язык и взаимодействовать с ним.

Основная цель этой книги — предоставить информацию, благодаря которой вы сможете пошагово создать LLM, подобную ChatGPT, используя код на основе архитектуры трансформера. В следующем разделе мы заложим основу для достижения этой цели.

1.1. Что такое LLM

Большая языковая модель — это нейронная сеть, предназначенная для понимания, генерации и обработки текста, похожего на человеческий. Такие модели представляют собой глубокие нейронные сети, обученные на огромных массивах текстовых данных, иногда содержащих значительную часть всего общедоступного текста в Интернете.

Слово «большая» в определении термина относится как к размеру модели по количеству параметров, так и к огромной выборке данных, на которой она обучается. Подобные модели часто содержат десятки или даже сотни миллиардов

параметров, которые представляют собой настраиваемые весовые коэффициенты в сети, во время обучения оптимизируемые для предсказывания следующего слова в последовательности. При данном предсказывании используется присущая языку последовательность, чтобы обучать модели пониманию контекста, структуры и взаимосвязей в тексте. Это очень простая задача, и многие исследователи удивляются, как LLM может решить ее столь эффективно. В следующих главах мы подробно обсудим процедуру подобного обучения и реализуем ее на практике.

Большие языковые модели используют архитектуру, называемую *трансформером*, позволяющую уделять выборочное внимание различным частям входных данных при составлении предсказаний модели. Это делает такие модели особенно эффективными в работе с нюансами и сложностями человеческого языка.

Кроме того, LLM способны генерировать текст, поэтому их часто называют формой генеративного искусственного интеллекта, сокращенно *генеративным ИИ* или *GenAI*. Как показано на рис. 1.1, ИИ охватывает более широкую область создания машин, которые могут выполнять задачи, требующие человеческого интеллекта, такие как понимание языка, распознавание образов и принятие решений, и содержит такие области, как машинное обучение и глубокое обучение.

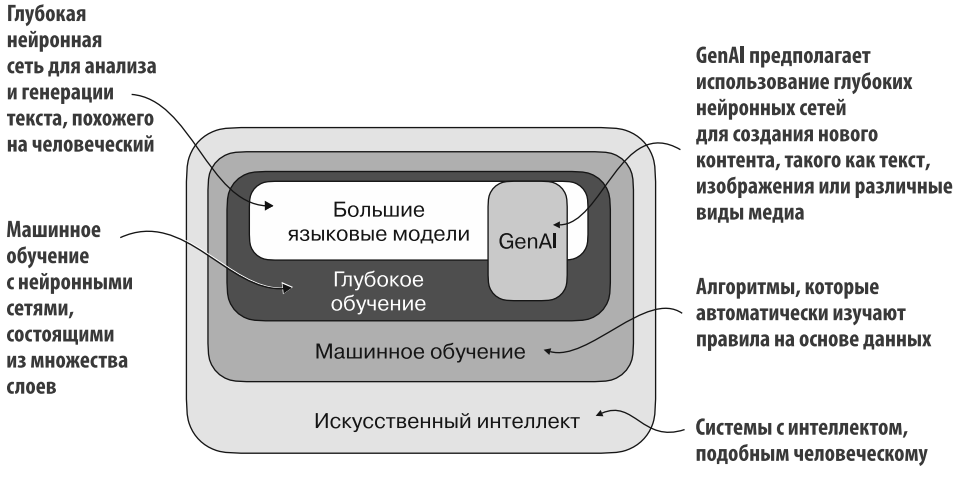

Рис. 1.1. Иерархическое описание взаимосвязи различных областей создания машин

Машинное обучение и глубокое обучение — это области, направленные на реализацию алгоритмов, которые позволяют компьютерам обучаться на основе данных и выполнять задачи, которые обычно требуют человеческого интеллекта.

Алгоритмы, используемые для реализации ИИ, являются предметом изучения в области машинного обучения. В частности, оно включает в себя разработку алгоритмов, которые могут обучаться на основе данных и генерировать предсказания или принимать решения, не нуждаясь в явном программировании.

Глубокое обучение — разновидность машинного; оно фокусируется на использовании нейронных сетей с тремя или более слоями (также называемых глубокими нейронными сетями) для моделирования сложных закономерностей и абстракций в данных.

Машинное и глубокое обучение в настоящее время доминируют в области ИИ, однако в ней применяются и другие подходы — например, использование систем, основанных на правилах, генетических алгоритмов, экспертных систем, нечеткой логики или символьных рассуждений.

В качестве примера практического применения машинного обучения представьте фильтр спама. Модель машинного обучения получает примеры электронных писем, помеченных как спам, и обычных писем. Минимизируя ошибки в своих предсказаниях на обучающей выборке данных, модель учится распознавать закономерности и характеристики, указывающие на спам, что позволяет ей классифицировать новые письма как спам или не спам.

При традиционном машинном обучении люди-эксперты могут вручную извлекать из текста электронного письма такие признаки, как частота определенных слов-триггеров (например, «приз», «выиграть», «бесплатно»), количество восклицательных знаков, использование заглавных букв или наличие подозрительных ссылок. Выборка данных, созданная на основе подобных признаков, будет использоваться для обучения модели.

В отличие от традиционного машинного, глубокое обучение не требует извлечения признаков вручную. Это означает, что экспертам не нужно определять и выбирать наиболее важные признаки для модели глубокого обучения (однако как традиционное машинное, так и глубокое обучение для классификации спама по-прежнему требуют присвоения меток, таких как «спам» или «не спам», которые должны быть присвоены либо экспертом, либо пользователями).

1.2. Применение больших языковых моделей

LLM обладают передовыми возможностями анализа и понимания неструктурированных текстовых данных, поэтому имеют широкий спектр применения в различных областях. Модели используются для машинного перевода, создания текстов (рис. 1.2), анализа тональности текста, обобщения текста и многих других задач. Кроме того, в последнее время LLM стали применяться для создания контента (например, статей), написания художественной литературы и даже

генерации компьютерного кода. Интерфейсы LLM позволяют пользователям и системам ИИ общаться на человеческом языке.

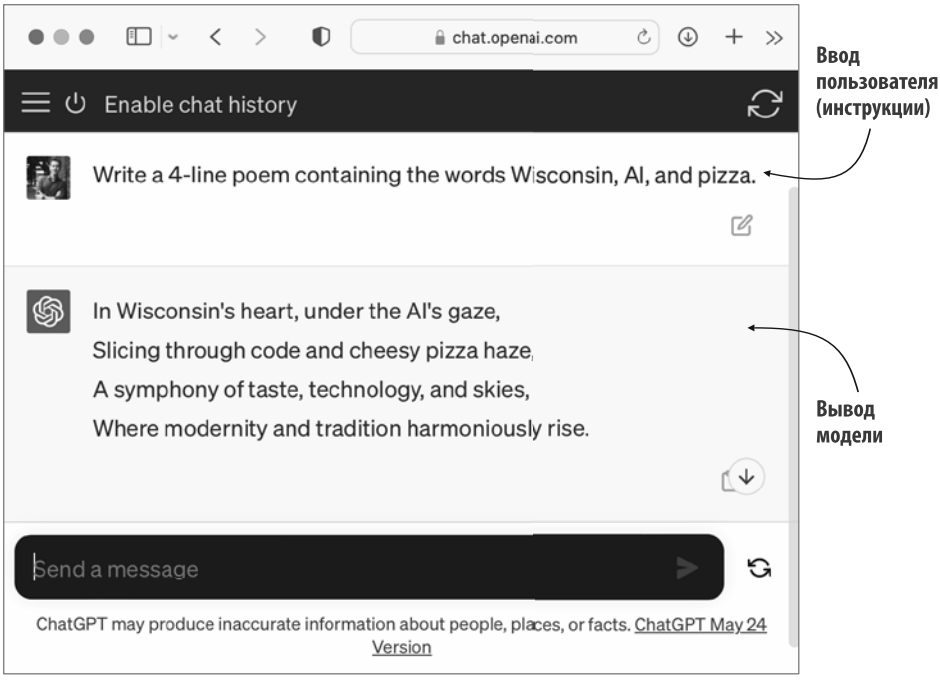

Рис. 1.2. ChatGPT пишет стихотворение по заданию пользователя

Вдобавок LLM являются ядром сложных чат-ботов и виртуальных помощников, таких как ChatGPT от OpenAI или Google Gemini (ранее называвшийся Bard), которые могут отвечать на запросы пользователей и дополнять традиционные поисковые системы, такие как Google Search или Microsoft Bing.

Более того, LLM можно использовать для эффективного извлечения данных из больших объемов текста в специализированных областях, таких как медицина или юриспруденция. Модели помогут отыскать нужные сведения в документах, обобщить длинные отрывки текста и найти ответы на технические вопросы.

Таким образом, LLM незаменимы для автоматизации практически любой задачи, связанной с анализом и генерацией текста. Их возможности практически безграничны, и, по мере того как мы продолжаем внедрять инновации на практике и изучать новые способы использования этих моделей, становится ясно, что у LLM имеется отличный потенциал изменить наше отношение к технологиям, сделав их понятными и более доступными.

1.3. Зачем создавать собственные модели

Написание модели с нуля — отличное упражнение, позволяющее понять ее механику и ограничения. Кроме того, оно дает знания, необходимые для предварительного обучения или тонкой настройки существующих архитектур LLM с открытым исходным кодом, выполняемой пользователем в целях работы с собственными наборами данных или решения задач, связанных с конкретной предметной областью.

> **ПРИМЕЧАНИЕ** Большинство современных LLM реализованы с использованием библиотеки глубокого обучения PyTorch. Мы тоже будем ее применять. Подробное введение в PyTorch можно найти в приложении А.

Исследования показали: пользовательские LLM, созданные для решения конкретных задач или для работы в определенных областях, по производительности могут превосходить универсальные модели, такие как ChatGPT, предназначенные для широкого спектра приложений.

Использование специализированных LLM дает несколько преимуществ, особенно в отношении конфиденциальности данных. Например, компании могут предпочесть не делиться подобными данными со сторонними организациями, такими как OpenAI. Кроме того, небольшие специализированные LLM можно развертывать непосредственно на устройствах клиентов — ноутбуках и смартфонах. Данную возможность в настоящее время изучают такие компании, как Apple.

Локальная реализация может значительно снизить задержки в получении ответа от модели и финансовые расходы, связанные с ее обслуживанием на сервере. Кроме того, специализированные LLM предоставляют разработчикам полную автономность, позволяя контролировать обновления и при необходимости вносить изменения в модель.

Примерами специализированных LLM являются BloombergGPT (предназначенная для финансовой сферы) и LLM, созданные для ответов на вопросы, связанные с медициной. Более подробная информация о подобных моделях представлена в приложении Б.

1.4. Этапы обучения LLM

Процесс обучения модели состоит из двух этапов: предварительного обучения и тонкой настройки (рис. 1.3). Слово «предварительное» в «предварительном обучении» относится к начальному этапу, на котором модель, подобная LLM, обучается на большой и разнообразной выборке данных для широкого понимания языка. Эта предварительно обученная модель затем служит в качестве базового ресурса, который может быть дополнительно усовершенствован с помощью

тонкой настройки. Это процесс, при котором модель обучается на меньшей выборке данных, более специфичной для конкретных задач или областей.

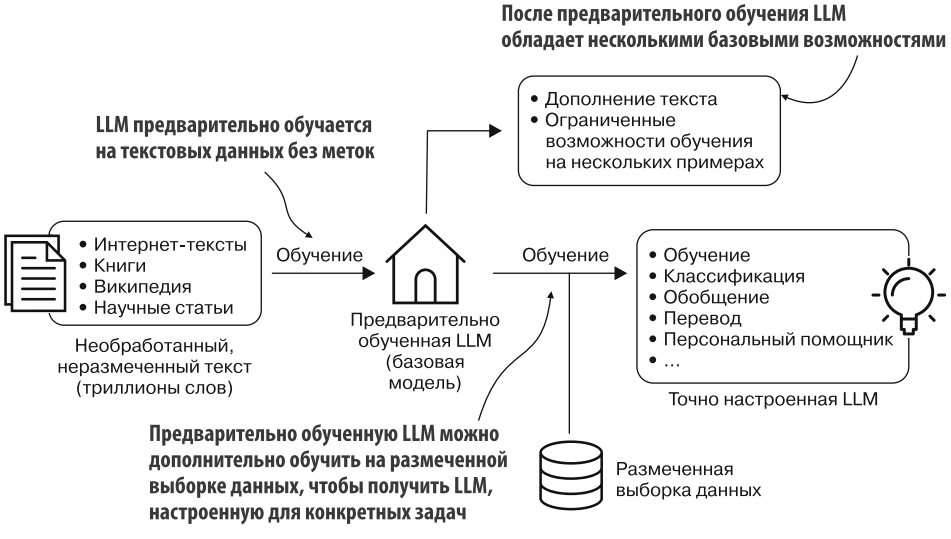

Рис. 1.3. Предварительное обучение LLM включает в себя предсказание следующего слова на больших текстовых выборках данных. Предварительно обученную модель затем можно точно настроить с помощью меньшей по размеру размеченной выборки данных

Разберем эти два этапа более подробно.

Сначала LLM обучается предсказывать следующее слово в тексте, обрабатывая большой массив разнообразных данных, часто именуемых *необработанным* текстом. Это обычный текст без какой-либо информации о метках. (Может применяться фильтрация, например удаление символов форматирования или документов на неизвестных языках.)

> **ПРИМЕЧАНИЕ** Читатели, имеющие опыт в машинном обучении, могут сказать, что информация о разметке данных обычно требуется для традиционных моделей машинного обучения и глубоких нейронных сетей, в которых используется обучение с учителем. Однако для предварительного обучения модели такая разметка не требуется. На данном этапе LLM используют самообучение, при котором модель сама генерирует собственные метки на основе входных данных.

Таким образом, на этапе *предварительного обучения* создается LLM, которую часто называют *базовой*. Типичный пример — GPT-3 (предшественница модели в ChatGPT). Подобная модель способна дополнять текст, то есть завершать наполовину написанное предложение, предоставленное пользователем. Кроме

того, она обладает ограниченными возможностями обучения всего на нескольких примерах вместо больших обучающих данных.

Когда базовая LLM создана, можно переходить ко второму этапу — *тонкой (или точной) настройке модели*. Здесь модель обучается на меньшей по размеру размеченной выборке данных, более характерной для конкретных задач или предметных областей.

Существует две наиболее популярные категории тонкой настройки LLM. При *тонкой настройке по инструкциям* размеченная выборка данных состоит из пар «инструкция — ответ», например фрагмент текста на языке оригинала и его правильный перевод на другой язык. При *тонкой настройке по классификации* размеченная выборка данных состоит из текстов и связанных с ними меток классов, например электронных писем с метками «спам» и «не спам».

Реализацию кода для предварительного обучения LLM, а также специфику тонкой настройки по инструкциям и классификации мы рассмотрим в следующих главах.

1.5. Введение в архитектуру трансформера

Большинство современных LLM основаны на архитектуре *трансформера* — архитектуре глубоких нейронных сетей, предложенной в статье 2017 года *Attention Is All You Need* (https://arxiv.org/abs/1706.03762), написанной группой авторов. Стоит учесть, что первый трансформер был разработан для машинного перевода с английского на немецкий и французский языки. Упрощенная версия архитектуры такого трансформера показана на рис. 1.4.

Архитектура трансформера состоит из двух подмодулей: кодировщика и декодировщика. Модуль кодировщика обрабатывает входной текст и кодирует его в виде набора числовых векторов, которые фиксируют контекстную информацию. Затем модуль декодировщика принимает эти векторы и генерирует выходной текст. Например, в задаче перевода кодировщик преобразует текст на исходном языке в векторы, а декодировщик преобразует их в текст на целевом языке.

На рис. 1.4 показан заключительный этап процесса перевода, на котором декодировщику нужно сгенерировать только последнее слово (Beispiel) на основе исходного входного текста (This is an example) и частично переведенного предложения (Das ist ein). (Если у вас возникли вопросы о том, как входные данные предварительно обрабатываются и кодируются, то не волнуйтесь: мы поговорим об этом в последующих главах, когда будем разбирать пошаговую реализацию модели.)

И кодировщик, и декодировщик состоят из множества слоев, соединенных так называемым *механизмом самовнимания* (на рис. 1.4 не показан, будет описан в главе 3). Это ключевой компонент трансформеров и LLM, который позволяет модели оценивать важность различных слов или токенов в последовательности

по отношению друг к другу. Данный механизм позволяет модели улавливать глубокие зависимости и контекстуальные связи во входных данных, улучшая ее способность генерировать связные и правильные выходные данные. Подробным обсуждением и пошаговой реализацией этого механизма мы займемся в главе 3.

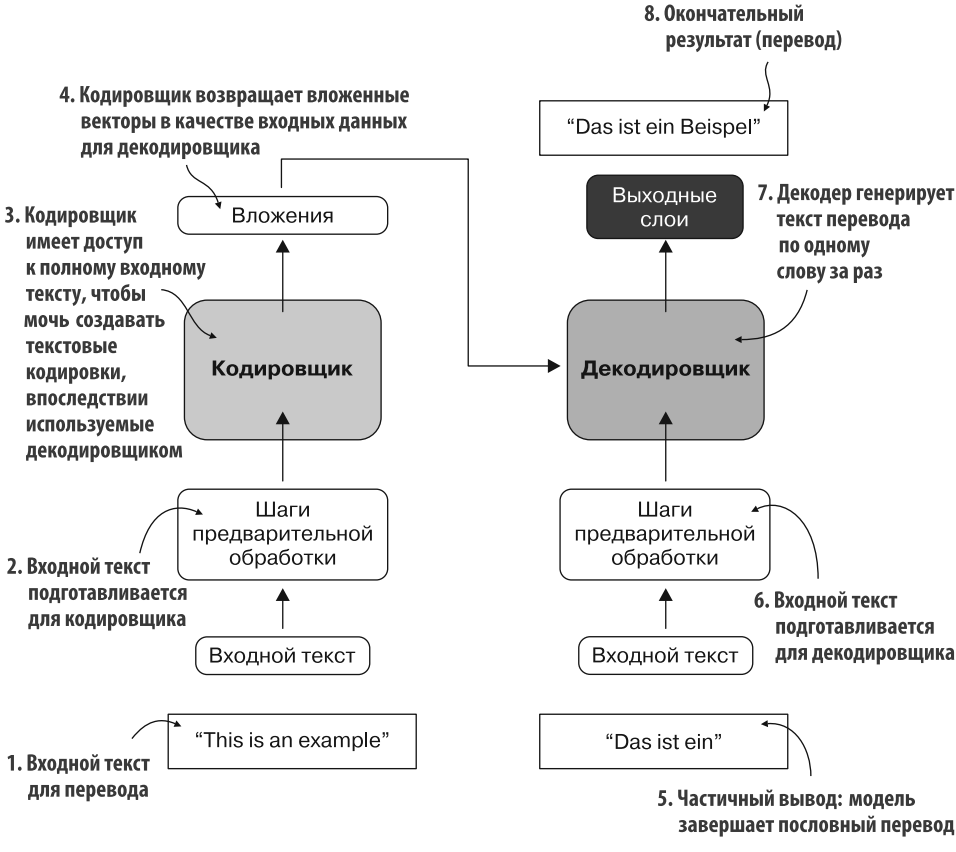

Рис. 1.4. Упрощенное изображение начальной архитектуры трансформера, который представляет собой модель глубокого обучения для языкового перевода

Более поздние варианты архитектуры трансформера, такие как BERT (сокращение от bidirectional encoder representations from transformers) и различные модели GPT (сокращение от generative pretrained transformers), основаны на концепции трансформеров и адаптированы для решения различных задач. Если вам интересно, то в приложении Б вы найдете информацию о других вариантах трансформеров.

BERT, основанный на подмодуле кодировщика первого трансформера, отличается от GPT подходом к обучению. В то время как GPT предназначен для

генеративных задач, BERT и его варианты специализируются на предсказании слов, скрытых маской (рис. 1.5). Эта уникальная стратегия обучения делает BERT эффективным в задачах классификации текста, таких как прогнозирование тональности текста и категоризация документов. На момент написания книги известно, что X (бывший Twitter) использует BERT для обнаружения вредного контента.

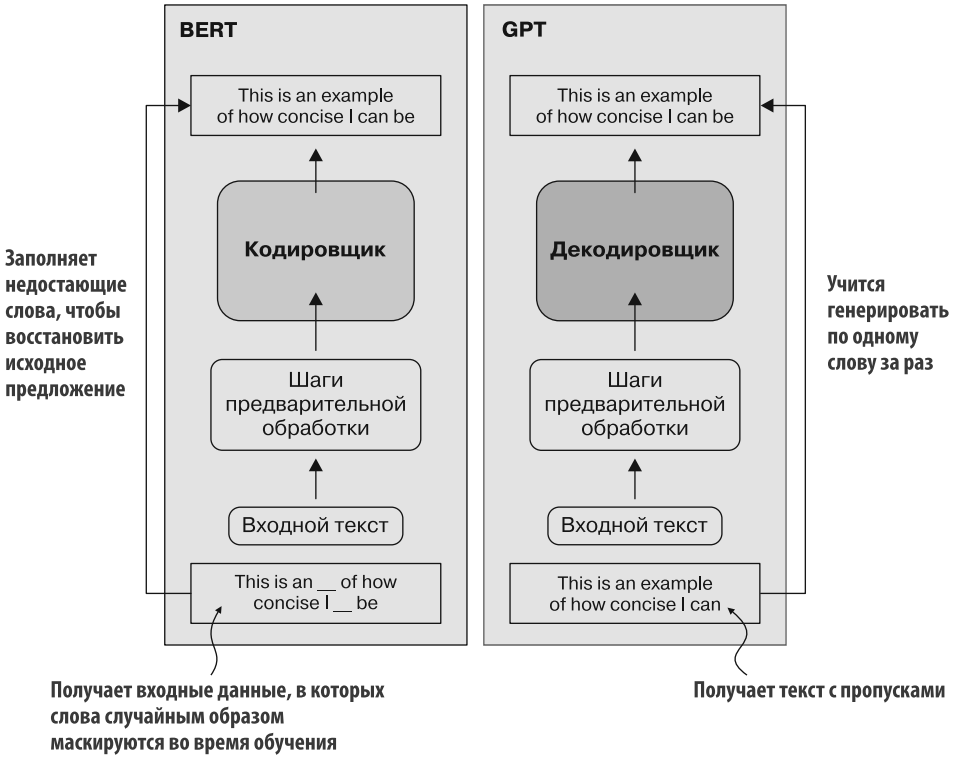

Рис. 1.5. Визуальное представление подмодулей кодировщика и декодировщика для трансформера. Фрагмент кодировщика *(слева)* представляет собой LLM-подобные модели, такие как BERT, которые фокусируются на предсказывании слов с подстановками и в основном используются для таких задач, как классификация текста. Фрагмент декодировщика *(справа)* представляет собой LLM-подобные модели, такие как GPT, предназначенные для генеративных задач и создания связных текстовых последовательностей

В свою очередь, GPT фокусируется на декодирующей части оригинальной архитектуры трансформера и предназначен для задач, требующих создания текстов, — машинного перевода, обобщения текста, написания художественной литературы, компьютерного кода и многого другого.