Автономная некоммерческая организация высшего образования
«Университет Иннополис»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(БАКАЛАВРСКАЯ РАБОТА)
по направлению подготовки
09.03.01 - «Информатика и вычислительная техника»

GRADUATION THESIS
(BACHELOR'S GRADUATION THESIS)
Field of Study
09.03.01 – «Computer Science»

Направленность (профиль) образовательной программы
«Информатика и вычислительная техника»
Area of Specialization / Academic Program Title:
«Computer Science»

Тема / Topic	**Автоматическая генерация дистракторов для текстовых вопросов с множественным выбором / Automatic generation of distractors for multiple choice questions**

Работу выполнил / Thesis is executed by	**Сабиров Руслан Ринатович / Sabirov Ruslan**	подпись / signature
Руководитель выпускной квалификационной работы / Supervisor of Graduation Thesis	**Иванов Владимир Владимирович / Ivanov Vladimir**	подпись / signature

Иннополис, Innopolis, 2021

Оглавление

Аннотация

Тесты - самый популярный способ оценки знаний студентов, однако они требуют много времени на подготовку. Решением проблемы может стать использование программного обеспечения для создания викторин, но уже разработанные решения не могут похвастаться генерацией качественных дистракторов. Мы поставили задачу реализовать метод, который генерировал бы правдоподобные и запутывающиие дистракторы из заданного отрывка, вопроса и правильного ответа. Наша гипотеза заключается в том, что привлечение модели ответа на вопрос и состязательных сетей может помочь в достижении цели.

Следуя подходу GAN, мы построили три модели: генерации дистракторов, модели ответа на вопрос (которая предсказывает, какой вариант является правильным) и дискриминационной модели (которая угадывает, откуда взята выборка: из реальных или сгенерированных данных).

Мы обнаружили, что привлечение дополнительной функции потерь QA улучшает качество случайно отобранных образцов. Также наблюдается значительное увеличение оценок BLEU и ROUGE в 1,7-2,3 раза. Кроме того, мы заметили, что иногда отсутствует корреляция между метриками, рассчитанными автоматически, и качеством, основанным на человеческом восприятии.

Предложенный метод может быть доработан для создания сложных и правдоподобных дистракторов. Наши результаты могут быть использованы для усовершенствования программного обеспечения по автоматизированному созданию викторин.

Глава 1

Введение

В последние годы генерация естественного языка развивается благодаря высокому интересу к автоматизации со стороны научных институтов, промышленности и энтузиастов. Дополнительный толчок к развитию тенденции был вызван разработкой современных архитектур, моделей и выпуском мощного аппаратного обеспечения.

Автоматические методы генерации элементов тестов нашли свое применение в различных областях. Наиболее популярной из них является образование: преподаватели могут дать студентам набор тестовых заданий и оценить их знания по результатам. Аналогичный подход может быть применен в сфере найма: в зависимости от результатов теста компания может решить, стоит ли рассматривать кандидата на работу или нет. Эти методы также могут быть применимы в развлекательной сфере для создания тестов для интеллектуальных игр (таких как «Что? Где? Когда?», «Kahoot!» и т.д.). Эти вопросы могут оценивать знание лексики и различные навыки: понимание текста, подведение итогов, анализ отношения, анализ последствий.

Викторины позволяют объективно проверить знания студентов или кандидатов по следующей схеме. Сначала экзаменатор определяет список вопросов, правильные ответы и дистракторы (неправильные ответы). После этого испытуемый решает задачи теста и отправляет решения в какую-либо систему или непосредственно преподавателю или рекрутеру. Ответы проверяются автоматически или вручную: ответы студента или кандидата сопоставляются с правильными и подсчитывается общий балл. Уровень субъективизма максимально снижен: ответ либо правильный, либо нет, промежуточного состояния нет.

Тем не менее, существуют две основные проблемы. Во-первых, создание тестов требует огромных усилий от экзаменаторов, которые и без этого перегружены: написание и проверка вопросов, ответов и дистракторов (триплет QAD) требуют значительного количества времени и повышенного внимания. Во-вторых, кандидаты и студенты могут обмениваться ответами между собой и искать вопросы в интернете.

Одно из решений этих проблем - сделать вопросы, ответы и дистракторы уникальными и отличающимися для разных экзаменуемых. Предлагаемое решение заключается в автоматической генерации триплета QAD на основе текста. Система будет автоматически генерировать тест, и единственное, что останется сделать экзаменатору, это отредактировать или удалить неподходящие варианты. Экзаменаторам все равно придется редактировать тест, но это гораздо меньше работы, чем без автоматической системы генерации теста.

Приведем небольшой пример. Для текста *"Тесно связанными областями теоретической информатики являются анализ алгоритмов и теория вычислимости. Ключевое различие между анализом алгоритмов*

и теорией вычислительной сложности состоит в том, что первая посвящена анализу количества ресурсов, необходимых конкретному алгоритму для решения задачи" возможен вопрос *"Какая область информатики анализирует требования к ресурсам конкретного алгоритма, изолированного от самого себя в рамках данной задачи?"* с правильным ответом *"анализ алгоритмов"* и дистракторами *"теория вычислительной сложности"*, *"информатика"*, *"алгоритмы и структуры данных"*, *"теоретическая информатика"*. Как видно, некоторые дистракторы взяты не из текста, в целом они могут быть сгенерированы на основе фонетики, онтологии, сходства эмбеддингов (векторных представлений) или других признаков.

В данной работе мы рассматриваем автоматическую генерацию вопросов с множественным выбором (MCQ), которые состоят из самого вопроса и набора вариантов (правильный ответ и набор дистракторов, неправильные варианты). Вопрос основывается на предоставленном абзаце текста, знании реального мира и понимании языка.

Значительное количество исследований было посвящено задачам генерации вопросов (QG) и ответов на вопросы (QA). Существуют также научные работы, связанные с генерацией дистракторов (DG), но эта область наименее изучена, и в ней можно провести множество экспериментов, поэтому наша работа сосредоточена на DG, хотя мы и рассматриваем и изучаем методы для решения задач QG и QA, так как эти задачи похожи и сильно зависят друг от друга.

Задача генерации дистракторов может быть сформулирована как задача seq2seq и для ее решения могут быть использованы соответствующие методы. Исследователи пытались решить задачу генерации дистракторов,

используя различные подходы, основанные на правилах и нейросетях. Первые используют базы знаний или метрики сходства. Вторые достигают лучших результатов и в основном используют модели LSTM, GAN, VAE и Трансформеры.

Основываясь на различных предыдущих работах, мы определили список критериев, по которым дистракторы должны считаться высококачественными. Однако, даже архитектуры SOTA[1] не отвечают всем этим критериям.

В некоторых работах рассматривается использование GANs[2] или моделей QA для обучения или оценки QG и DG. Тем не менее отдельное и совместное использование QA-моделей или GANs для генерации дистракторов не получило должного внимания. Интуиция подсказывает, что привлечение состязательных методов может значительно повысить качество генерируемых дистракторов и элементов теста в целом. Дискриминатор из GAN обеспечивает более правдоподобную генерацию дистракторов, в то время как QA-модель делает генерируемые дистракторы достаточно сложными.

В литературе в основном используется человеческая оценка или автоматические метрики. Хотя первая является наиболее важной и точной, она подразумевает участие человека, что отнимает время и деньги. Второй вариант быстрый и дешевый, но он не полностью отражает человеческую оценку. Компромиссным решением может стать оценка на основе адверсариального подхода: она призвана имитировать человеческое восприятие, при этом она достаточно быстрая и доступная.

После проведения обзора литературы мы поставили следующие ис-

[1] Состояние искусства

[2] Generative Adversarial Networks

следовательские вопросы:

- Как применить совместное обучение генератора и модели QA для генерации высококачественных дистракторов?
- Как использовать подход GAN для повышения качества генерируемых дистракторов?

Ключевыми вкладами работы являются:

1. Мы собрали список критериев для качественного дистрактора.
2. Мы изучили влияние совместного обучения моделей DG и QA на качество генерации дистракторов.

Данная дипломная работа имеет следующую структуру. Она начинается с Обзора литературы (2, в которой мы рассказываем об основных методах для генарации дистракторов и смежных задач. Раздел Методология (3) рассказывает о предлагаемой архитектуре для генерации дистракторов. Следующий раздел, Реализация (4), знакомит с деталями реализации (используемое аппаратное и программное обеспечение). Следующая глава, Результаты и анализ (5), представляет результаты экспериментов, интерпретирует и обсуждает их. И, наконец, раздел Заключение (6) подводит итоги исследования и определяет возможные направления дальнейшей работы.

Глава 2

Обзор литературы

В этой главе содержится необходимая информация для понимания проблемы генерации вопросов и отвлекающих факторов. Понимание существующих подходов очень важно при проведении научных исследований, поскольку оно помогает определить преимущества и ограничения существующих решений, чтобы избежать ошибок и извлечь пользу из предыдущего передового опыта.

Глава организована следующим образом:

- Раздел 2.1 объясняет основную терминологию, используемую в литературе;
- Раздел 2.2 вводит основы обучения по принципу seq2seq и описывает наиболее распространенные модели;
- Раздел 2.3 рассказывает о наиболее часто применяемых генеративных моделях;
- Раздел 2.4 является обзором методов генерации вопросов;

- Раздел 2.5 объясняет подходы, используемые для генерации дистракторов;
- Раздел 2.6 объясняет и обсуждает работу, которая стала основой для данное исследование.
- Раздел 2.7 перечисляет и классифицирует существующие датасеты;
- Раздел 2.8 описывает методы, используемые для оценки дистракторов;
- Раздел 2.9 подводит итоги главы и знакомит с вопросами исследования.

2.1 Терминология

В этой главе мы представим наиболее распространенные определения, которые используются в литературе и которых мы будем придерживаться в данной работе.

2.1.1 Quiz item

Quiz item (предмет викторины) - это общий термин для обозначения тройки из вопроса, ответа и набора дистракторов, используемых в тестах или викторинах ([1], [2]). Обычно он состоит из следующих компонентов:

- **Вопрос** — это вопросительное выражение (не всегда в форме классического вопроса с вопросительным знаком).

- **Ответ** (также: ключ, правильный ответ) — это вариант, который правильно отвечает на вопрос.
- **Дистракторы** (или альтернативные ответы, или неправильные ответы, или отвлекающие факторы) — это неправильные варианты.
- **Параграф** (также: текст, абзац) — стимулирующий материал, на основе которых испытуемые отвечают на вопрос.

Для целей данного исследования мы будем использовать термины, выделенные жирным шрифтом.

2.1.2 Типы вопросов

Типы вопросов в теории тестов обычно классифицируются следующим образом (например, см. в [1]):

- **Открытый вопрос** (также: открытый вопрос) — вопрос, заданный в естественной форме, без отвлекающих элементов. Например, "Который час?".
- **Вопрос с пропуском** (fill-the-gap) — вопрос, который требует от тестируемых вставить пропущенное слово или слова в предложение. Дистракторы либо предоставляются (закрытый подтип), либо нет (открытый подтип). Пример открытого подтипа: "Лондон является столицей _____". Предоставление списка вариантов ["США "Великобритания "Россия "Италия"] делает вопрос закрытым.
- **Вопрос с множественным выбором** (MCQ - multiple choice question) — вопрос, заданный в естественной форме с предоставлен-

ным списком возможных вариантов ответа, который состоит из Ответа и набора Отвлекающих факторов. Например, вопрос: "Сколько бит в байте?"и возможные варианты ответа: 2, 4, 8, 1024.

2.1.3 Задачи

В литературе (например, [3], [4], [5]) выделяют три основные проблемы, связанные с викторинами:

- **Генерирование вопросов** (QG - question generation) — задача создания правильного вопроса на основе некоторого текста, изображения или базы знаний.
- **Генерирование дистракторов** (DG - distractor generation) — задача генерирования правильного дистрактора, при заданных вопросе и параграфе (необязательно).
- **Системы ответов на вопросы** (QA - question answeing) — задача дать правильный ответ на вопрос, учитывая сам вопрос и варианты (необязательно).

Это широко используемая структура, которая может применяться как для текстовой и визуальной модальностей, так и для их комбинации. Несмотря на различия в модальностях, методы, используемые для визуальных модальностей, могут быть применены к текстовым, и наоборот.

Эта исследовательская работа была сосредоточена на простых текстовых вопросах викторины, в частности, на улучшении генерации дистракторов. Мы будем использовать подход глубокого обучения, поскольку он является более общим и имеет больше применений, как мы увидим позже.

2.2 Основные сведения о seq2seq

Поскольку QG и DG являются задачами обработки естественного языка (NLP), очень важно понимать наиболее распространенные модели и методы NLP, которые также применимы в задачах QG и DG. Применяемые модели глубокого обучения можно классифицировать как seq2seq и генеративные модели, они будут рассмотрены в этой и следующей главе соответственно.

Схема обучения "последовательность-последовательность"(seq2seq) принимает входную последовательность переменной длины в качестве входа и производит последовательность переменной длины в качестве выхода. Обучение Seq2seq нашло свое применение в различных областях: Нейронный машинный перевод, распознавание речи, резюмирование текста, создание подписей к изображениям и многие другие.

2.2.1 RNN Encoder-decoder

Энкодер-декодер - одна из самых популярных схем преобразования последовательностей в последовательности. Нейронная сеть RNN Encoder-decoder была впервые представлена в [6] автором Cho и др. RNN расшифровывается как Recurrent neural networks (рекуррентная нейронная сеть, РНС), RNN называются рекуррентными, потому что они выполняют одну и ту же задачу для каждого элемента последовательности таким образом, что выход модели зависит от предыдущих вычислений и предыдущих входов. Это означает, что выходная лексема зависит от предыдущих входов и выходов. Модель принимает исходную последовательность $x = (x_1, x_2, ..., x_{T_x})$ в качестве входа и производит целевую последователь-

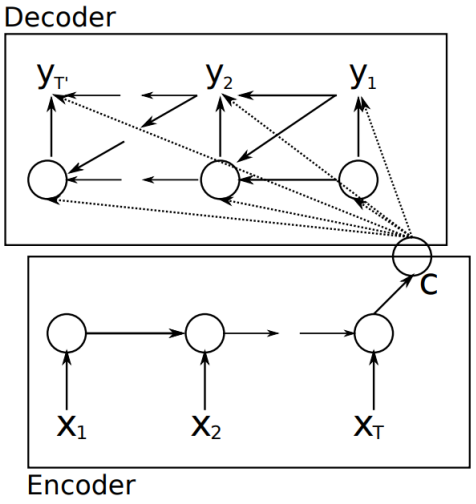


Рис. 2.1: RNN Архитектура энкодера-декодера из [6]

ность $y = (y_1, y_2, ..., y_{T_y})$ в качестве выхода. Модель состоит из двух основных компонентов (оба реализованы в виде RNN): энкодера, который обучается данным и создает скрытый вектор контекста, и декодера, который генерирует последовательность в целевой области на основе скрытого вектора. Архитектура модели представлена на Рис. 2.1.

Скрытое состояние кодирующей РНС обучается путем обработки каждого токена из входной последовательности следующим образом: $h_t = f(h_{t-1}, x_t)$, где h_t - текущее скрытое состояние, h_{t-1} - скрытое состояние от предыдущей метки времени, x_t - текущий входной токен, а f - некоторая нелинейная функция. На основе изученных скрытых состояний h_t формируется контекстный вектор c: $c = q(h_1, h_2, ..., h_{T_x})$, где q - нелинейная функция. Контекстный вектор рассматривается как сжатие или обобщение входной последовательности x.

Целью декодера является предсказание следующего токена y_t с учетом скрытого состояния h_t. Скрытое состояние декодера h_t и условная веро-

ятность генерации следующего токена вычисляются следующим образом:

$$h_t = f(h_{t-1}, y_{t-1}, c)$$

$$p(y_t \mid y_{t-1}, y_{t-2}, ..., y_1, c) = g(y_{t-1}, c, h_t),$$

где f, g - нелинейные функции активации.

Авторы показали, что модель способна улавливать лингвистические закономерности и предлагать хорошо сформированные целевые фразы в задаче машинного перевода.

2.2.2 LSTM

LSTM (Long Short Term Memory) является одной из самых популярных РНС. Модель LSTM была впервые представлена в [7] авторами Hochreiter и Schmidhuber, а затем модифицирована, расширена и применена многими исследователями. Авторы показали, что для длинных последовательностей у RNN есть две проблемы: градиенты либо раздуваются (explode), либо исчезают (vanish).

Рекуррентная нейронная сеть имеет форму повторяющейся цепочки ячеек. Повторяющаяся ячейка в обычной РНС имеет только один слой, в то время как LSTM имеет четыре слоя (см Рис. 2.2). LSTM состоит из трех компонентов, известных как ворота: входные ворота (обновляется ячейка), ворота забывания (память устанавливается в 0), выходные ворота (текущая информация отображается в будущем).

Слой Forget Gate (Ворота забывания) решает, какая информация может быть отброшена из состояния клетки. Он учитывает предыдущее скрытое состояние h_{t-1} и текущий вход x_t и возвращает значение от 0 до 1

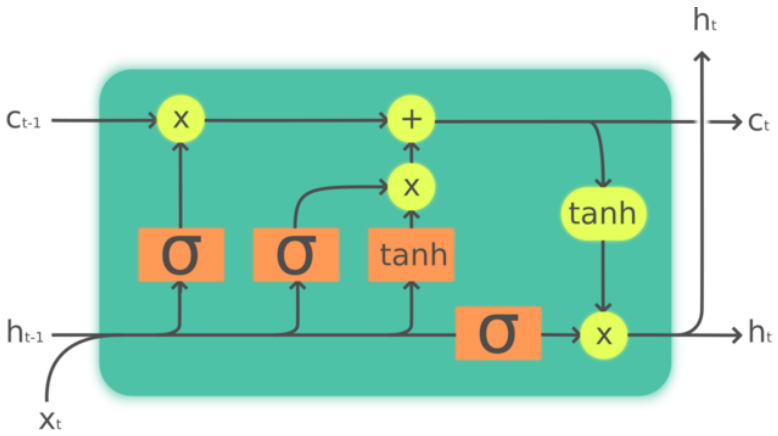


Рис. 2.2: A LSTM cell from [8] (лучше смотреть в цвете). Оранжевые прямоугольники представляют слои, желтые эллипсы - точечные операции, а стрелки, расходящиеся из одной точки - операции копирования

для каждого числа в состоянии клетки: 0 означает "полностью сохранить 1 означает "полностью отбросить".

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

Следующий шаг - решить, какая информация будет храниться в состоянии клетки. Сначала сигмоидальный гейт входного слоя определяет, какие значения должны быть обновлены. Затем tanh-слой строит вектор новых значений-кандидатов $\tilde{C}_t$, которые могут быть добавлены в состояние клетки.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right)$$

$$\tilde{C}_t = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right)$$

После этого нужно заменить старое состояние клетки $\tilde{C}_{t-1}$ на новое $\tilde{C}_t$. Умножаем старое состояние $\tilde{C}_{t-1}$ на f_t, забывая то, что решили забыть. После этого мы добавляем $i_t * \tilde{C}_t$, значения кандидатов, умноженные на i_t,

балл обновления:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Наконец, мы должны решить, какую информацию мы хотим получить в качестве выходных данных. Выходные значения зависят от ячейки, к которой применяются некоторые фильтры. Сначала мы применяем сигмоидный слой, который решает, какая информация из состояния клетки будет выведена. Затем значения состояния клетки пропускаются через tanh-слой для получения значений в диапазоне $[-1;\ 1]$ и умножаются на выход сигмоидного слоя, что позволяет получить на выходе только необходимую информацию.

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Наиболее выдающиеся результаты среди РНС получены с помощью LSTM, для большинства задач они работают намного лучше, чем простые РНС. Возникает естественный вопрос: каков следующий большой шаг? Наиболее распространено мнение, что следующим прорывом станет механизм внимания.

2.2.3 Трансформеры

Механизм внимания

Механизм внимания (attention mechanism) был представлен Vaswani и др. в [9]. Авторы утверждают, что стандартная архитектура ED (encoder-decoder) значительно хуже справляется с переводом длинных предложе-

ний. Они также отмечают, что модель LSTM работает последовательно (каждый узел нуждается в состоянии предыдущего узла для создания своего собственного состояния) и поэтому не использует возможности современных графических процессоров (предназначенных для параллельных вычислений).

Трансформер

Модель трансформера ([10]) - это версия архитектуры энкодера-декодера, использующая механизм внимания. Он был разработан для имитации поведения человека при переводе предложений: разделить предложение на несколько значимых частей, перевести их по очереди и объединить перевод с имеющимся результатом. Идея механизма внимания основана на контексте, какой район считается важным для конкретного узла. Для этого вычисляются веса внимания, то есть сколько внимания мы уделяем i-му входному слову. В отличие от модели RNN, полная входная последовательность проходит через модель одновременно.

Архитектура модели трансформера показана на Рис. 2.3. Левая часть модели состоит из следующих частей:

- Входное встраивание (Input Embedding) отображает входное слово на вектор из пространства встраивания.
- Позиционное кодирование (Positional Encoding) - это вектор, который придает входному встраиванию контекст, основанный на положении слова в предложении.
- Энкодер:

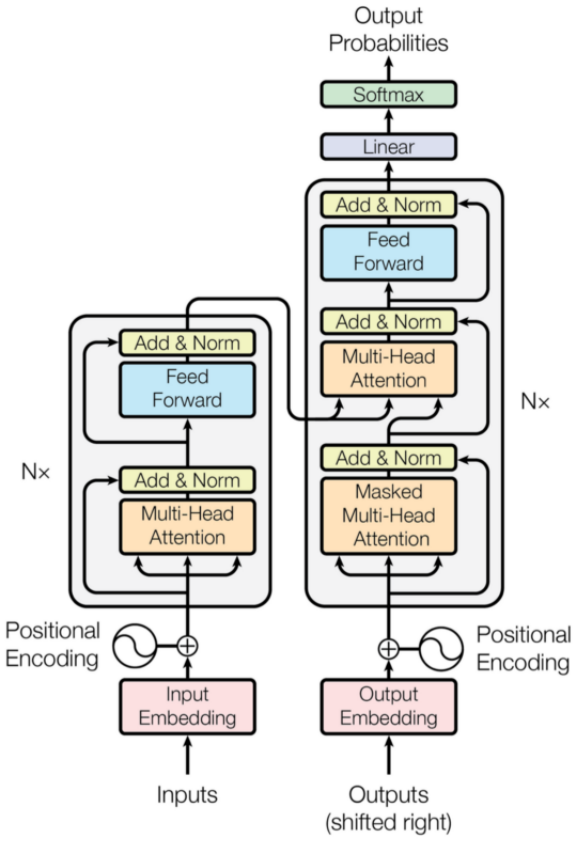


Рис. 2.3: The Transformer — модель архитектуры от [10] (лучше видно в цвете)

— Многоголовое внимание (Multi-head attention) определяет, на какой части входного сигнала должна сосредоточиться модель, и генерирует вектор значений внимания для каждого входного слова.

— Сумма & Нормализация (Sum & Norm) суммирует все значения внимания, присвоенные конкретному слову, а затем нормализует их (используется для решения проблемы, когда внимание фокусируется на самом слове, что является бесполезной информацией) .

— Нейронная сеть с прямой связью (Feed-forward net) преобразует результаты внимания в форму, более пригодную для использования следующим компонентом.

Компоненты, расположенные в правой части рисунка представления модели, работают с выходными словами (описание для повторяющихся блоков опущено):

- Выходное встраивание, или Output Embedding (например, переведенное предложение из обучающей базы данных).
- Блок декодера:
  - Маскированное многоголовое внимание (Masked Multi-Head Attention): какие слова определяют контекст для каких слов. Оно называется маскированным, потому что при генерировании следующего слова с целевого языка используются все переводы ранее увиденных целевых слов .
  - Многоголовое внимание (внимание энкодера-декодера): сопоставляет векторы входного и выходного внимания.
- Линейный слой (feed-forward layer) расширяет размеры до количества слов на целевом языке.
- Слой Softmax преобразует вектор в распределение вероятности.
- Выходное слово - это слово, которое имеет максимальную вероятность.

Трансформеры способны изучать дальние связи между текстовыми лексемами. Основным недостатком трансформеров является скорость обучения и вывода, однако они поддаются распараллеливанию, что позволяет исследователям использовать возможности современных графических процессоров.

BERT

Механизм внимания стал популярен в NLP благодаря модели BERT (Bidirectional Encoder Representations from Transformers) [11], которая использует кодировщики из трансформеров.

Модель BERT была обучена на большом наборе данных для понимания языка и контекста независимо от задачи. Предварительно обученная модель BERT может быть тонко настроена с помощью всего одного дополнительного выходного слоя для решения конкретной задачи (например, ответа на вопрос, автоматического перевода, анализа текста и т.д.). Общие процедуры предварительного обучения и тонкой настройки BERT показаны на рисунке Рис. 2.4.

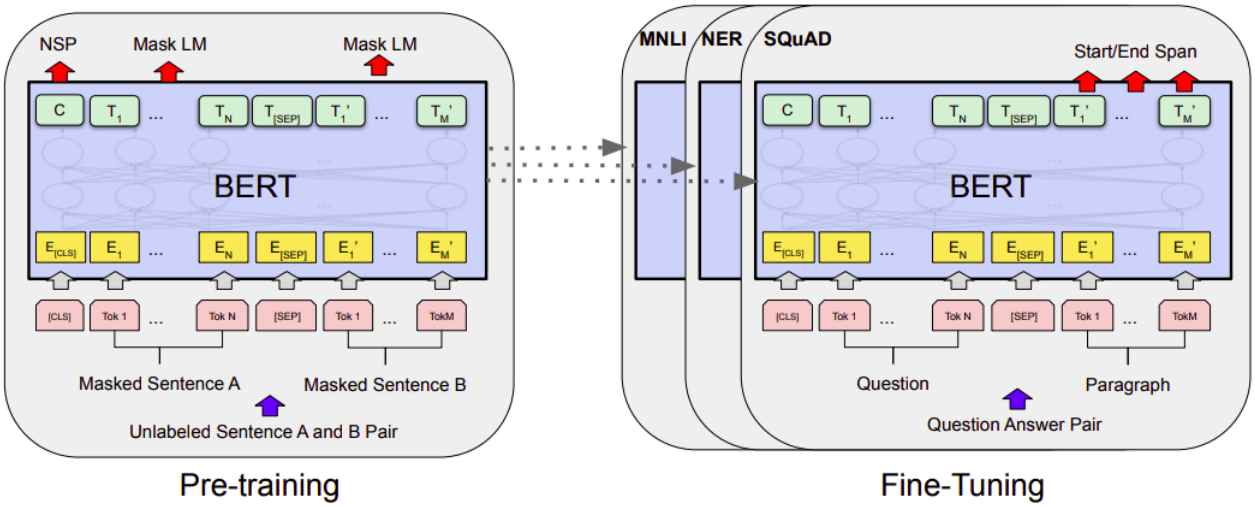


Рис. 2.4: Общие процедуры предварительной подготовки и тонкой настройки для BERT от [11]

Модель предварительно обучена одновременно на двух задачах без надзора: Masked language modeling (MLM), Next sentence prediction (NSP). Задача MLM похожа на вопросы с заполнением пробелов, то есть некоторые слова в предложении заменяются специальным маркером $[MASK]$,

и модель обучается предсказывать замаскированные слова. Что касается задачи NSP, модель получает на вход два предложения и делает вывод о том, следует ли второе предложение за первым. Обучаясь на этих двух задачах вместе, BERT получает хорошее понимание языка. Задачи решаются одновременно, т.е. на вход модели подаются два предложения, разделенные лексемой $[SEP]$, в которых некоторые лексемы (слова) заменены на $[MASK]$. На первом этапе входные слова преобразуются в эмбеддинги с помощью предварительно обученной модели эмбеддинга. Поскольку одновременно решаются две задачи, то и выход модели состоит из двух частей: C, бинарная оценка классификации для задачи NSP и набор вкраплений для маскированных слов обоих предложений $T_1, T_2, ..., T_N, T'_1, T'_2, ..., T'_N$.

Адаптация BERT для решения конкретной задачи состоит из следующих частей: тонкая настройка предварительно обученных слоев и обучение дополнительного слоя для получения выходных данных, специфичных для конкретной задачи. Поскольку с нуля обучаются только параметры выходного слоя, а параметры других слоев лишь слегка изменяются, обучение проходит быстро. Devlin и др. описывает конкретное применение BERT для решения задачи ответа на вопрос на наборе данных SQuAD. Вопрос и абзац передаются через модель в качестве входных данных, и модель выдает положение начальных и конечных слов ответа (предполагается, что ответ содержится в абзаце).

Авторы утверждали, что BERT превосходит существующие решения в проблеме ответов на вопросы.

2.3 Основные сведения о генеративных моделях

Второй тип моделей NN, которые в основном применяются в задачах QG и DG, - это генеративные модели. Генеративные модели, как следует из их названия, генерируют примеры входных данных путем изучения их статистики.

2.3.1 Вариационные автоэнкодеры

Автоэнкодер

Автоэнкодер, Autoencoder ([12]) - это особый тип глубокой нейронной сети, которая обучается для изучения скрытого представления данных путем проецирования входных данных в кодированное пространство и последующего восстановления (декодирования) кодированного представления на выходе. Более конкретно, автоэнкодер решает следующую задачу: при входных данных x $\mathbb{R}^n$, кодер - это функция $f(\cdot) : \mathbb{R}^n \to \mathbb{R}^d$, декодер - функция $g(\cdot) : \mathbb{R}^d \to \mathbb{R}^n$, где $\hat{x} = g(f(x))$ - восстановленная версия x. Целью процесса обучения является минимизация потерь при реконструкции $\mathcal{L}(x, \hat{x})$. Пример AE показан в Рис. 2.5.

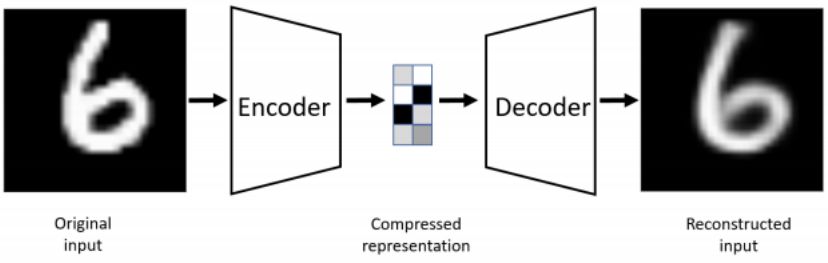


Рис. 2.5: Пример автоэнкодера из [13]

Автоэнкодеры применимы для решения многих задач, не требующих контроля, включая, но не ограничиваясь, сжатие данных, уменьшение размерности, восстановление поврежденных данных и кластеризацию латентного пространства. Существует множество типов автоэнкодеров, наиболее популярными являются следующие ([13]):

- Undercomplete Автокодировщики имеют латентную размерность, которая меньше входной размерности;
- Overcomplete Автокодировщики имеют латентную размерность, которая больше, чем входная размерность;
- Регуляризованные автоэнкодеры используют функцию потерь, которая заставляет модель обладать другими свойствами, кроме способности копировать (реконструировать) входные данные:
  - Разреженные автокодировщики штрафуют активацию скрытых слоев так, что только несколько слоев активируются, когда образец подается в сеть;
  - Деноизирующие автокодировщики учат наиболее надежные характеристики и отбрасывают шум;
- Вариационные автокодировщики обобщают данные через вероятностное распределение.

Вариационный автоэнкодер

Сжатое представление в VAE - это распределение вероятностей. В VAE модель кодера иногда называют моделью распознавания, а модель

декодера - генеративной моделью. Для любой выборки скрытых распределений мы ожидаем, что наша модель декодера сможет точно восстановить входные данные. Целью VAE является нахождение распределения $q_\phi(z \mid x)$ некоторых латентных переменных, которое мы можем выбрать из $z \sim q_\phi(z \mid x)$ для генерации новых выборок $\tilde{x}$ из $p_\theta(x \mid z)$, где ϕ и θ обозначают параметры кодера и декодера соответственно. Обзор модели показан на Рис. 2.6.

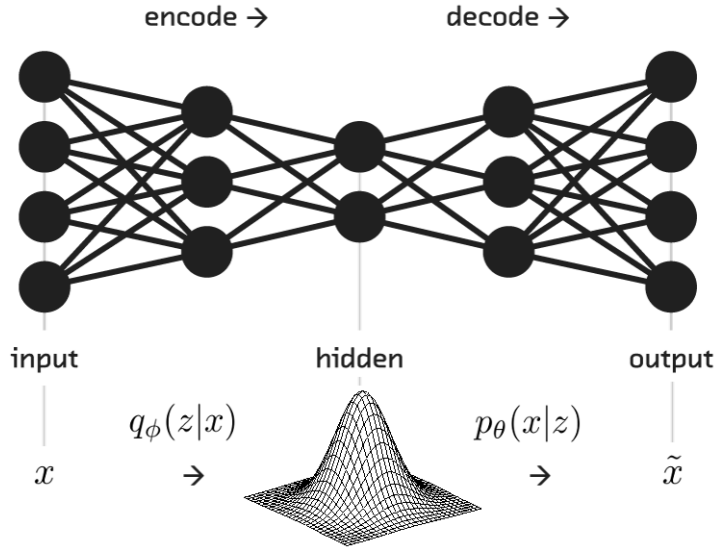


Рис. 2.6: Архитектура VAE от [14]

Потери VAR определяются по следующей формуле:

$$\mathcal{L}(\phi, \theta, x) = D_{\mathrm{KL}}\left(q_\phi(z \mid x) \| p_\theta(z)\right) - \mathbb{E}_{q_\phi(z|x)}\left(\log p_\theta(x \mid z)\right),$$

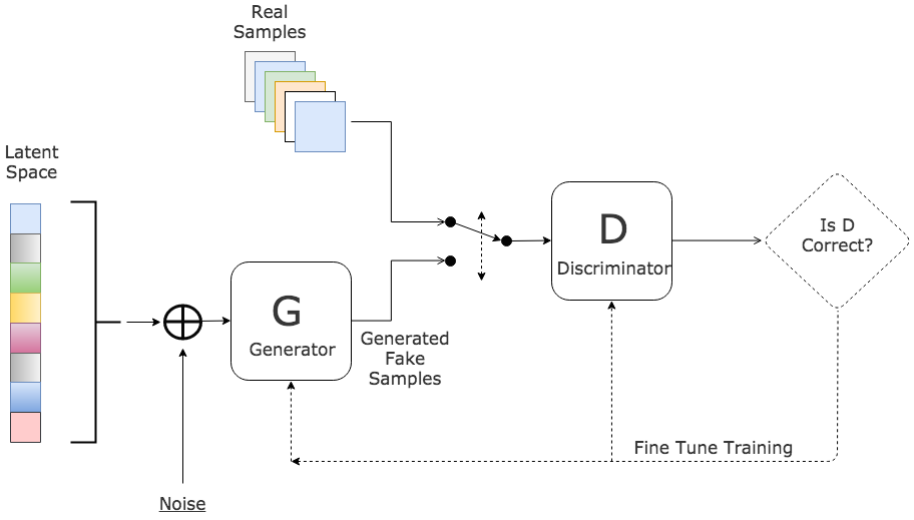
где D_{KL} означает дивергенцию Куллбэка-Лейблера. $p(z)$ может быть любой функцией распределения вероятности, например, гауссовским распределением.

Модель VAE элегантно сжимает данные в простое распределение ве-

роятности и дает очень хорошие результаты. С другой стороны, изображения, созданные с помощью VAE, как правило, размыты.

2.3.2 Генеративно-состязательная сеть

Генеративные состязательные сети (GAN) были впервые представлены Goodfellow, Pouget-Abadie, Mirza и др. в [15]. Обучение GAN представляет собой игру двух игроков: модель генератора G обманывает дискриминатор, генерируя образцы, похожие на настоящие, а модель дискриминатора D различает настоящие и поддельные образцы. В оригинале [15] образцы - это изображения, но это может быть любой тип данных, который может быть сгенерирован G и дискриминирован D. Обзор фреймворка представлен в Рис. 2.7.



.

Рис. 2.7: Структура генеративной адверсариальной сети (GAN) от [16]

Целью модели G является изучение распределения вероятностей $p_{\text{model}}(x)$, которое аппроксимирует распределение вероятностей реальных выборок $p_{\text{data}}(x)$. Модели G и D могут быть определены формулами:

$$\mathrm{G} : z \mapsto G\left(z;\ \theta^{(G)}\right)$$

.

$$\mathrm{D} : x \mapsto D\left(x;\ \theta^{(D)}\right)$$

.

$D(x)$ представляет собой вероятность того, что x получено из реальных данных, а не из p_{model}. z - случайный шум, используемый для генерации выборки. Локальный минимум обеих потерь можно определить как равновесие Нэша. Равновесие Нэша достигается, когда Генератор производит изображения, которые Дискриминатор не может отвергнуть (точность дискриминации составляет 0,5). Дискриминатор D обучается для максимизации вероятности присвоения правильной метки как обучающим примерам (т.е. x), так и образцам из G (т.е. $G(z)$). В то же время G обучается минимизировать $log(1-D(G(z)))$. Тогда функция потерь определяется следующим образом:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\text{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Модель GAN нашла свое применение во многих областях и производит высококачественные современные образцы. Тем не менее, она довольно сложна, а обучение нестабильно.

2.4 Генерация вопросов

Несмотря на то, что наша работа посвящена задаче генерации дистракторов, стоит также обратить внимание на существующие решения по

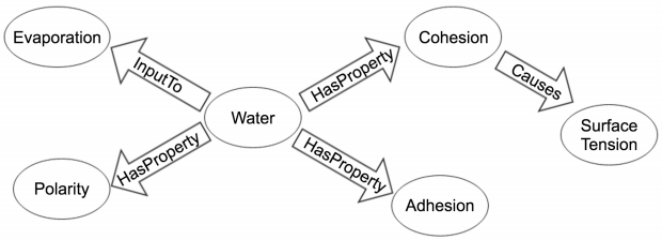


Рис. 2.8: Часть онтологии биологии от [4]

генерации вопросов, поскольку генерация элементов викторины требует как QG, так и DG.

2.4.1 Критерии для хорошего вопроса

В литературе предлагается множество критериев хороших вопросов с точки зрения преподавания и обучения. Наиболее часто обсуждаются четыре критерия, приведенные ниже:

1. Фокусируется на ключевом моменте текста ([17], [18]);
2. Сбалансированная сложность ([17]);
3. Апеллирует к пониманию и навыкам мышления высшего порядка учащихся, например, резюмирование, анализ отношения, оценка последствий ([4]).

2.4.2 Методы составления вопросов

Применяемые в настоящее время методы делятся на две основные группы: подходы на основе правил и подходы на основе нейронных сетей. Подходы, основанные на правилах, используют структуру данных и созданные вручную правила для создания вопроса. Подходы на основе ней-

ронных сетей являются более сложными и могут отражать более глубокие отношения между объектами.

Подходы, основанные на правилах Методы генерации вопросов на основе правил учитывают структуру данных (онтологию или структуру базы данных, или синтаксис предложения) и применяют некоторые предопределенные правила для создания вопроса.

Stasaski и Hearst [4] применил онтологию Biology для генерации вопросов на основе правил. Пример объектов и отношений показан на рисунке 2.8. Вместо того чтобы генерировать вопрос из триплета узел-отношение-узел, авторы рассмотрели возможность генерации вопросов на основе графовой структуры онтологии. Узел рассматривается как ответ, а три случайно выбранных исходящих отношения - как основа для вопроса. Например, если взять узел *"Вода"* и его связи *(DissolvesIn, "соль")*, *(HasProperty, "когезия")* и *(InputTo, "испарение")*, то вопрос будет звучать так: "Что растворяет соль, имеет когезию и является исходным для испарения?". После первого эксперимента авторы также изменили алгоритм, чтобы генерировать вопросы на основе двух связей, а не трех, как было изначально, и это увеличило результаты количественной оценки. Они также добавили в систему правила грамматической коррекции, чтобы вопросы викторины звучали более человекоподобно. Ограничение описанного метода заключается в том, что он сильно зависит от онтологии, которая доступна не для всех доменов и тем.

В Heilman в [19] была представлена трехэтапная схема генерации фактологических вопросов. На первом этапе сложные предложения преобразуются в простые фактологические предложения путем извлечения фак-

тологических утверждений и замены местоимений соответствующими существительными. На втором этапе входные предложения преобразуются в набор вопросов-кандидатов путем изменения порядка слов в предложении. Третий этап включал ранжирование набора с помощью статистической модели. Основным недостатком описанной схемы является то, что она требует высокоструктурированных данных с предварительными знаниями о синтаксисе. Таким образом, она вряд ли применима из-за отсутствия соответствующих данных и языков со сложным синтаксисом.

Подходы на основе нейронных сетей Методы, основанные на NN, используют модели глубокого обучения для генерации вопроса.

Первая нейронная сеть QG была представлена Serban и др. в [20]. Авторы рассматривали задачу QG как перевод из структурированной базы знаний в вопросы на человеческом языке. Они использовали архитектуру кодер-декодер с механизмами GRU и внимания. Однако модель попрежнему сильно зависит от базы знаний и поэтому может быть использована только в узких областях с существующими KB.

Jain, Zhang и Schwing [21] попытались решить проблему генерации визуальных вопросов. Они использовали вариационные автоэнкодеры и LSTM для создания набора возможных вопросов на основе входного изображения. Авторы предложили вариационный автоэнкодер. LSTM применяется для кодера (q-распределение) и декодера (p-распределение). Входное изображение и предложение кодируются в латентное представление с помощью q-распределения. Учитывая представление изображения и случайную выборку z, p-распределение реконструирует вопрос. В процессе вывода выборка z генерируется либо из нормального, либо из равномер-

ного распределения. Для получения дополнительной информации о VAE обратитесь к 2.3.1. Авторы утверждают, что парадигма вариационного автоэнкодера была выбрана потому, что они более устойчивы при обучении по сравнению с адверсарными сетями. Jain и др. сообщают, что сгенерированные вопросы можно считать либо простыми (на них можно ответить, просто взглянув на изображение), либо сложными (требующими глубокого человеческого понимания объектов и их взаимодействия). Они также выделили два основных типа проблем. Первая называется *неудачи распознавания* и вызвана неспособностью модели правильно распознать объекты на входном изображении. Второй тип неудач называется *co-occurrence based failure*, и он приводит к вопросам об объектах, которые не присутствуют на изображении, но часто встречаются вместе с представленными (например, вопросы о птицах на нептичьих изображениях деревьев).

2.5 Генерация дистракторов

В некоторых работах [22], [18] говорится, что генерация дистракторов является наиболее сложной задачей при создании элементов викторины.

2.5.1 Критерии для хорошего дистрактора

Критерии для хорошего дистрактора, которые были указаны в предыдущих работах:

1. Правдоподобный, трудно отличить от ответа ([23], [5], [1], [2], [4])
2. Semantically Consistent with Question and Answer ([24], [25], [17], [2])
3. Грамматически согласуется с вопросами и ответами ([24])

4. Не является правильным вариантом ([23], [1], [4], [2])

5. Не является модифицированной версией ответа или другого дистрактора ([24])

6. Иметь какой-то след в статье — необязательно ([24])

2.5.2 Поколение кандидатов

Самый простой подход - генерировать дистракторы человеком (как в [26]). Однако такой подход требует больших затрат времени и средств, а наша цель - сократить и то, и другое, поэтому мы заинтересованы в автоматической генерации дистракторов.

Еще одно важное соображение связано с идеальным количеством дистракторов. Graesser и Wisher [27] доказали, что идеальное количество дистракторов равно трем плюс правильный ответ, следовательно, наш дизайн исследования подразумевает необходимость генерировать три дистрактора.

Как и в задаче QG, автоматические методы можно разделить на две большие группы: основанные на правилах и основанные на нейронных сетях, обе из которых способны генерировать три дистрактора. Первый метод более интерпретируемый и в основном подходит для однословных дистракторов, в то время как второй может генерировать более сложные дистракторы. Существуют также методы, которые не подходят под эти категории, они обсуждаются в подразделе "Другие подходы"ниже.

Подходы, основанные на правилах Подходы, основанные на правилах, обычно используют некоторый датасет или базу знаний и метрики

сходства для создания дистракторов, которые похожи на ответ.

В [23], [26] авторы предложили использовать частотный подход. В этом методе генерируемый дистрактор - это просто слово в том же POS-теге, что и ключевое слово, которое имеет наибольшее количество вхождений с ним в большом корпусе. Идея основана на интуиции, что слова с близкой частотой встречаемости, вероятно, известны изучающим язык с одинаковым уровнем знаний. Существует огромное ограничение подхода: процедура не подходит для ключей с несколькими словами (метод, применяемый к каждому слову в ключе независимо, будет генерировать неправдоподобную и сомнительную последовательность).

Метод, основанный на коллокациях ([23]) похож на предыдущий, и он производит дистрактор, который имеет наиболее частую коллокацию либо с левым соседним словом ключа, либо с правым соседним словом. Как и в подходе, основанном на частоте, для ключей с более чем одним словом подход, основанный на коллокации, генерирует маловероятный дистрактор.

Lee и др. в [23] применил метод, основанный на неродных английских корпорациях. Метод генерирует отвлекающий предлог, который чаще всего по ошибке ставился между левым и правым соседними словами ключа неродными носителями языка. Авторы показали, что этот подход является наиболее привлекательным для студентов (по сравнению с методами, основанными на частоте и коллокации). Тем не менее, представленный подход вряд ли применим к открытым POS-тегам[1] из-за отсутствия доступных аналогичных данных.

В нескольких исследовательских проектах ([28], [26]) использовались различные измерения лингвистического сходства. К ним относятся следу-

[1]https://universaldependencies.org/u/pos/

ющие методы, но не ограничиваются ими: морфологический[2] (слова, принадлежащие к различным частям речи, но имеющие одну и ту же базовую форму, *bored — boring*), орфографический (слова с двумя или тремя измененными буквами, например, *bread — beard*), фонетические (слова с 2-3 измененными фонемами или с фонемами, замененными на похожие, например, *file — fly*), и графемный (дистрактор с наименьшим расстоянием Левенштейна от целевого слова). Слова-кандидаты сканируются по большому корпусу, и наиболее близкое слово с точки зрения некоторого измерения сходства считается дистрактором.

Pino и др. в [28] попытался объединить различные подходы, основанные на сходстве, чтобы получить дистракторы, расположенные дальше, чем целевое слово. Однако дистракторы, созданные с помощью смешанных стратегий, таких как OrthMorph (сочетание орфографического и морфологического методов) или PhonMorph (сочетание фонетического и морфологического методов), менее правдоподобны, и эксперименты показали, что испытуемые выбирают такие дистракторы реже.

Liang и др. в [22] расширил меры сходства для ключей с 2-5 словами и сформулировал задачу генерации дистракторов как задачу Learning to rank. После этого авторы свели задачу к задаче бинарной классификации, применив в качестве моделей классификации логистическую регрессию, Random Forest ([29]) и LambdaMART ([30]). Авторы обучили модель, используя дистракторы из входного образца в качестве положительных примеров и подмножество дистракторов из других образцов в качестве отрицательных примеров. Характеристики для классификации были получены путем комбинирования различных мер сходства для вопроса q, ответа a и

[2]Термины ‘морфологический‘ и ‘морфологический‘ относятся к одному и тому же понятию. Для простоты и последовательности термин ‘морфологический‘ используется везде

дистрактора d:

- Emb Sim (сходство встраивания),
- POS Sim (сходство по Жаккарду между POS-тегами a и d),
- ED (расстояние редактирования, или расстояние Левенштейна),
- Token Sim (парное сходство по Жаккарду между токенами q, a и d),
- Length (длина токенов a и d и разница между ними),
- Суффикс (длина самого длинного общего суффикса a и d,
- Freq (средняя частота слов в a и d),
- Single (единственное и множественное число последовательности a и d),
- Num (есть ли числа в a и d),
- Wiki Sim (сходство встраивания сущностей Википедии).

Авторы [22] обнаружили, что каскадное обучение (первая модель выбирает небольшое подмножество кандидатов, а вторая модель производит окончательное ранжирование), работает лучше, чем обучение с использованием одной модели. Liang и др. использовал две модели последовательно, одну за другой, однако параллельное пакетирование с двумя, тремя или более моделями потенциально может работать гораздо лучше.

Zesch и др. в [2] предложил метод, который использует контекстно-чувствительные и контекстно-нечувствительные правила вывода. Этот метод генерирует дистракторы, которые семантически похожи на целевое сло-

во, но не похожи в конкретном смысле, порождаемом контекстом цели. Например, *"сильный"* и *"мощный"* семантически похожи, но контекст, добавленный словом *"чай"*, делает их разными (*"сильный чай"* против *"мощный чай"*). Авторы сначала получили список возможных дистракторов, используя контекстно-зависимые правила (например, для целевого слова *"приобрести"* дистракторами могут быть *"купить"*, *"иметь"*, *"приобрести"*). Затем они применяли контекстно-зависимые правила для создания черного списка дистракторов. Такие слова являются правильными ответами и не могут быть использованы в качестве дистракторов (например, черный список для ключевого слова *"приобрести"* в предложении *"Microsoft приобретает Skype"* может быть *"купить"*, *"продать"*, *"приобрести"*). Окончательный набор дистракторов получается путем вычитания дистракторов из черного списка из первоначального списка. Недостатком предложенного метода является то, что он не применим ни для длинных ответов и дистракторов (из-за недостатка данных), ни для вопросов, заданных в естественной форме (из-за природы метода).

В [4] К. Стасаки и М. Херст используют онтологическую структуру для задач QG и DG. Узлы свойств n_1, n_2, n_3 связаны с узлом правильного ответа n отношениями r_1, r_2, r_3, соответственно. Чтобы убедиться, что узел-дистрактор m не является правильным ответом на вопрос, авторы проверили, что максимум два узла из n_1, n_2, n_3 связаны с m каким-либо отношением. Они изучили пять подходов к генерации дистракторов:

- Два совпадающих отношения (используются два отношения из r_1, r_2, r_3),
- Одно совпадение и одно новое отношение (используется одно отношение из r_1, r_2, r_3 и новое r_4),

- Два новых отношения (ни одно из r_1, r_2, r_3 и новые r_5, r_6 не используются),

- Одно совпадающее отношение (используется одно отношение из r_1, r_2, r_3),

- Структура узла (измерение сходства для пары узлов). Сходство структуры узлов рассчитывается по формуле

$$s_{n,m} = \text{count}(c_n \cap c_m) - \sum_r |l_{n,r} - l_{m,r}|$$

где c_n - множество узлов, которые соединены с узлом n, $l_{n,r}$ - количество связей, которые имеет узел n типа r.

Stasaski и Hearst [4] также предложил подход, основанный на онтологии. Каждый узел n из графа онтологии G состоит из серии слов, и он встраивается как e_n. Дистрактор выбирается путем сравнения узла-кандидата дистрактора n с узлом правильного ответа n по формуле

$$d = \underset{m \in G, m \neq n}{\text{argmax}} \left(\text{sim}[e_n, e_m]\right)$$

. где $\text{sim}[e_n, e_m]$ обозначает косинусное сходство между векторами e_n и e_m, а m - узел графа G. В другой методике авторы применили приведенную выше формулу для нахождения наиболее похожего узла на каждый из компонентов вопроса n_1, n_2 и n_3. Ограничения подходов, основанных на онтологиях, обусловлены их природой: онтологии обычно специфичны для конкретной области и доступны не для всех доменов.

Подходы на основе нейронных сетей Основываясь на IR-GAN (впервые представленной в [31]), модифицированной версии генеративной состязательной сети для информационного поиска, Liang и др. в [22] предложил состязательную схему обучения, состоящую из двух компонентов: генератора G и дискриминатора D. Генеративная модель G выдает условную вероятность генерации дистракторов с учетом вопроса и ключа — $P(d \mid q, a)$. Дискриминантная модель D пытается угадать, происходит ли данный дистрактор из G или из реальных обучающих данных. Несмотря на то, что данный подход основан на нейронных сетях, в ходе экспериментов он показал себя хуже, чем Random Forest. Предположительно, модель может быть улучшена путем обновления архитектуры, тонкой настройки предварительно обученных моделей или оптимизации гиперпараметров.

Gao и др. в [24] предложил использовать сеть обучения по принципу "последовательность-последовательность"и брать в качестве входных данных для NN интервал текста и вопрос для генерации списка дистракторов. Они описали иерархическую кодирующую-декодирующую структуру с динамическими и статическими механизмами внимания. Сначала данные проходят через кодер, который изучает контекстуализированное представление всей статьи на уровне слов и предложений. Затем механизм динамического внимания изучает важность каждого предложения. На третьем этапе статическое внимание используется для того, чтобы заставить динамическое внимание не фокусироваться на нерелевантных предложениях или предложениях, которые способствуют правильному ответу. И наконец, информация о вопросе сжимается, и снова задействуется механизм динамического внимания. Авторы заявили, что явное добавление сигналов наблюдения к обучению статического внимания может улучшить качество

ГД.

Lu, Ye, Ren и др. [5] сообщают о результатах, связанных с визуальным DG. В этой работе они, во-первых, представили новую задачу генерации текстовых дистракторов для VQA (DG-VQA). Их целью было создание наиболее запутанных дистракторов для задачи VQA. Авторы применили решение reinforcement learning, в котором модель DG получает вознаграждение от хорошо обученной модели VQA в зависимости от правильности выбранного моделью VQA варианта. В терминах RL, модель DG становится агентом политики, сгенерированный дистрактор становится действием, а модель VQA - средой. Применение моделей VQA имеет два основных преимущества. Во-первых, этот подход пытается оценить запутывающую способность дистрактора. Во-вторых, модель является несамостоятельной и не требует маркировки образцов дистракторов. Предложенное соревнование показало, что модели VQA могут быть использованы для обучения более точных и эффективных визуальных моделей QG и DG.

В работе [32] от Zhou, Luo и Wu предлагается использовать иерархическую сеть Co-Attention. Авторы утверждают, что существующие модели Seq2Seq не имитируют взаимодействие между статьей и вопросом, а также отношения между дистрактором и статьей. Предлагаемая модель способна отразить отношения между статьей и вопросом благодаря иерархической архитектуре с усиленным совместным вниманием. Связь между дистрактором и ответом обусловлена дополнительной потерей семантического сходства, что заставляет генерируемый дистрактор быть более релевантным статье. Кодер вычисляет представление предложения, комбинируя следующие вкрапления: представление слов внутри предложения на уровне предложения и представление вектора признаков предложения с

учетом вопроса. Состояние декодера инициализируется сжатым вопросом, чтобы обеспечить грамматическую согласованность между дистрактором и вопросом. Иерархическая оценка внимания рассчитывается итеративно с учетом представлений на уровне слов и конечных представлений предложений. В качестве направления будущей работы авторы рассматривают возможность учета правильного ответа. Эта идея потенциально может генерировать более запутанные дистракторы, но при этом может генерировать правильные (недостоверные) варианты.

Chung, Chan и Fan в [33] предлагают новую модель под названием BDG (BERT-based Distractor Generation). Модель итеративно делает вывод, т.е. предсказывает i-ую лексему длинного дистрактора. Изначально на вход модели подается C (комбинация текстового отрезка P, вопроса Q и ответа A). После этого модель предсказывает следующую лексему t_i в дистракторе на основе последовательности C и ранее предсказанных лексем дистрактора $t_1, t_2, ..., t_{i-1}$. Чтобы избежать генерации некогерентного вывода, результаты предыдущего декодирования учитываются при декодировании следующего дистрактора. Кроме того, авторы объединили модель BDG с параллельным MLM для повышения производительности модели DG. Chung и др. столкнулись с тем, что модель BGD имеет тенденцию генерировать дистракторы, похожие на ответ A. Чтобы отбить желание предсказывать лексемы из A при предсказании следующей лексемы t_i, они ввели отрицательную потерю ответа. Применив мультимодальный подход и введя регуляризацию Answer negative, авторы показали, что их модель является самой современной моделью для задачи DG для набора данных RACE. Подход обсуждается в отдельном разделе (2.6), так как модель используется в качестве базовой для предлагаемого решения.

Offerijns и др. [34]работали над всеми тремя проблемами: QG, DG, QA. Тем не менее, они сосредоточились на генерации дистракторов. Для этого они использовали модель GPT-2. Как и в исследовании [33], модель принимает контекст C в качестве входных данных, кроме того, она использует последовательность истинных дистракторов в качестве входных данных. Авторы сравнили модели GPT-2-small и GPT-2-medium. Удивительно, но маленькая версия показала лучшие результаты по показателям BLEU и ROUGE. Offerijns и др. использовал несколько примечательных методов. Во-первых, они применили штраф за повторение (предложенный в [35]), который наказывает модель за генерацию синтаксически похожих дистракторов. Во-вторых, заметив, что модель генерирует менее трех дистракторов, они повторяют шаги генерации до тех пор, пока не будет сгенерировано три дистрактора. Наконец, они ввели QA-фильтрацию, чтобы отсеять вопросы и дистракторы, которые в той или иной степени являются неправильными. Эксперименты показали, что автоматические оценки для метода лучше по сравнению с моделями, предложенными в [24], [32].

Другие подходы Альдабе и Мариткслар в [36] предложили различные методы, основанные на латентном семантическом анализе (LSA, [37]), который встраивает слова в векторное пространство на основе их встречаемости в наборе документов. Авторы извлекли из векторного пространства слова-кандидаты на отвлечение, которые лучше всего соответствуют запросу и предложению-носителю. Базовый метод LSA просто возвращает слова, которые не входят в предложение-носитель и имеют одинаковый POS с ключом. LSA & Semantic & Morphology генерирует отвлекающие слова, если они имеют хотя бы один семантический признак, например, *animate*,

human, учитывая формы слов. Подход LSA & Specialised Dictionary отдает предпочтение словам, которые относятся к той же теме в энциклопедическом словаре в качестве ключа. В методе LSA & Knowledge-based авторы предложили использовать PageRank для сортировки отвлекающих узлов в графе LKB (Lexical Knowledge Base) на основе их сходства с узлами, представляющими слова входного предложения.

2.5.3 Проверка надежности (некорректности)

Существенным критерием для сгенерированного дистрактора является неправильный выбор, поэтому дистрактор не должен семантически совпадать с ответом. Сюда относятся синонимы, переформулировки ключей, а также другие правильные ответы, которые определенно удовлетворяют постановке вопроса и правильно отвечают на него. В тестах, где пункты правил имеют только один правильный ответ, несколько правильных вариантов могут запутать тестируемых. Таким образом, отказ от ненадежных дистракторов крайне важен.

Некоторые работы (например, [36]) проверяют надежность вручную, с участием человека.

Как мы видели в литературе, огромное количество работ основано на идее, что сгенерированный дистрактор, не появляющийся в некоторой коллекции данных или базе знаний, вместе со своими соседними словами может считаться надежным.

Например, Зенч и Меламуд в работе [2] использовали контекстно-чувствительные правила вывода, принимая во внимание контекст основного предложения. Если сгенерированный дистрактор попадает в черный список, созданный контекстно-чувствительными правилами, он может быть

отвергнут.

Работа от Sumita и др. [38] проверяют надежность путем поиска в интернете ствolового предложения, где пустота заполняется сгенерированным дистрактором. Они предположили, что если есть хотя бы одно совпадение, дистрактор может быть отвергнут. Несмотря на широкое языковое разнообразие и огромное количество данных в Интернете, нулевое совпадение не обязательно означает, что дистрактор надежен.

Correia, Baptista, Mamede и др. [26] предложили использовать лексический ресурс PAPEL, который поддерживает отношения синоним, гипоним и гипероним и исключить дистракторы с таким типом отношений с целевым словом. Они также использовали онтологическую модель MWN.PT, которая каталогизирует каждое слово в конкретной области. Исследователи использовали это свойство для исключения слов в одной и той же области (например, слова "юрист"и "врач"представляют профессию и могут быть исключены).

2.6 Базовый подход

В качестве базового подхода для задачи DG используется модель генерации дистракторов на основе BERT (BDG) и ее модификации из работы [33]. Мы уже упоминали эту работу в предыдущем разделе 2.5, но это был поверхностный обзор без глубоких деталей. В отличие от этого, данный раздел полностью посвящен подробному объяснению подхода. Подробное описание этой конкретной работы необходимо, так как предлагаемое нами решение основано на этом исследовании, и многие понятия и формулы из этой работы используются в разделе Методология 3.

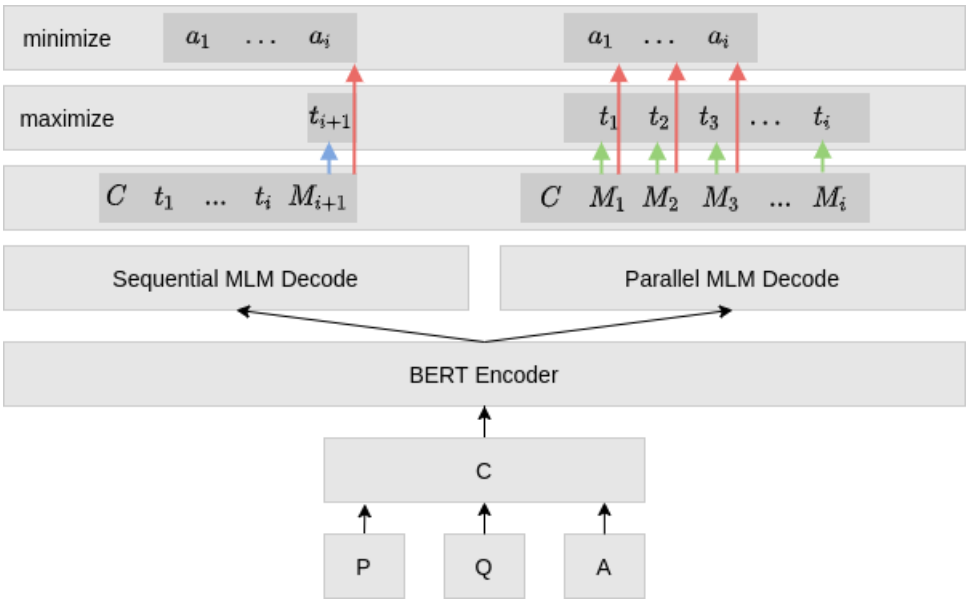


Рис. 2.9: Архитектура BDG [33]

Авторы статьи, Chung и др., нацелились на решение двух существенных проблем в задаче генерации дистракторов. Первая проблема - это качество генерируемых дистракторов, которое может быть определено по соответствию или несоответствию набору критериев, определенных в подразделе 2.5.1). Для решения первой проблемы они использовали модель BERT (подробнее о BERT см. в Разделе 2.2.3). Вторая проблема - генерация множественных дистракторов, которая определяется как совместная генерация более одного дистрактора на вопрос. Для решения второй проблемы Chung и др. рассмотрел задачу генерации нескольких дистракторов как задачу максимального покрытия (выбор из множества вариантов не более k подмножеств так, чтобы объединение выбранных подмножеств имело максимальный размер). Решения этих задач описаны в подразделах 2.6.1 и 2.6.2 соответственно.

Общая схема модели BDG показана на Рис. 2.9. Здесь показаны входные данные для модели (вектор C), оба режима обучения (последователь-

ный MLM и параллельный MLM) и цели модели (максимизация сходства с целевыми дистракторами и минимизация сходства с правильным ответом).

2.6.1 Качество генерируемых дистракторов

Как уже говорилось в предыдущих главах, дистракторы играют важную роль в проверке знаний учащихся. Качественные дистракторы позволяют сделать это гораздо лучше.

Iter.	Input Sequence	Predict
1	[C] C [S] [M]	Because
2	[C] C [S] Because [M]	Henry
3	[C] C [S] Because Henry [M]	didn't
4	[C] C [S] Because Henry didn't [M]	want
5	[C] C [S] Because Henry didn't want [M]	to
6	[C] C [S] Because Henry didn't want to [M]	go
7	[C] C [S] Because Henry didn't want to go [M]	
8	[C] C [S] Because Henry didn't want to go. [M]	[S]

Таблица I: Пример выполнения схемы BDG

Последовательное Маскирующее Языковое моделирование

Модель BDG в настройке Sequential Masked Language Modeling (Последовательное Маскирующее Языковое моделирование) генерирует дистрактор токен за токеном (дистрактор состоит из нескольких слов, и модель генерирует его слово за словом). Последовательное МЛМ показано в левой части Рис. 2.9). В начале процесса генерации модель берет триплет C (который состоит из абзаца P, вопроса Q и ответа A) и производит $\hat{t}_1$[3],

[3]Мы изменили обозначение токена дистрактора с d_j, как было изначально, на t_j, чтобы не путать одиночный дистрактор с его токеном

первый токен дистрактора. На следующих этапах модели дается конкатенация триплета C и последовательности лексем дистракторов, сгенерированных ранее $(\hat{t}_1, \hat{t}_2, ...\hat{t}_i)$, и предлагается выдать следующую лексему $\hat{t}_{i+1}$ (замаскированную лексемой $[M]$). Чтобы избежать генерации бессвязного вывода, при декодировании следующего дистрактора учитываются ранее декодированные результаты. Пример работы показан в Таблица I.

Формально, на i-й итерации BERT принимает последовательность $X_i = ([\mathrm{C}], C, [\mathrm{S}], \hat{t}_1, \ldots, \hat{t}_i, [\mathrm{M}])$ в качестве входа и производит вектор $\mathbf{h}_{[\mathrm{M}]}$, скрытое представление лексемы $[M]$, которая используется в X_i. Предсказание следующей лексемы $\hat{t}_{i+1}$ осуществляется путем применения линейного преобразования слоя $\mathbf{W}_{\mathrm{DG}}$ $\mathbb{R}^{h \times |V|}$ и последующей активации softmax для размера словаря:

$$p(w \mid X_i) = p(t_{i+1} \mid C, t_{1:i}) = \mathrm{softmax}\left(\mathbf{h}_{[\mathrm{M}]} \cdot \mathbf{W}_{\mathrm{DG}} + \mathbf{b}_{\mathrm{DG}}\right)$$
$$\hat{t}_{i+1} = \mathrm{argmax}_w \, p\left(w \mid X_i\right)$$

Потеря направлена на максимизацию предсказанной оценки получения токена t_i с учетом вектора C и последовательности предшествующих токенов $t_{1:i}$ и оптимизацию параметров модели θ:

$$\underset{\theta}{\text{minimize}} - \sum_{(C,D)\in\mathcal{D}} \phi_{\mathrm{S}}(C, D)$$

$$\phi_{\mathrm{S}}(C, D) = \sum_{i=0}^{|D|} \left(\log_2 p\left(t_{i+1} \mid C, t_{1:i};\ \theta\right)\right), \tag{2.1}$$

где $\mathcal{D}$ - датасет, из которого взяты пары вектор контекста и дистрактор (C, D).

Многозадачность с параллельным MLM Эксперименты Chung и др. показали, что модель с настройкой последовательного MLM хорошо справляется с вопросами на уровне предложений и в то же время плохо справляется с вопросами, ориентированными на резюме, которые требуют от модели способности семантически обобщать отрывок (например, "О чем этот текст?"). В качестве решения был предложен многозадачный подход с Parallel MLM; в его настройках Sequential и Parallel MLM выполняются независимо друг от друга (см. левую и правую стороны Рис. 2.9). В отличие от последовательного MLM, параллельный MLM маскирует все слова внутри дистрактора специальными лексемами $[\mathrm{M}]_{\mathrm{t}_i}$ и выдает всю выборку дистрактора.

Входная последовательность X для P-MLM основана на векторе C и масках-токенах $[\mathrm{M}]_{\mathrm{t}_i}$, а именно $X = ([\mathrm{C}], C, [\mathrm{S}], [\mathrm{M}]_{\mathrm{t}_1}, [\mathrm{M}]_{\mathrm{t}_2}, \ldots, [\mathrm{M}]_{\mathrm{t}_{|\mathrm{D}|}})$. BERT-стеки возвращают $\mathbf{h}_{[M]_{t_i}}$ $\mathbb{R}^h$, скрытое представление $[\mathrm{M}]_{\mathrm{t}_i}$ из X.

Предсказание лексемы $\hat{t}_i$ осуществляется путем применения линейного преобразования слоя $\mathbf{W}_{\mathrm{P-MLM}}$ $\mathbb{R}^{h \times |V|}$ и последующей активации softmax к объему словаря:

$$p(w \mid X) = \mathrm{softmax}\left(\mathbf{h}_{[\mathbb{M}]_{\mathrm{t_i}}} \cdot \mathbf{W}_{\mathrm{P-MLM}} + \mathbf{b}_{\mathrm{P-MLM}}\right)$$
$$\hat{t}_i = \mathrm{argmax}_w \Pr(w \mid X)$$

Функция потерь для P-MLM определяется следующим образом:

$$\underset{\theta}{\mathrm{minimize}} - \sum_{(C,D)\in\mathcal{D}} \phi_{\mathrm{P-MLM}}(C, D)$$

$$\phi_{\mathrm{P-MLM}}(C, D) = \sum_{\forall\, t_i} \log_2 p\left(t_i \mid C, [\mathrm{M}]_{t_i};\ \theta\right) \tag{2.2}$$

Авторы предлагают совместно обучать BDG и P-MLM, используя многозадачную функцию потерь:

$$\underset{\theta}{\text{minimize}} - \sum_{(C,D)\in\mathcal{D}} \phi_{\text{PM}}(C, D)$$

$$\phi_{\text{PM}}(C, D) = \sum_{(C,D)\in\mathcal{D}} \left[\phi_{\text{S}}(C, D) + \gamma \cdot \phi_{\text{P-MLM}}(C, D)\right], \tag{2.3}$$

где $\phi_{\text{S}}(C, D)$ - потеря, определенная в уравнении 2.1, а γ - гиперпараметр, управляющий весом между двумя задачами.

Отрицательная регуляризация

Как заметил Chung и др., модель BGD часто генерирует дистракторы, похожие на ответ A. В качестве решения этой проблемы авторы использовали отрицательную потерю ответа, чтобы наказать модель за генерацию лексем t_i, которые похожи на лексемы из последовательности ответов A.

$$\underset{\theta}{\text{minimize}} - \sum_{(C,D)\in\mathcal{D}} \left(\phi_{\text{AN}}(C, D) + \gamma \cdot \phi_{\text{P-MLM}}(C, D)\right)$$

$$\phi_{\text{AN}} = \sum_{i=0}^{|D|} \left(\log_2 p\left(t_{i+1} \mid C, t_{1:i};\ \theta\right) + \sum_{a_j\in A} \log_2\left(1 - p(a_j \mid C, [\text{M}]_{a_j};\ \theta)\right)\right. \tag{2.4}$$

2.6.2 Генерация нескольких дистракторов

Очень важно генерировать несколько дистракторов, а не только один, так как в идеале в наборе вариантов [27] есть три дистрактора и один ответ.

Выбор отвлекающих факторов путем максимизации энтропии

Методики предыдущих работ обычно подразумевают генерацию одного дистрактора. Некоторые другие подходы включают генерацию нескольких дистракторов, что обычно делается с помощью лучевого поиска. Поиск по лучу обычно дает образцы, относящиеся к одному понятию ("Молодая женщина, которая носит белую шляпу"и "Женщина, которая носит белую шляпу"), что делает предмет викторины бесполезным.

Идея максимизации энтропии заключается в том, чтобы выбрать набор дистракторов, учитывая семантическое разнообразие, а не индивидуально выбирать топ-к дистракторов на основе вероятности предсказания. Генерация набора разнообразных дистракторов решает не только проблему генерации множества дистракторов, но и общее качество набора дистракторов и всего задания теста.

В дополнение к модели $\mathbb{DG}_{\mathrm{S}}$, модель понимания чтения с множественным выбором $\mathbb{M}_{\mathrm{MRC}}$ участвует в генерации множественных дистракторов. Она обучается путем максимизации вероятности ответа при минимизации вероятности дистрактора.

Алгоритм следующий:

1. $\mathbb{DG}_{\mathrm{S}}$ генерирует набор дистракторов $\hat{D} = \left\{ \hat{d}_1, \hat{d}_2, \ldots, \hat{d}_n \right\}$;

2. Формируйте тройки, состоящие из трех сгенерированных дистракторов: $\left\{ (d_i, d_j, d_k) \mid i \neq j \neq k, d_i, d_j, d_k \in \hat{D} \right\}$;

3. Добавьте к тройке ответ A и составьте набор вариантов $O = \{\hat{d}_1, \hat{d}_2, \hat{d}_3, A\}$;

4. Среди всех наборов опционов найдите O, который максимизирует эн-

тропию:

$$\text{максимизация } -\sum_{o_i \in O} p_{o_i} \log_2 p_{o_i}$$

. где p_{o_i} - вероятность, назначенная MRC для варианта o_i в качестве правильного ответа.

BDG-EM

Идея подхода BDG-EM (BERT Distractor Generation - Entropy Maximization) заключается в улучшении подхода максимизации энтропии путем генерации $\hat{d}_1, \hat{d}_2, \hat{d}_3$ с использованием различных вариаций модели BDG:

- $\mathbb{DG}_{\text{S}}$, базовая модель с простым последовательным MLM;
- $\mathbb{DG}_{\text{PM}}$, модель с многозадачным обучением Parellel MLM (P-MLM);
- $\mathbb{DG}_{\text{AN+PM}}$, модель с отрицательной регуляризацией ответа и многозадачным обучением P-MLM.

В BDG-EM тройное множество определяется следующим образом:

$$\left\{ (d_i, d_j, d_k) \mid d_i \in \hat{D}, d_j \in \hat{D}_{\text{PM}}, d_k \in \hat{D}_{\text{PM+AN}} \right\}$$

2.6.3 Заключение

Базовый подход был обучен и протестирован на обновленном наборе данных RACE. Авторы рассчитали метрики BLEU и ROUGE для всех трех конфигураций BDG, модели на основе GPT, модели DS-Att ([24]) и модели CO-Att ([32]). Сравнение производительности показано в Таблица II. Наи-

	BLEU 1	BLEU 2	BLEU 3	BLEU 4	ROUGE L
BDG $_{\text{AN+PM}}$	39.52	24.29	17.28	13.28	33.40
BDG $_{\text{PM}}$	**39.81**	**24.81**	**17.66**	**13.56**	**34.01**
BDG	35.30	20.65	13.66	9.53	31.11
GPT	36.49	20.75	13.31	9.31	31.59
DS-Att.	27.32	14.69	9.29	6.47	15.12
CO-Att.	28.65	15.15	9.77	7.01	15.39

Таблица II: Сравнение производительности при подсчете баллов по токенам. Лучшие результаты по каждой метрике выделены жирным шрифтом

лучшие результаты показала конфигурация BDG_{PM}, которая превзошла существующие решения в задаче генерации дистракторов.

Модель BDG и ее модификации достигли результатов SOTA в задаче DG, однако, учитывая набор критериев для хорошего дистрактора (см. раздел 2.5.1), здесь есть много места для улучшений. Стремясь удовлетворить этим условиям, мы построили различные рамки, состоящие из нескольких моделей, описанных далее. Предлагаемое решение обсуждается в главе 3.

2.7 Датасеты

В Liang и др. в [22] говорится, что не существует эталонного набора данных для задачи DG, что затрудняет сравнение методов. Тем не менее, существует множество наборов данных, используемых в различных смежных работах, которые могут быть применимы к нашим экспериментам и заслуживают внимания.

Для генерации вопросов или дистракторов можно использовать три типа наборов данных: датасеты типа "клозет датасеты на основе диапазонов и датасеты с экзаменов.

2.7.1 Датасеты закрытого типа

В наборах данных типа cloze вопросы формулируются путем удаления слова или последовательности слов в предложении. Такие вопросы называются клозетными вопросами или вопросами с заполнением пробелов.

Примеры: CNN/Daily Mail ([39]), CBT ([40]), BT ([41]), WDW ([42]).

2.7.2 Датасеты на основе параграфов

Вопросы из наборов данных на основе параграфов строятся на основе некоторого текстового параграфа, и, таким образом, элемент теста из этого типа набора данных имеет соответствующий текст.

Примерами являются SQuAD ([43], [44]), SberQUAD ([45]), NEWSQA ([46]), MS MARCO ([47]) и TriviaQA ([48]).

Одним из самых популярных является SQuAD, Stanford Question Answering Dataset. Он состоит из сгенерированных человеком вопросов для задачи понимания прочитанного, написанных на английском языке. Вопросы основаны на отрывках текста из статей Википедии. Версия 1.1 SQuAD ([43]) содержит 100 000+ пар вопрос-ответ на 500+ статей Википедии. SQuAD 2.0 ([44]) расширяет SQuAD 1.1 более чем 50 000 вопросов без ответов, написанных краудворкерами, чтобы выглядеть похожими на вопросы с ответами. Русская версия SQuAD называется SberQuAD ([45]) и содержит более 50 000 триплетов параграф-вопрос-ответ.

2.7.3 Датасеты с экзаменов

Вопросы, относящиеся к этому типу наборов данных, обычно формируются как вопросы с множественным выбором и содержат список возмож-

ных дистракторов. Некоторые датасеты также предоставляют статистику о том, как испытуемые отвечали на список вопросов.

Примеры: AI2 ESSQ ([49]), NTCIR QA Lab ([50]), EET at CLEF QA Track ([51]), RACE ([52]), MCTest ([53]).

Самым современным является RACE, который был собран из экзаменов по английскому языку для китайских школьников средних и старших классов. Он состоит из около 28 000 отрывков и около 100 000 вопросов, составленных экспертами в данной области (преподавателями английского языка). В отличие от SQuAD, RACE имеет от одного до трех истинных дистракторов на образец.

2.8 Оценка методов генерации дистракторов

Как сообщает Liang, Yang, Dave и др. в [22], не существует эталонного набора данных для DG, что затрудняет прямое сравнение методов.

В литературе предлагаются различные способы сравнения качества методов. В целом их можно разделить на три группы: оценка человеком, оценка на основе метрик и состязательная оценка. Хотя человеческая оценка является наиболее представительной, она требует больших затрат времени и денег. Автоматические метрики легко и быстро вычисляются, но они не так точны, как человеческая оценка. Оценка на основе состязательности сложна в реализации и интерпретации, но она очень близка к человеческому восприятию.

2.8.1 Человеческая оценка

Авторы многих работ (например, [4], [2]) просили людей оценить вопросы и дистракторы либо по количественным метрикам, либо по качественным метрикам. Академический уровень оценщиков варьируется от студентов-бакалавров до профессоров. К сожалению, такой подход приводит к затратам времени и денег и вряд ли применим для сравнения в режиме реального времени. Более того, даже с учетом предоставленного списка критериев оценки, результаты сильно отличаются от оценщика к оценщику. Mitkov и др. в [17], [18] попросили студентов пройти сгенерированный тест, а затем авторы оценили дистракторы, основываясь на том, как испытуемые ответили на каждый пункт отдельно и на тест в целом.

В [17], [18], [36] авторы рассчитывали классические измерения теории тестов (например, трудность пункта, дискриминационная способность и полезность или эффективность дистракторов). Студенты были разделены на три группы в соответствии с уровнем их предварительных знаний: низкий, средний и высокий.

- Трудность предмета рассчитывается как $ID = \frac{C}{T} \cdot 100$, где C - число студентов, правильно ответивших на вопрос, а T - общее число студентов.

- Формула дискриминационной способности выглядит следующим образом: $DP = \frac{C_U - C_L}{T/2}$, где C_U и C_L - количество студентов в верхней и нижней группах соответственно, которые правильно ответили на вопрос.

- Полезность дистракторов (или эффективность дистракторов) оценивается путем сравнения количества студентов из верхней и нижней

групп, которые выбрали определенный дистрактор (дистрактор считается полезным, если он привлек больше студентов из нижней группы, чем из верхней).

Гудрич в [25] предложил измерять потенцию, которая рассчитывается как процент студентов, сделавших определенный выбор. На основе этих расчетов авторы [28], [23] и [26] определили список вариантов как правильный ответ плюс дистракторы, сгенерированные различными моделями, и подсчитали, сколько раз был выбран каждый тип дистрактора. Дистракторы, которые выбирались чаще всего, считались сложными, а модель, которая их генерировала, - лучшей.

В некоторых работах, связанных с изучением языка (например, [28], [26], [23]), авторы предлагали подсчитывать успеваемость студентов с разными родными языками. Он похож на показатель полезности дистракторов, в том смысле, что носители языка похожи на студентов из старшей группы, и они должны показывать более высокие результаты, чем не носители языка.

Pino и др. [28] также предложили измерять среднее время, необходимое испытуемым для выбора ответа или одного из дистракторов. Авторы утверждали, что это измерение может коррелировать со сложностью пункта.

Исследователи изучали генерацию дистракторов для клозетных вопросов в [26]. Correia и др. провел два эксперимента. В предварительном эксперименте они задавали вопросы без выбора ответа участникам тестирования и сравнивали ответы каждого из четырех испытуемых с ответом. Это позволило выявить наиболее распространенные причины неправильного выбора. Авторы подчеркнули, что выбор слова, которое не является

целевым, не означает, что он неадекватен.

2.8.2 Автоматическая оценка

Автоматические метрики сравнивают сгенерированные дистракторы с предопределенной базовой истиной.

В чем заключается основная идея такого рода автоматических метрик? Чем ближе автоматически сгенерированный образец к образцу, сгенерированному человеком, тем он лучше. Таким образом, для оценки работы модели требуется набор эталонных (сгенерированных человеком) образцов.

Lu и др. в [5] утверждает, что несмотря на то, что автоматические метрики (такие как BLEU и ROUGE) в основном применяются для оценки дистракторов, они не могут определить, может ли сгенерированный дистрактор запутать тестируемых. Кроме того, единственный способ правильно оценить сгенерированные дистракторы с помощью автоматических метрик - это сравнить их с полным списком истинных дистракторов, который невозможно собрать, учитывая различные лингвистические инструменты и техники (синонимы, перефразирование, синтаксические вариации и т.д.). Тем не менее, они очень полезны для оценки качества дистракторов благодаря скорости процедуры оценки и ее интерпретируемости.

BLEU

Gao и др. в [24], Jain и др. в [21], Chung и др. в [33] применялся показатель BLEU для оценки сходства сгенерированных дистракторов с истиной.

Показатель BLEU был представлен Papineni и др. в [54] для оценки машинного перевода (MT). Оценка BLEU использует модифицированную

форму точности для сравнения сгенерированного дистрактора с эталонной истиной. Авторы подчеркнули, что предложения-кандидаты, представляющие собой последовательность слов, обычно берут свои подпоследовательности из разных эталонных образцов. Например, для заданных эталонных переводов "There is a kitty on the table"и "Cat is on the table "Kitty is on the table"является хорошим кандидатом. Оценка BLEU вычисляется для каждого предложения отдельно, а затем усредняется по всему набору данных.

Papineni и др. заявил, что одним из самых простых подходов к оценке модели MT является подсчет точности: разделите количество слов (униграмм) перевода-кандидата, которые встречаются в любом эталонном переводе, на общее количество слов в переводе-кандидате. Недостаток простой точности в том, что она присваивает максимальный балл предложению-кандидату, которое построено путем многократного повторения любого слова из опорного предложения. Авторы поняли, что опорное слово следует считать исчерпанным после того, как найдено подходящее слово кандидата. Они назвали эту идею модифицированной униграммной точностью p_1 и формализовали ее следующим образом:

$$\text{Count}_{\text{clip}} = \min(\text{Count}, \text{Max\_Ref\_Count})$$

$$p_1 = \frac{\sum_{w \in \mathcal{C}} \text{Count}_{\text{clip}}(w)}{\sum_{w' \in C} \text{Count}(w')}$$

.

где *Count* - количество раз, когда слово-кандидат встречается в переводе кандидата, *Max_Ref_Count* - максимальное количество раз, когда слово-кандидат встречается в любом одном эталонном переводе, а w - един-

ственное слово из предложения кандидата $\mathcal{C}$.

Модифицированная n-граммная точность p_n для нескольких предложений-кандидатов рассчитывается аналогичным образом:

$$p_n = \frac{\sum_{\mathcal{C}\in} \sum_{n\text{-}gram\in\mathcal{C}} \text{Count}_{\text{clip}}(n\text{-}gram)}{\sum_{\mathcal{C}'\in} \sum_{n\text{-}gram'\in\mathcal{C}'} \text{Count}(n\text{-}gram')}$$

Заметив, что p_n наказывает переводы, которые длиннее ссылок, авторы ввели штраф за краткость (BP). Они обозначили длину перевода-кандидата как c, а эффективную длину корпуса ссылок как r (сумма длин наилучших совпадений для каждого предложения-кандидата в корпусе) и рассчитали штраф за краткость по формуле:

$$BP = \begin{cases} 1 & \text{если } c > r \\ e^{(1-r/c)} & \text{если } c \leq r \end{cases}$$

Окончательная формула для BLEU-score выглядит следующим образом

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

.

Оценка BLUE варьируется от 0 до 1, где 1 - наилучшее значение. Однако даже человеческий перевод не сможет получить оценку 1, потому что оценка 1 получается только в том случае, если предложения кандидата и ссылки полностью идентичны. Тем не менее, BLEU-score является стандартным алгоритмом для оценки и сравнения подходов MT.

ROUGE

Авторы [24], [33] также применяли оценку ROUGE. Оценка ROUGE - это модифицированная версия отзыва: $ROUGE_1$, $ROUGE_2$, $ROUGE_L$ - отзывы униграмм, биграмм и длиннейшей общей подпоследовательности (LCS), соответственно. Оценка ROUGE была предложена автором Lin в [55] для автоматической оценки резюме.

ROUGE-N был представлен как n-граммный отзыв между резюме кандидата и набором эталонных резюме:

$$ROUGE_N = \frac{\sum_{S \in \{ReferenceSummaries\}} \sum_{n\text{-}gram \in S} \mathrm{Count}_{\mathrm{match}}\left(n\text{-}gram\right)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{n\text{-}gram \in S} \mathrm{Count}\left(n\text{-}gram\right)}$$

Где $\mathrm{Count}_{\mathrm{match}}(n\text{-}gram)$ это максимальное количество n-грамм, которые встречаются как в резюме кандидата, так и в наборе ссылочных резюме. Суммирование производится по всем ссылочным резюме в номинаторе, чтобы дать более высокие баллы n-граммам, которые встречаются в нескольких ссылочных резюме.

ROUGE-L - это модифицированная версия ROUGE-N, которая основана на самой длинной общей подпоследовательности (LCS) резюме кандидата и ссылки. Для предложения эталонного резюме S длиной n и предложения резюме кандидата C длиной m он рассчитывается по следующей формуле:

$$R_{lcs} = \frac{LCS(S, C)}{m}$$

.

$$P_{lcs} = \frac{LCS(S, C)}{n}$$

.

$$ROUGE_L = F_{lcs} = \frac{\left(1 + \beta^2\right) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}$$

.

ROUGE-L имеет важное преимущество перед ROUGE-N: вместо последовательных совпадений он требует только совпадения в предложении, таким образом, лучше выражая порядок слов на уровне предложения.

Метрики ранжирования Liang и др. в [22] сравнивали модели на основе стандартных метрик ранжирования: top recall (R@10), precision (P@1, P@3), mean average precision (MAP@10), normalized discounted cumulative gain (NDCG@10) и mean reciprocal rank (RR).

2.8.3 Адверсариальная оценка

Liang и др. в [22] использовал GAN, в котором генератор G и дискриминатор D конкурируют друг с другом: по мере увеличения производительности D, производительность G также увеличивается. В конце обучения генератор становится "идеальным то есть он не может отличить, поступает ли входная выборка из G или из реальных обучающих данных.

Лу и др. в [5] доказали, что VQA-модель может быть использована в качестве оценщика. В то время как QG-модель генерирует более сложные дистракторы, QA-модель пытается ответить на вопрос как можно точнее. Авторы измерили ΔAcc, которая определяется как разница между производительностью модели VQA на дистракторах из набора данных и сге-

нерированных дистракторах. Высокий показатель ΔAcc означает хорошее качество дистракторов.

2.9 Выводы

В этой главе мы рассмотрели основы задач QG и DG. Сначала был представлен необходимый фон, состоящий из основных техник НЛП, которые используются в различных задачах, включая генерацию вопросов и генерацию дистракторов. После этого мы перечислили терминологию, используемую в литературе. Далее мы определили критерии хороших вопросов и описали существующие решения задач QG и DG. Решение обеих задач основано либо на правилах, созданных вручную (используют предварительные знания о языке, синтаксисе или базе знаний), либо на нейронных сетях (являются более общими и применимы для более широкого круга задач). Оба подхода позволяют уменьшить или полностью исключить участие человека в отсеве элементов викторины. Следующие подразделы были посвящены экспериментам: мы исследовали существующие датасеты викторин и подходы к оценке сгенерированных дистракторов.

Изучив литературу, мы обнаружили два основных **пробела**, связанных с генерацией дистракторов:

1. Большинство исследователей оценивали дистракторы по:

    - применение автоматических метрик, которые не могут полностью измерить качество; пункт, связанный с работой человека, требующий затрат времени и денег.

2. Большинство авторов генерировали вопросы и дистракторы отдель-

но, что может привести к худшим результатам.

Для решения вышеупомянутых проблем мы предлагаем два **исследовательских вопроса**:

1. Как использовать совместное обучение моделей DG и QA для повышения качества генерируемых дистракторов?
2. Как использовать подход GAN для повышения качества генерируемых дистракторов?

Глава 3

Методология

В этой главе описывается общая структура исследования: методы и процессы, использованные в ходе исследования, а также причины выбора конкретного подхода.

Глава организована следующим образом. Раздел Набор данных (3.1) посвящен объяснению источника и структуры набора данных, выбранного для исследования. Раздел Предобработка (3.2) описывает процедуры, которые используются для предварительной обработки данных для дальнейшей передачи их в модель. Раздел Модели (3.3) представляет модели, используемые для решения задачи генерации дистракторов [1]. Раздел Архитектуры (3.4) демонстрирует и объясняет предлагаемые архитектуры: модели, данные, методы оптимизации и их взаимосвязь. Раздел Выводы (3.5) подводит итог главы и предлагает заключительные мысли.

[1] Дистракторы - неправильные варианты в викторине с множественным выбором

3.1 Набор данных

Мы решили использовать набор данных RACE (Large-scale ReAding Comprehension Dataset From Examinations)[52] для эталонной оценки предложенного подхода и сравнения его с базовым подходом. Набор данных был первоначально предложен Lai и др. Он был собран в 2017 году из экзаменов по английскому языку для китайских школьников средних и старших классов в возрасте от 12 до 18 лет. Это набор данных по пониманию прочитанного с множественным выбором, состоящий из отрывков текста (текст, на основе которого нужно ответить на вопрос) и списка соответствующих элементов теста (вопрос, набор вариантов и правильный ответ). Набор данных состоит из пяти типов вопросов:

- Сопоставление слов: ответ на вопрос представлен в тексте с теми же словами, что и в самом вопросе.
- Перефразирование: Вопрос представляет собой перефразированную версию одного предложения из отрывка (ответ может быть найден в этом предложении).
- Рассуждение по одному предложению: Ответ может быть найден путем понимания и осмысления одного предложения.
- Рассуждение по нескольким предложениям: Ответ может быть найден путем понимания и осмысления нескольких предложений.
- Двусмысленный/недостаточный: Вопрос не имеет ответа или на него невозможно ответить на основе данного отрывка.

Мы отдали предпочтение набору данных RACE среди других наборов данных MCQ по нескольким причинам. Во-первых, это один из самых

популярных наборов данных. Во-вторых, он был собран на реальных экзаменах и, следовательно, состоит из вопросов, которые с большей вероятностью похожи на те, которые преподаватели могут использовать в процессе обучения. В-третьих, большинство вопросов теста (53,9%) относятся к однословным или многословным рассуждениям, которые являются наиболее сложными среди других и требуют от испытуемых более глубокого понимания текста.

Существует также недостаток набора данных, связанный с источником данных. Эти вопросы направлены на проверку способности студентов понимать иностранный язык и поэтому не проверяют знания студентов о терминологии, формулах и их взаимосвязи. Тем не менее, эти проблемы выходят за рамки нашего исследования и требуют отдельного изучения, поэтому мы их не рассматриваем.

Gao и др. в своей работе о генерации дистракторов [24] адаптировал оригинальный набор данных RACE. Авторы обнаружили, что некоторые дистракторы из оригинального набора данных RACE не имеют семантического отношения к статье, и вырезали их. Статистика по количеству образцов в сплитах и средней длине отрывков, вопросов и вариантов показана на рис. III. Эти дистракторы не сбивают с толку тестируемых и не являются правдоподобными. Кроме того, набор данных не сильно уменьшился после удаления этих образцов. Поэтому в данном исследовании мы используем обновленную версию RACE.

	Оригинальный [52]	**Обновленный** [24]
# train samples	129 226	96 501
# dev samples	16 132	12 089
# test samples	6 884	12 284
Avg Passage Len	321.9	347.0
Avg Question Len	10.0	9.9
Avg Option Len	5.3	5.5

Таблица III: Сравнение оригинальной и обновленной версий наборов данных RACE. Образец представляет собой четверку из абзаца, вопроса, правильного ответа и дистрактора

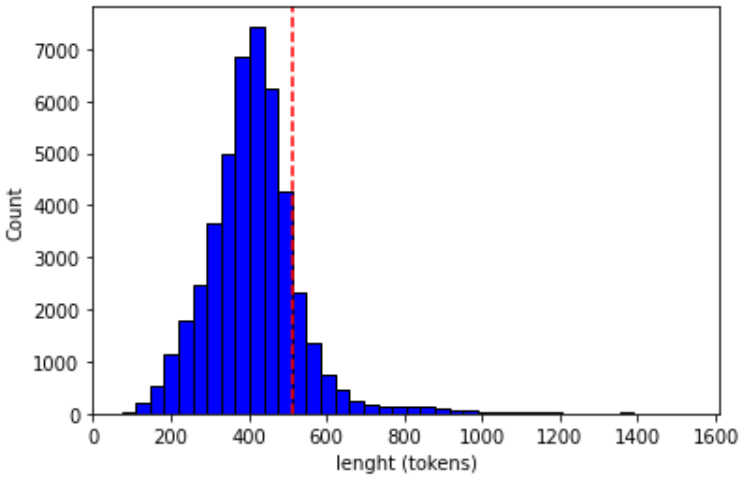


Рис. 3.1: Гистограмма длин образцов в тренировочной части обновленного набора данных RACE. Пунктирная линия показывает порог 512.

3.2 Предобработка

Обновленный набор данных RACE состоит из 3 файлов, по одному для каждого из сплитов train, dev и test. Он построен в формате *jsonlines* со следующими полями: абзац, вопрос, ответ и дистрактор. Каждый из дистракторов представлен в отдельном образце.

В процессе предварительной обработки мы оставляем каждый дистрактор в отдельном примере, поскольку каждый элемент викторины состоит из разного количества дистракторов, а выбранная модель (BERT)

ограничена входной длиной в 512 лексем. Для того чтобы получить качественные элементы викторины с дистракторами, которые неправильно отвечают на вопрос и в чем-то похожи на правильный ответ, мы должны передать в модель абзац, вопрос и ответ. Мы рассчитали длину каждого образца в обучающих данных в токенах. Гистограмма длин и порог в 512 токенов показаны на Рис. 3.1. Мы видим, что порог в 512 токенов довольно хорош, так как он уже сокращает 13% данных. Мы также видим, что уменьшение этого порога затронет гораздо больше образцов и может привести к гораздо худшей модели. Поэтому для этих 13% данных мы сокращаем начальную часть текста, так как считаем, что эта часть содержит наименее важную информацию, в отличие от вопроса, ответа, середины и конца текста. И, наконец, эта сокращенная версия сохраняется в разделенном CSV-файле с колонками CQA (контекст + вопрос + ответ), цель (дистрактор) и отрицательная цель (правильный ответ).

3.3 Модели

В этом и последующих разделах мы описываем основную часть исследования - предлагаемый подход. Он основан на базовом подходе, представленном [33], и привносит некоторые прогрессивные идеи для улучшения качества генерируемых дистракторов. Мы предлагаем использовать три модели, каждая из которых стремится удовлетворить критерии для качественных вопросов (сложные, путающие, не похожие на правильный ответ).

3.3.1 Модель генератора

Модель генератора (G) является костяком предлагаемого подхода. Получив на вход контекст, вопрос и ответ, она генерирует высококачественный дистрактор. Мы решили использовать модель BDG ([33]).

Мы выбрали этот подход по трем причинам. Во-первых, он основан на двунаправленных кодирующих представлениях от трансформаторов (BERT), самой современной модели (SOTA) для 104 языков. Во-вторых, модель показывает отличные результаты в задаче генерации дистракторов, и она считается SOTA модель для задачи генерации дистракторов (DG). Наконец, модель использует различные техники, которые заставляют генерируемые дистракторы соответствовать важным условиям: дистрактор не должен быть ни правильным вариантом, ни модифицированной версией другого дистрактора.

Мы рассматриваем три варианта модели: стандартная схема BDG $\mathbb{DG}_{\text{S}}$, схема BDG с параллельным многозадачным обучением $\mathbb{DG}_{\text{PM}}$, схема BDG с параллельным многозадачным обучением и отрицательной регуляризацией ответа $\mathbb{DG}_{\text{AN+PM}}$. Их функции потерь взяты из базового подхода [33] (2.6) без модификаций: ϕ_{S}, ϕ_{PM}, $\phi_{\text{AN+PM}}$. Поскольку каждая из конфигураций модели и соответствующие им функции потерь легко заменяемы, мы обозначаем потери генератора как ϕ_G (он может обозначать любую из ϕ_{S}, ϕ_{PM}, $\phi_{\text{AN+PM}}$).

3.3.2 Модель дискриминатора

Получив элемент теста, модель дискриминатора (D) угадывает, откуда он взят - из набора данных или из генератора. Модель дискриминатора

гарантирует, что сгенерированные дистракторы правдоподобны. Идея модели была навеяна подходом GAN, в нотации которого модель BDG рассматривается как модель генератора. Модель с той же целью была использована Liang и др. в статье [22].

Функция потрель адаптирована из оригинальной фунции GAN из статьи [15]. Функция потерь GAN определяется следующим образом:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\text{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Здесь дискриминатор D стремится максимизировать значение $\phi_{\text{D}-\text{GAN}} = \log(D(\boldsymbol{x})) + \log(1 - D(G(\boldsymbol{z})))$. В то же время генератор G минимизирует $\phi_{\text{G}-\text{GAN}} = \log(1 - D(G(\boldsymbol{z})))$.

3.3.3 Модель ответа на вопрос (QA)

Модель ответов на вопросы (для простоты QA) пытается ответить на вопрос и предсказать правильный вариант. Модель QA делает генерируемые дистракторы достаточно сложными. Модель так же QA используется для улучшения дистракторов в [5].

Поскольку это один из типов задачи классификации (выбрать один класс из нескольких), она использует одну из функций потерь классификации. В данной работе мы используем фукнцию перекрестное энтропии:

$$\phi_{\text{QA}} = -\sum_{c=1}^{M} y_{o,c} \log\left(p_{o,c}\right)$$

.

Кросс-энтропийная потеря (cross-entropy loss) наказывает как ошибки

типа 1, так и ошибки типа 2 но особенно те предсказания, которые являются уверенными и ошибочными. Наилучшая потеря равна нулю. Чем меньше CE, тем хуже модель Генератора (она не способна генерировать жесткие опции).

3.4 Архитектуры

Объединив вышеупомянутые модели, мы определили три архитектуры:

1. Генератор + QA.
2. Генератор + Дискриминатор.
3. Генератор + Дискриминатор + QA.

Ниже мы подробно описываем каждую архитектуру, используя архитектурную диаграмму и текстовое описание. Поскольку диаграммы и компоненты похожи, необходимо заранее определить все термины и обозначения:

- Генератор $\mathbb{G}$, Ответ на вопрос $\mathbb{QA}$ и Дискриминатор $\mathbb{D}$ определены выше в разделе 3.3.
- Набор данных RACE - это обновленная версия набора данных, описанного в разделе 3.1. Она состоит из пар контекста C (триплет из тестовой строки, вопроса и ответа) и реального дистрактора D_{real}.
- D_{fake} является дистрактором, сгенерированным $\mathbb{G}$ (термин "fake"взят из статьи GAN [15]). На основе этого рассчитывается фунция потери генерации.

- Answer prediction - это вектор оценок, присвоенных QA-моделью каждому варианту. Он используется для расчета потери классификации.
- Предсказание реальности/подделки - это выход модели Дискриминатора. Эти оценки используются для расчета фукнции мини-макс GAN.
- Общая потеря объединяет различные потери, используемые для оптимизации $\mathbb{G}$.

3.4.1 Генератор + QA

Этот подход объединяет модели Генератора и QA. Архитектура представлена в Рис. 3.2[2]. После того, как модель Генератора генерирует фальшивый образец, он конкатенируется с вектором C и передается модели QA. Она присваивает балл каждому варианту в рамках пункта теста, после чего мы вычисляем потери при классификации.

Общая потеря определяется как потеря генерации, вычитаемая из потери классификации с некоторым весом:

$$\phi_{\text{total}} = \phi_{\text{G}} - \lambda \cdot \phi_{\text{QA}} \tag{3.1}$$

Затем мы используем суммарные потери для оптимизации параметров генератора. В отличие от этого, веса модели $\mathbb{QA}$ не обновляются, мы используем предварительно обученную модель.

[2]Обратите внимание, что одинаковые компоненты схем расположены в одних и тех же местах (оставлено свободное пространство), чтобы упростить сравнение разных рисунков

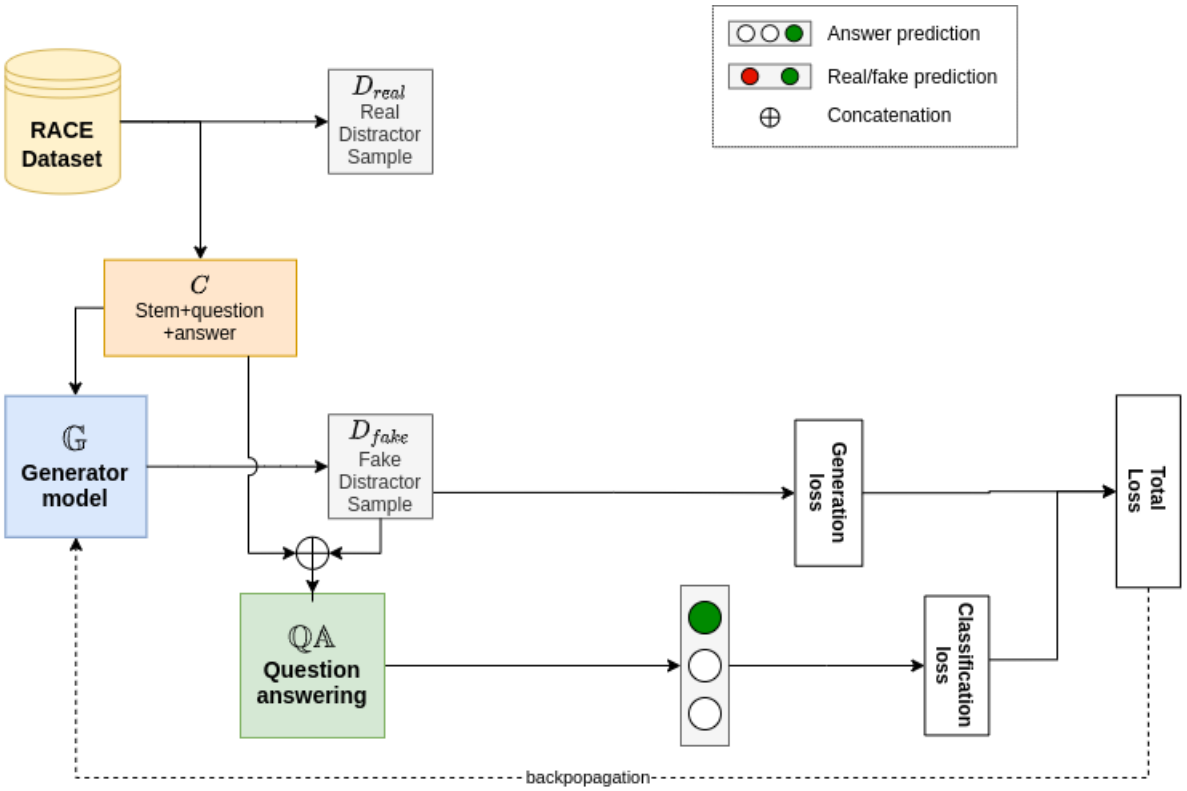


Рис. 3.2: Конфигурация Генератор + QA

3.4.2 Генератор + Дискриминатор

Архитектура Генератор + Дискриминатор объединяет модели Генератора и Дискриминатора вместе. Архитектура представлена в Рис. 3.3. Модель дискриминатора использует как D_{real}, так и D_{fake}. Мы объединяем их с последовательностью C и строим пары (C, D_{real}) и (C, D_{fake}). После этого мы независимо передаем эти пары в модель Дискриминатора. Она предсказывает оценки, которые прогнозируют, является ли данный образец настоящим (полученным из набора данных) или поддельным (сгенерированным генератором). Дискриминаторная часть потери минимакс GAN обновляет модель Дискриминатора. Генераторная часть потери идет в общую потерю, которая объединяет ее с потерей генерации от BDG: $\phi_{\text{total}} = \phi_G + \pi \cdot \phi_{G-GAN}$.

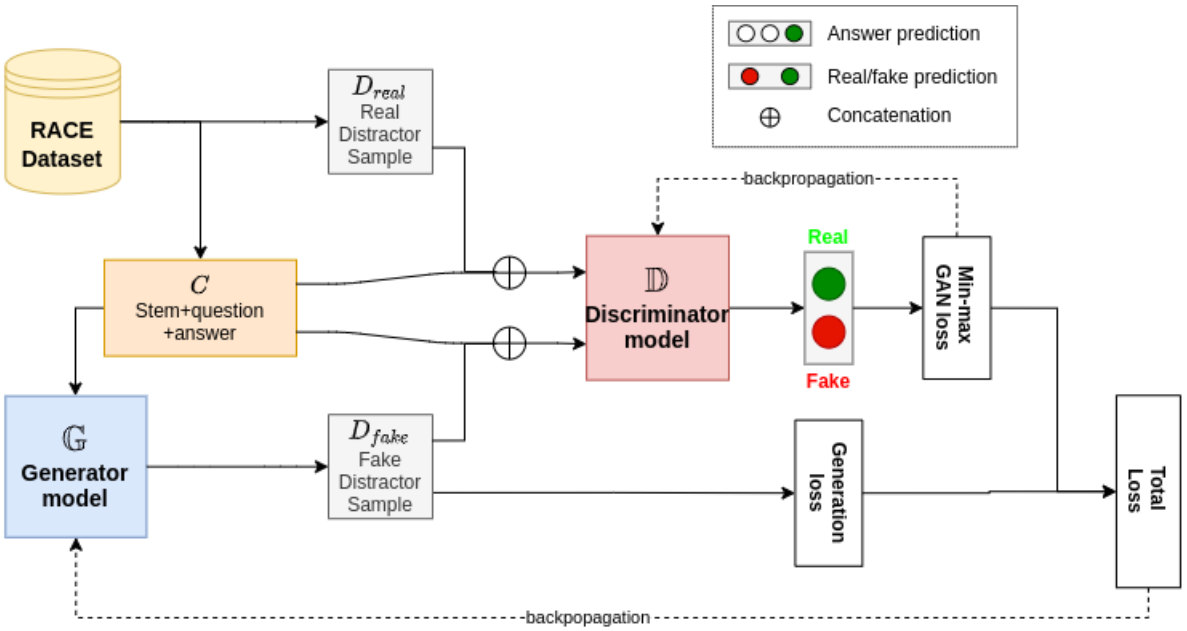


Рис. 3.3: Конфигурация Генератор + Дискриминатор

3.4.3 Генератор + Дискриминатор + QA

Архитектура Генератор + Дискриминатор + QA объединяет все три модели в единую структуру. Архитектура представлена в Рис. 3.4. Она объединяет идеи двух вышеупомянутых архитектур в одну, чтобы еще больше повысить качество. Аналогично архитектуре Генератор + Дискриминатор, мы вычисляем мини-макс потери GAN. Более того, по аналогии с классификацией Генератор + QA, мы вычисляем потери классификации.

Общая потеря генератора определяется следующим образом: $\phi_{total} = \phi_{\mathrm{G}} - \lambda \cdot \phi_{\mathrm{QA}} + \pi \cdot \phi_{\mathrm{G-GAN}}$.

3.5 Выводы

BDG сама по себе генерирует высококачественные дистракторы, но мы считаем, что их качество может быть улучшено. В этой главе мы показали три модели, которые используются для генерации дистракторов и

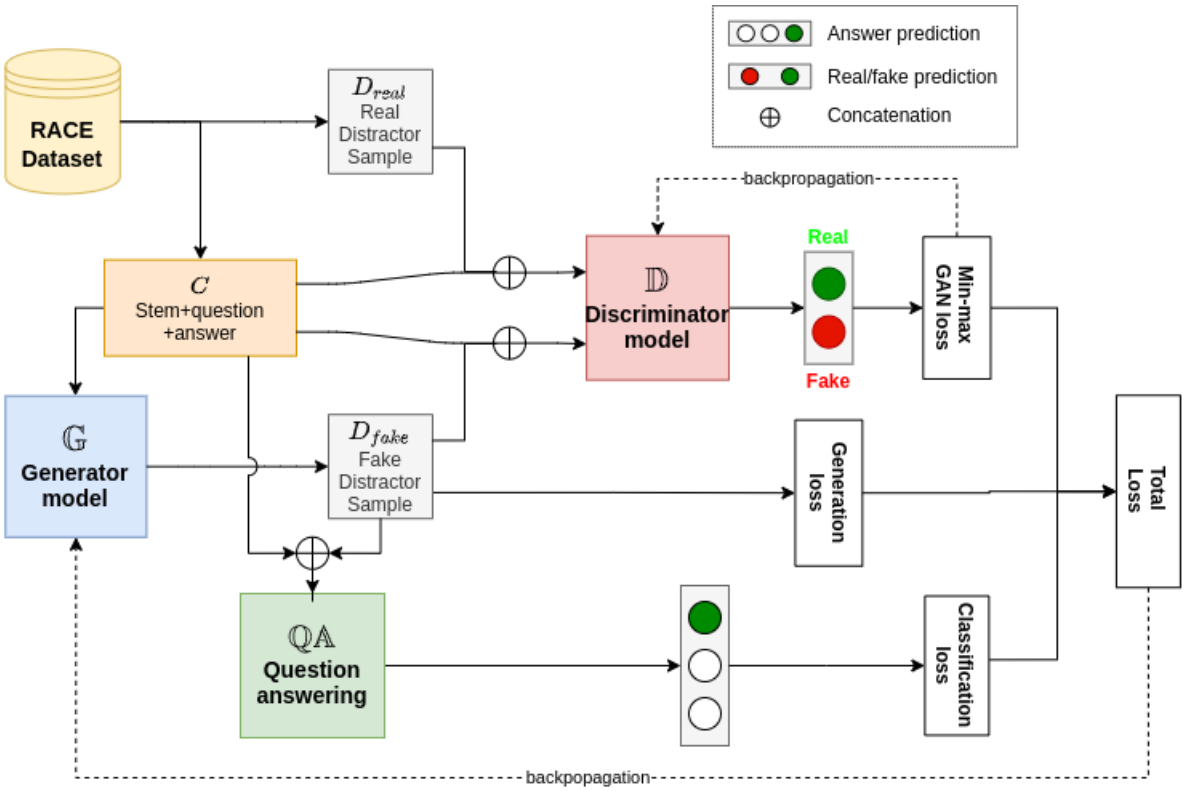


Рис. 3.4: Конфигурация Генератор + Дискриминатор + QA

повышения их качества путем придания им правдоподобности и/или сложности. Основываясь на представленной методике, мы предлагаем провести несколько экспериментов: Генератор (базовый эксперимент), Генератор+QA, Генератор+Discriminator, Генератор+Дискриминатор+QA. Подробное описание реализации всех моделей и архитектур, а также структуры экспериментов представлено в главе 4.

Глава 4

Реализация

Этот раздел посвящен описанию деталей реализации и демонстрации результатов. В разделе 4.1 мы описываем базовую реализацию. Раздел 4.2 посвящен описанию препятствий и проблем, с которыми мы столкнулись. В разделе 4.3 мы показываем, как мы реализовали наш подход, представленный в главе Методология. И, наконец, в разделе 4.4описаны эксперименты.

4.1 Базовая реализация

Chung, Chan и Fan **chung202020bertbased** заявляют, что их реализация базового подхода (2.6) основана на фреймворке transformers от *Hugging Face* [56]. Все эксперименты проводились на *bert-base-cased* модели. Они использовали $AdamW$ с начальной скоростью обучения $5 \cdot 10^{-5}$. Авторы обучили модель 6 эпох с размером пакета равным 6 на двух графических процессорах RTX Titan.

Chung и др. также предоставили код для базового подхода. Он распространяется в двух отдельных репозиториях VCS (система контроля вер-

сий): *TFkit*[1] и *BDG*[2]. *TFkit* - это надстройка над библиотекой *transformers* от HuggingFace, которая упрощает использование архитектуры трансформеров на различных задачах и моделях с небольшими изменениями конфигурации. Он содержит все необходимые скрипты для обучения, проверки и тестирования базовой модели. Второй репозиторий, *BDG*, содержит отдельные скрипты для предварительной обработки набора данных, вычисления различных статистик набора данных, обучения, оценки и вычисления QA оценок модели после того, как модель полностью обучена.

Код от Chung и др. использует следующие библиотеки:

- *PyTorch* — для работы с наборами данных, загрузчиками данных, моделями.
- *Transformers* [56] — для легкого доступа к предварительно обученным моделям и краткой тонкой настройки.
- *nlp2* — работа с аргументами функций.
- *pickle* — сохранение и загрузка предварительно обработанных данных.
- *nlg-eval* — автоматические метрики для тестирования решений NLG (генерация естественного языка).
- *tensorboardX* — протоколирование и визуализация потерь и метрик.
- *tqdm* — прогресс-бар с подсчетом оставшегося времени.
- *sys*, *os*, *pathlib* — работа с файлами, каталогами и путями к ним.

[1] https://github.com/voidful/TFkit/
[2] https://github.com/voidful/BDG/

Весь код организован в виде *Python* скриптов (.py) с несколькими файлами: для обучения, оценки, вычисления метрик, потерь, оптимизации, загрузки данных и работы с моделями.

4.2 Трудности реализации

Пытаясь воспроизвести результаты и реализовать предложенный подход, мы столкнулись с некоторыми трудностями и препятствиями, которые мы описываем ниже в этом разделе.

4.2.1 Репозитории и версии

Первая проблема была связана с двумя разными репозиториями. В коде почти не было документации и комментариев. И назначение каждого репозитория, скрипта и функции было неясно. Поэтому мы были вынуждены самостоятельно выяснять назначение каждого компонента.

Кроме того, каждый репозиторий имеет несколько версий, и нет четкого указания на то, какая версия используется в экспериментах для исследования. Версии различались в процедурах работы с аргументами скриптов, обмена данными между различными устройствами, версиях используемых библиотек, структурах проекта, работе с длинными последовательностями и т.д. Таким образом, выбор правильной версии был очень важен.

Попытавшись воспроизвести результаты с различными версиями кода, мы не смогли точно воспроизвести результаты из статьи. Чтобы решить эту проблему, мы связались с авторами и попросили предоставить нам правильную версию кода.

4.2.2 Железо

Авторы оригинальной статьи [33] обучали модели на двух графических процессорах RTX Titan, каждый из которых имеет производительность 130 терафлопс. Понятно, что обычный ПК или ноутбук не имеет сопоставимых характеристик, и мы должны использовать специализированный сервер. У нас был доступ к трем различным серверам, но каждый из них имел свои ограничения. У нас был постоянный доступ к Google Colab Pro с его графическим процессором Nvidia P100 (21 терафлопс). К сожалению, сессия Colab аварийно завершалась каждые 12-24 часа из-за особенностей платформы. Также был университетский сервер с одним графическим процессором Nvidia V100 (100 терафлопс), но мы были ограничены несколькими неделями. И, наконец, у нас были ресурсы сервера DGX с восемью Nvidia V100 в течение нескольких дней.

Поскольку ресурсы GPU, оперативной памяти и времени были ограничены, мы не смогли запустить базовый подход на каждом сервере. Возникли три основные проблемы:

1. Объем оперативной памяти был недостаточен для предварительной обработки и кэширования данных;

2. Объем оперативной памяти и/или памяти GPU был недостаточен для одновременного хранения в памяти модели и пакета данных;

3. Базовые эксперименты с теми же настройками, что и в [33], потребовали много времени на выполнение (150 часов на эксперимент).

   Мы нашли различные варианты решения этой проблемы:

- Оптимизация кода: мы не нашли никаких улучшений, которые можно было бы сделать для заметной оптимизации скриптов.

- Уменьшение длины входной последовательности: как мы показали в разделе 3.2, уменьшение длины входной последовательности с 512 может привести к потере существенной информации для многих образцов.

- Обрезка набора данных и удаление некоторых образцов: это, скорее всего, ухудшит качество генерируемых дистракторов, так как модель будет обучаться на меньшем количестве образцов.

- Уменьшение количества эпох обучения: аналогично предыдущему пункту, способность модели в задаче DG ухудшится.

- Упрощение оптимизатора: Первоначально Google использовал Adam в качестве оптимизатора для BERT. Chung и др. **chung202020bertbased** использовал AdamW, оптимизатор Adam с модифицированным весовым распадом. Авторы BERT [11] утверждают, что переход на более простой оптимизатор может снизить потребление памяти, но в то же время это может повлиять на результаты. Мы решили последовать рекомендациям и использовать AdamW.

- Уменьшение размера пакета: это хорошая отправная точка, но она не помогает в случае, когда скрипт падает из-за нехватки оперативной и/или графической памяти во время предобработки (когда создание партии и обучение еще даже не началось).

- Изменение формы предобработки, кэширования и загрузки данных: изменение библиотек для загрузки и кэширования данных и формата файла для сохранения может заставить скрипт работать без сбоев и даже оптимизировать время работы.

- Изменение предварительно обученной модели. С некоторым компромиссом в отношении качества генерируемых дистракторов, это рассматривалось как один из лучших вариантов решения проблемы маломощного оборудования.

- Накопление градиентов. В этом подходе градиент рассчитывается за несколько предопределенных шагов и выполняет обратное распространение только после выполнения всех этих шагов. Эта техника применима в определенной степени, так как иногда ускоряет обучение, но требует дополнительной памяти для хранения вычисленных градиентов.

Поскольку только последние четыре варианта были более или менее подходящими, мы остановились на них. Глубокое их обсуждение представлено в следующем разделе, как и другие детали нашей реализации.

4.3 Наша реализация

4.3.1 Модель генератора

Наша реализация основана на реализации, описанной в разделе 4.1. Мы оставили структуру проекта как есть и внесли несколько изменений, направленных на решение описанных выше проблем (наборы данных и за-

грузчики данных, предварительно обученные трансформаторы), упрощение рабочего конвейера и реализацию предложенного подхода.

Мы поделились нашим кодом в репозитории в VCS[3].

Наборы данных и загрузчики данных

Базовый код кэшировал и загружал данные, используя бинарный формат *pickle*. Некоторые серверы были не в состоянии загрузить и кэшировать весь сплит поезда из-за недостатка оперативной памяти. Мы также попробовали сохранить кэшированные данные, используя форматы *jsonl* и *csv* (библиотеки *jsonlines* и *pandas* соответственно), они сохраняли данные в текстовом формате (не бинарном) и могли загрузить больше данных, но этого было недостаточно для загрузки данных всего тренировочного набора. И, наконец, мы реализовали функцию для предварительной обработки данных с помощью библиотеки *datasets* от HuggingFace. Это позволило нам полностью загрузить и кэшировать набор данных за относительно небольшое время.

Предобработанные трансформеры

Поскольку предварительно обученная модель *bert-base-cased*, используемая в оригинальной статье [33], потребляет ∼400 Mb памяти, и она медленна в обучении, мы решили рассмотреть родственные модели. Сравнение моделей показано в Таблица IV. Видно, что ALBERT и DistilBERT наиболее похожи на *bert-base* (с точки зрения LHA и, следовательно, мощности модели), поэтому они могут быть использованы в качестве замены в экспериментах. Мы также видим, что хотя *bert-tiny* сильно отличается от *bert-*

[3]https://github.com/russab0/distractor_generation

	Size	LHA	# params
bert-base-cased	415	L=12, H=768, A=12	110
albert-base-v2	45	L=12, H=768, A=12	11
distilbert-base-cased	251	L=12, H=768, A=12	66
prajjwal1/bert-medium	159	L=8, H=512, A=8	41
prajjwal1/bert-small	110	L=4, H=512, A=8	29
prajjwal1/bert-mini	43	L=4, H=256, A=4	11
prajjwal1/bert-tiny	17	L=2, H=128, A=2	4

Таблица IV: Сравнение семейства моделей BERT. Размер указан в мегабайтах, количество параметров — в миллионах. LHA означает основные характеристики модели: L = количество слоев, H = скрытый размер, A = количество головок внимания

base, они принадлежат к одному семейству, и *bert-tiny* можно использовать для тестирования кода и новых возможностей.

Логгирование

Авторы [33] использовали *tensorboardX* для регистрации метрик и потерь. Мы перешли на другое решение, *wandb*, которое имеет более продвинутую функциональность. Оно также поддерживает регистрацию и визуализацию метрик и потерь, но эта статистика и графики хранятся онлайн, чтобы можно было легко сравнивать прогоны с разных серверов. Кроме того, *wandb* хранит все конфигурации в едином формате, так что прогоны можно фильтровать и сравнивать. *Wandb* также подсчитывает статистику использования оборудования, что важно для корректировки конфигурации. И, наконец, *wandb* позволяет хранить дополнительные файлы онлайн, такие как веса и состояния модели.

Использование процессора

Чтобы быстро убедиться, что код работает после некоторых изменений, мы запустили его локально на машине с 2 Гб GPU. Этого оказалось недостаточно для работы даже с *bert-tiny*, и мы внедрили дополнительную функцию, чтобы заставить скрипт использовать CPU даже при наличии GPU.

Мы также связали среду Colab с Google Drive для хранения наборов данных и обученных моделей.

4.3.2 QA модель

Поскольку модель QA в архитектурах Генератор+QA и Генератор+QA+Discriminator определена как предварительно обученная, мы обучили ее заранее. Мы взяли скрипт из репозитория HuggingFace [56] для обучения трансформера QA на наборе данных SWAG и адаптировали его для другой структуры набора данных RACE. Мы взяли те же гиперпараметры, что и в [33]. В качестве базовых моделей мы использовали *bert-base-cased* и *distilbert-base-cased*. Максимальная длина установлена на 512. Размер пакета варьируется от 2 до 8 (в зависимости от возможностей оборудования). Мы использовали оптимизатор AdamW со скоростью обучения, равной 10^{-5}. Модель обучается в течение 10 эпох.

4.4 Постановка экспериментов

Эксперименты проводятся следующим образом: сначала запускается скрипт обучения (который сохраняет обученную модель в отдельном фай-

ле), а затем запускается скрипт тестирования на обученной модели.

Параметры обучения выбраны таким образом, чтобы они не отличались от экспериментов из базовой статьи [33] и были выполнимы на доступном оборудовании. Среди всех конфигураций BDG была обычная для ускорения экспериментов. Максимальная длина последовательности установлена равной 512. *Bert-base-cased* и *distilbert-base-cased* используются в основных экспериментах, а *prajjwal1/bert-tiny* используется только в небольших экспериментах для проверки выполнения кода. Мы использовали обновленный набор данных RACE: отдельно train split для обучения и test split для валидации и тестирования. Размер пакета настраивается в зависимости от мощности оборудования: он варьировался от 2 до 64 образцов на пакет. Количество эпох установлено на 6. Модель оптимизирует свои параметры с помощью AdamW со скоростью обучения $5 \cdot 10^{-5}$. Все эксперименты выполняются с использованием одного и того же случайного сида, равного 609.

Каждый эксперимент выполняется в специальной виртуальной среде Python 3.7 с предопределенными необходимыми библиотеками. Выполнения логируется локально и онлайн в *wandb*: логирование включает время выполнения, использование оборудования, потери, метрики и т.д.

4.5 Выводы

В этой главе мы описали базовую реализацию, нашу реализацию и представили постановку экспериментов. Результаты и их анализ представлены в главе 5.

Глава 5

Результаты и анализ

В этой главе мы представляем достигнутые результаты и подробно обсуждаем их.

Мы запланировали провести четыре эксперимента:

1. Базовый подход без модификаций (для воспроизведения результатов из базовой статьи);
2. Эксперимент с архитектурой Generator+QA.
3. Эксперимент с архитектурой Generator+Discriminator.
4. Эксперимент с архитектурой Generator+Discriminator+QA.

К сожалению, у нас не было достаточно времени и аппаратных ресурсов, поэтому мы запустили только первые два из них.

5.1 Результаты экспериментов

Поскольку модель QA в нашей методологии является предварительно обученной, мы обучили ее заранее и измерили точность, чтобы сравнить

Модель	Точность
bert-base-cased [33]	78
bert-base-cased	64
distilbert-base-cased	51

Таблица V: Результаты экспериментов над моделью QA, проведенных нами и Chung и др.

различные модели. Полученные результаты представлены в Таблица V. Как можно заметить, *distilbert-base-cased* значительно уступает *bert-base-cased*. Более того, между двумя экспериментами по *bert-base-cased* существует 14% разрыв в точности: это может быть связано с тем, что эксперимент из [33] использует накопление градиента за 2 шага, в то время как мы используем накопление градиента за 10 шагов. Поскольку мы применяем QA-модель для предсказания ответа из 3-4 вариантов, точности в 64% вполне достаточно. Таким образом, мы использовали BERT в наших экспериментах с архитектурой QA.

Результаты основных экспериментов по предложенным подходам представлены в Таблица VI. Видно, что наши результаты и результаты Chung и др. на *bert-base-cased* отличаются примерно на 3-6 пунктов, что может означать либо ошибку в одном из экспериментов, либо случайную вариативность генератора (случайные семена могли отличаться). Также очевидно, что использование менее мощной модели (*distilbert* вместо *bert*) приводит к вдвое худшим результатам. Тем не менее, *distilbert* можно использовать для оценки эффективности предложенного подхода по сравнению с базовым.

Наш первый эксперимент с предложенными методами был связан с архитектурой QA. В этом эксперименте модель QA предварительно обучается на наборе данных RACE, а основное обучение проводится таким

	BLEU 1	BLEU 2	BLEU 3	BLEU 4	ROUGE L
bert Baseline [33]	35.30	20.65	13.66	9.53	31.11
bert Baseline	29.32	16.72	10.48	6.99	25.42
distilbert Baseline	14.86	6.46	3.11	1.57	14.92
distilbert QA ($\lambda = 1$)	18.85	10.08	5.66	3.14	78.80
distilbert QA ($\lambda = 7$)	25.03	14.19	8.42	5.23	70.74
distilbert QA ($\lambda = 7$) fine-tune	25.86	12.80	6.81	3.74	21.39

Таблица VI: Результаты первичных экспериментов, проведенных нами и Chung и др. Модель показывает имя предварительно обученного трансформатора, архитектуру и дополнительный параметр λ. λ относится к параметру взвешивания потерь QA, введенному в (3.1)

образом, что модель Generator оптимизирует свои параметры, используя собственные потери и потери модели QA. Мы отметили, что предложенная архитектура QA значительно увеличивает автоматические метрики. Например, для эксперимента с $\lambda = 7$ BLEU-1 увеличился в 1,78 раза.

Мы также заметили, что внедренная архитектура QA увеличивает оценку ROUGE-L примерно в 5 раз. Мы посчитали это суперувеличение любопытным и решили изучить сгенерированные примеры вручную: оказалось, что большинство из них были пустыми или содержали одни и те же слова несколько раз: например, "погода погода погода погода погода". Мы поняли, что предложенный метод дает хорошие оценки BLEU и ROUGE, но плохие примеры. Результата генерации таких плохих примеров может быть связан с тем, что в начале обучения модель генератора не может создать ничего полезного, и эти бесполезные примеры используются моделью QA

для обновления весов модели генератора. После этого мы провели эксперимент, в котором сначала обучалась конфигурация Генератор в одиночку, а затем с совместными потерями QA-модели. В результате были получены лучшие оценки и выборки. Эксперименты четко показали, что высокое значение автоматических метрик не означает высокое качество генерируемых образцов. Это подтвердило наши опасения об отсутствии корреляции между автоматическими метриками и оценкой человека.

Из-за недостатка времени и ресурсов не все запланированные эксперименты были проведены и, таким образом, не все предложенные архитектуры были оценены. Это может стать направлением для дальнейших исследований.

5.2 Сравнение с существующими работами

В этом разделе мы сравниваем наши результаты с результатами других исследований задачи DG на наборе данных RACE: Chung и др. [33], Gao и др. [24], Zhou и др. [32]. Оценки BLEU и ROUGE вышеупомянутого и нашего подходов представлены в Таблица VII.

Несмотря на то, что подход QA превосходит базовую модель, он не может конкурировать с реальными исследовательскими подходами (разница составляет до 15 баллов). Мы предполагаем, что это связано с тем, что мы используем облегченный DistilBERT, который менее мощный, чем BERT, который обычно используется в исследованиях машинного обучения, включая [33]. Мы считаем достигнутые результаты удовлетворительными. Тем не менее, для полной уверенности нам необходимо использовать BERT и провести еще один раунд экспериментов.

	BLEU 1	**BLEU 2**	**BLEU 3**	**BLEU 4**	**ROUGE L**
BDG $_{\text{AN+PM}}$ [33]	39.52	24.29	17.28	13.28	33.40
BDG $_{\text{PM}}$ [33]	39.81	24.81	17.66	13.56	34.01
BDG [33]	35.30	20.65	13.66	9.53	31.11
DS-Att. [24]	27.32	14.69	9.29	6.47	15.12
CO-Att. [32]	28.65	15.15	9.77	7.01	15.39
QA	25.86	12.80	6.81	3.74	21.39

Таблица VII: Сравнение подходов на наборе данных RACE на основе метрик BLEU и ROUGE.

5.3 Выводы

В этой главе мы показали и интерпретировали достигнутые результаты, а также сравнили их с другими подходами. Мы удовлетворены достигнутыми результатами, но уверены, что есть много возможностей для улучшения.

Глава 6

Заключение

В данной работе мы изучили влияние GAN и совместного обучения моделей DG и QA на качество генерируемых дистракторов. Мы заметили, что наилучшие результаты (с точки зрения автоматических метрик и ручной оценки) достигаются при подходе, при котором Генератор и QA обучаются независимо, а затем совместно, так что Генератор оптимизирует свои параметры в соответствии с предсказаниями QA-модели.

Для оценки гипотезы и проведения экспериментов мы использовали обновленный набор данных RACE. Мы использовали подход BDG [33] в качестве базового, модель BDG рассматривается как Генератор. Наш ключевой вклад в эту область заключается в том, что мы ввели две модели, которые могут сделать генерируемые дистракторы более сложными и правдоподобными. Первая модель, QA-модель (модель ответа на вопрос), берет контекст, вопрос и набор вариантов и определяет, какой из вариантов является правильным ответом. Вторая модель, Дискриминатор (от GANs), берет элемент теста (включая набор дистракторов) и определяет, был ли элемент теста взят из реальных данных или сгенерирован моделью Гене-

ратора.

Эксперименты показали, что автоматические метрики не всегда коррелируют с человеческими оценками. Наилучшие результаты показал подход Генератор+QA с конфигурацией, в которой и Генератор, и QA были предварительно обучены, а затем Генератор был тонко настроен с использованием потерь модели QA.

Важно отметить, что, несмотря на то, что подход QA не дотягивает до существующих научных подходов, он имеет шанс превзойти их в будущих исследованиях: путем нахождения наилучшей комбинации предварительно обученной модели трансформатора (например, BERT, BART, GPT), конфигурации BDG (простой, AN, AN+PM) и весовых гиперпараметров. Будущие исследования также могут быть посвящены реализации и исследованию архитектуры Генератор+Дискриминатор+QA.

Список литературы

[1] R. P. McDonald, *Test theory: A unified treatment*. psychology press, 2013.

[2] T. Zesch и O. Melamud, «Automatic generation of challenging distractors using context-sensitive inference rules,» в *Proceedings of the Ninth Workshop on Innovative Use of NLP for Building Educational Applications*, 2014, с. 143—148.

[3] S. Rakangor и Y. Ghodasara, «Literature review of automatic question generation systems,» *International journal of scientific and research publications*, т. 5, № 1, с. 1—5, 2015.

[4] K. Stasaski и M. A. Hearst, «Multiple Choice Question Generation Utilizing An Ontology,» в *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, Copenhagen, Denmark: Association for Computational Linguistics, сент. 2017, с. 303—312. DOI: 10.18653/v1/W17-5034. url: https://www.aclweb.org/anthology/W17-5034.

[5] J. Lu, X. Ye, Y. Ren и Y. Yang, *Good, Better, Best: Textual Distractors Generation for Multi-Choice VQA via Policy Gradient*, 2019. arXiv: 1910.09134 [cs.CV].

[6] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk и Y. Bengio, «Learning phrase representations using RNN encoder-decoder for statistical machine translation,» *arXiv preprint arXiv:1406.1078*, 2014.

[7] S. Hochreiter и J. Schmidhuber, «Long Short-Term Memory,» *Neural Computation*, т. 9, № 8, с. 1735—1780, 1997. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://doi.org/10.1162/neco.1997.9.8.1735. url: https://doi.org/10.1162/neco.1997.9.8.1735.

[8] G. Chevalier. (2018). «The LSTM cell.» Accessed on 18.10.2020, url: https://commons.wikimedia.org/wiki/File:The_LSTM_cell.png.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser и I. Polosukhin, *Attention Is All You Need*, 2017. arXiv: 1706.03762 [cs.CL].

[10] J. Uszkoreit, «Transformer: A novel neural network architecture for language understanding,» *Google AI Blog*, т. 31, 2017.

[11] J. Devlin, M.-W. Chang, K. Lee и K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, 2019. arXiv: 1810.04805 [cs.CL].

[12] D. E. Rumelhart, G. E. Hinton и R. J. Williams, «Learning Internal Representations by Error Propagation,» в *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. Cambridge, MA, USA: MIT Press, 1986, с. 318—362, ISBN: 026268053X.

[13] D. Bank, N. Koenigstein и R. Giryes, «Autoencoders,» *arXiv preprint arXiv:2003.05991*, 2020.

[14] (2016). «Under the Hood of the Variational Autoencoder (in Prose and Code).» Accessed on 23.10.2020, url: https://blog.fastforwardlabs.com/2016/08/22/under-the-hood-of-the-variational-autoencoder-in-prose-and-code.html.

[15] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville и Y. Bengio, *Generative Adversarial Networks*, 2014. arXiv: 1406.2661 [stat.ML].

[16] A. Gharakhanian. (2016). «Generative Adversarial Networks – Hot Topic in Machine Learning.» Accessed on 22.10.2020, url: https://www.linkedin.com/pulse/gans-one-hottest-topics-machine-learning-al-gharakhanian.

[17] R. Mitkov и H. Le An, «Computer-aided generation of multiple-choice tests,» в *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing*, 2003, с. 17—22.

[18] R. Mitkov, H. Le An и N. Karamanis, «A computer-aided environment for generating multiple-choice test items,» *Natural language engineering*, т. 12, № 2, с. 177, 2006.

[19] M. Heilman, «Automatic factual question generation from text,» дис. ... док., Ph. D. thesis, Carnegie Mellon University, 2011.

[20] I. V. Serban, A. Garcia-Duran, C. Gulcehre, S. Ahn, S. Chandar, A. Courville и Y. Bengio, «Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus,» *arXiv preprint arXiv:1603.06807*, 2016.

[21] U. Jain, Z. Zhang и A. G. Schwing, «Creativity: Generating Diverse Questions Using Variational Autoencoders,» в *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition (CVPR), июль 2017.

[22] C. Liang, X. Yang, N. Dave, D. Wham, B. Pursel и C. L. Giles, «Distractor Generation for Multiple Choice Questions Using Learning to Rank,» в *Proceedings of the 13th Workshop on Innovative Use of NLP for Building Educational Applications, BEA@NAACL*, ACL, 2018, с. 284—290.

[23] J. Lee и S. Seneff, «Automatic generation of cloze items for prepositions,» в *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[24] Y. Gao, L. Bing, P. Li, I. King и M. R. Lyu, «Generating Distractors for Reading Comprehension Questions from Real Examinations,» *CoRR*, т. abs/1809.02768, 2018. arXiv: 1809.02768. url: http://arxiv.org/abs/1809.02768.

[25] H. C. Goodrich, «Distractor efficiency in foreign language testing,» *Tesol Quarterly*, с. 69—78, 1977.

[26] R. P. d. S. Correia, J. Baptista, N. Mamede, I. Trancoso и M. Eskenazi, «Automatic generation of cloze question distractors,» в *Second language studies: acquisition, learning, education and technology*, 2010.

[27] A. Graesser и R. Wisher, «Question generation as a learning multiplier in distributed learning environments. United States Army Research Institute for the Behavioral and Social Sciences,» Technical Report 1121, тех. отч., 2001.

[28] J. Pino и M. Eskenazi, «Semi-automatic generation of cloze question distractors effect of students' l1,» в *International Workshop on Speech and Language Technology in Education*, 2009.

[29] L. Breiman, «Random forests,» *Machine learning*, т. 45, № 1, с. 5—32, 2001.

[30] C. J. Burges, «From RankNet to LambdaRank to LambdaMART: An Overview,» *Learning*, т. 11, № 23-581, с. 81, 2010.

[31] J. Wang, L. Yu, W. Zhang, Y. Gong, Y. Xu, B. Wang, P. Zhang и D. Zhang, «IRGAN,» *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, авг. 2017. DOI: 10.1145/3077136.3080786. url: http://dx.doi.org/10.1145/3077136.3080786.

[32] X. Zhou, S. Luo и Y. Wu, «Co-Attention Hierarchical Network: Generating Coherent Long Distractors for Reading Comprehension,» *arXiv preprint arXiv:1911.08648*, 2019.

[33] H.-L. Chung, Y.-H. Chan и Y.-C. Fan, *A BERT-based Distractor Generation Scheme with Multi-tasking and Negative Answer Training Strategies*, 2020. arXiv: 2010.05384 [cs.CL].

[34] J. Offerijns, S. Verberne и T. Verhoef, *Better Distractions: Transformer-based Distractor Generation and Multiple Choice Question Filtering*, 2020. arXiv: 2010.09598 [cs.CL].

[35] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong и R. Socher, *CTRL: A Conditional Transformer Language Model for Controllable Generation*, 2019. arXiv: 1909.05858 [cs.CL].

[36] I. Aldabe и M. Maritxalar, «Automatic distractor generation for domain specific texts,» в *International Conference on Natural Language Processing*, Springer, 2010, с. 27—38.

[37] S. C. Deerwester, S. T. Dumais, G. W. Furnas, R. A. Harshman, T. K. Landauer, K. E. Lochbaum и L. A. Streeter, *Computer information retrieval using latent semantic structure*, US Patent 4,839,853, июнь 1989.

[38] E. Sumita, F. Sugaya и S. Yamamoto, «Measuring non-native speakers' proficiency of english by using a test with automatically-generated fill-in-the-blank questions,» в *Proceedings of the second workshop on Building Educational Applications Using NLP*, 2005, с. 61—68. url: https://www.aclweb.org/anthology/W05-0210.pdf.

[39] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman и P. Blunsom, «Teaching machines to read and comprehend,» в *Advances in neural information processing systems*, 2015, с. 1693—1701.

[40] F. Hill, A. Bordes, S. Chopra и J. Weston, «The goldilocks principle: Reading children's books with explicit memory representations,» *arXiv preprint arXiv:1511.02301*, 2015.

[41] O. Bajgar, R. Kadlec и J. Kleindienst, «Embracing data abundance: Booktest dataset for reading comprehension,» *arXiv preprint arXiv:1610.00956*, 2016.

[42] T. Onishi, H. Wang, M. Bansal, K. Gimpel и D. McAllester, «Who did what: A large-scale person-centered cloze dataset,» *arXiv preprint arXiv:1608.05457*, 2016.

[43] P. Rajpurkar, J. Zhang, K. Lopyrev и P. Liang, «SQuAD: 100,000+ questions for machine comprehension of text,» *arXiv preprint arXiv:1606.05250*, 2016.

[44] P. Rajpurkar, R. Jia и P. Liang, *Know What You Don't Know: Unanswerable Questions for SQuAD*, 2018. arXiv: 1806.03822 [cs.CL].

[45] P. Efimov, A. Chertok, L. Boytsov и P. Braslavski, «SberQuAD — russian reading comprehension dataset: Description and analysis,» в *International Conference of the Cross-Language Evaluation Forum for European Languages*, Springer, 2020, с. 3—15.

[46] A. Trischler, T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman и K. Suleman, «NewsQA: A machine comprehension dataset,» *arXiv preprint arXiv:1611.09830*, 2016.

[47] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder и L. Deng, «MS MARCO: A human-generated machine reading comprehension dataset,» 2016.

[48] M. Joshi, E. Choi, D. S. Weld и L. Zettlemoyer, «TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension,» *arXiv preprint arXiv:1705.03551*, 2017.

[49] D. Khashabi, T. Khot, A. Sabharwal, P. Clark, O. Etzioni и D. Roth, «Question answering via integer programming over semi-structured knowledge,» *arXiv preprint arXiv:1604.06076*, 2016.

[50] H. Shibuki, K. Sakamoto, Y. Kano, T. Mitamura, M. Ishioroshi, K. Y. Itakura, D. Wang, T. Mori и N. Kando, «Overview of the NTCIR-11 QA-Lab Task.,» в *Ntcir*, 2014.

[51] A. Peñas, C. Unger и A.-C. N. Ngomo, «Overview of CLEF question answering track 2014,» в *International Conference of the Cross-Language Evaluation Forum for European Languages*, Springer, 2014, с. 300—306.

[52] G. Lai, Q. Xie, H. Liu, Y. Yang и E. Hovy, «RACE: Large-scale reading comprehension dataset from examinations,» *arXiv preprint arXiv:1704.04683*, 2017.

[53] M. Richardson, C. J. Burges и E. Renshaw, «MCTest: A challenge dataset for the open-domain machine comprehension of text,» в *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, с. 193—203.

[54] K. Papineni, S. Roukos, T. Ward и W.-j. Zhu, «BLEU: a Method for Automatic Evaluation of Machine Translation,» 2002, с. 311—318.

[55] C.-Y. Lin, «ROUGE: A package for automatic evaluation of summaries,» в *Text summarization branches out*, 2004, с. 74—81.

[56] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest и A. M. Rush, *HuggingFace's Transformers: State-of-the-art Natural Language Processing*, 2019. arXiv: 1910.03771 [cs.CL].

Приложение А

Примеры сгененированных дистраторов

В этом приложении мы приводим примеры дистракторов, сгенерированных различными моделями для образцов из набора данных RACE. Эти примеры могут быть полезны для оценки сгенерированных дистракторов человеком. Приложение структурировано таким образом, что каждый раздел объединяет примеры одной и той же модели. Каждый подраздел внутри раздела представляет собой отдельный образец. Каждый образец состоит из параграфа, вопроса, правильного ответа и двух списков дистракторов: результаты, полученные человеком (истина из набора данных) и результаты, полученные моделью (сгенерированные моделью). Мы также представляем рассчитанные оценки BLEU и ROUGE для сгенерированных дистракторов.

Поскольку набор данных велик как по количеству образцов, так и по размеру выборки, мы представили только несколько образцов. Чтобы выбрать репрезентативные образцы, мы применили решение G+QA ко все-

му тестовому набору и рассчитали оценки BLEU и ROUGE. После этого мы вычислили средний балл и отсортировали его в порядке убывания. И, наконец, мы извлекаем образцы на каждой 1000-й позиции и фиксируем эти образцы по их уникальным идентификаторам. По этим уникальным идентификаторам отбираются примеры для других подходов. Последний раздел посвящен сравнению сгенерированных дистракторов разными моделями.

A.1 Базовое решение DistilBERT

A.1.1

Абзац: New Zealand is in the Pacific Ocean and it's made up of two islands : the North Island and the South Island. About a thousand years ago, the Maori people came from the islands of the Pacific Ocean to New Zealand and made their home. Since then, people have come from all over the world to live in New Zealand. New Zealand has three official languages : English, Maori and New Zealand Sign Language. Many places in New Zealand have Maori names. "Kia ora "is a Maori language greeting. In 1865, Wellington took Auckland's place and became the capital ( ) of New Zealand, but the largest city is Auckland. Both cities are on the North Island. New Zealand has high mountains, active volcanoes, lakes, rainforests and beautiful sandy beaches, which made it a location for the movie The Lord of the Rings. New Zealand has mild ( ) temperatures and lots of sunshine. January and February are the warmest months of the year, while July is the coldest. New Zealand has many special birds. The best known is the kiwi. The kiwi, about the size of a large chicken, can't fly. It is the symbol of the country and you can find pictures of kiwis on New Zealand stamps and coins. The kiwi is named after its calls - - kewe, kewe. New Zealanders are also called Kiwis, but very. few New Zealanders have ever seen a kiwi bird in the wild! New Zealand is also famous for its sheep. There are millions of sheep living in farms all over New Zealand.,.

Вопрос: We can learn from the passage that Wellington

Ответ: is the capital city of New Zealand

Результаты модели	**Истинное значение (из датасета)**
- the	- is on the South Island - is the largest city in New Zealand

Оценки: $BLEU_1 : 1.0, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.3$

A.1.2

Абзац: Healthy relationships are fun and make you feel good about yourself. The relationships that you make in your teenage years will be a special pan of your life. They will teach you some of the most important lessons about who you are. This may help you understand different kinds of relationships, what makes each relationship special, and how to communicate in a positive way. What makes a relationship healthy? Communication and Sharing : The most important part of any healthy relationship between two people is being able to talk and listen to each other. You and the other person can find out what your common interests are. You can share your feelings with the other person and trust that he or she will be there to listen to you and support you. In healthy relationships, people don't lie. Communication is based on honesty and trust. By listening carefully and sharing your thoughts and feelings with other people, you show them that they play an important part in your life. Respect and Trust : In healthy relationships, you learn to respect and trust important people in your life. Disagreements may still happen, but you learn to stay calm and talk about how you feel. Talking calmly helps you to understand the real reason for not getting along. It makes it much easier to figure out how to fix it. In healthy relationships, working through disagreements often makes the relationship stronger. In healthy relationships, people respect each other for who they are. This includes respecting and listening to yourself and your feelings so you can set boundaries and feel comfortable. You will find that you learn to understand experiences and feelings of others as well as having them understand your experiences and feelings. How do I know that I have a healthy relationship with someone? You know that you are in a healthy relationship with someone because you feel good about yourself when you are around that person. Unhealthy relationships can make you feel sad, angry, scared, or worried. Healthy peer relationships involve an equal amount of giving and taking in the relationship. In unhealthy relationships, there is an unfair balance. You may often feel that you are giving the other person more attention than that he or she gives to you. You should feel safe around the other person and feel that you can trust him or her with your secrets. In a healthy relationship, you like to

Вопрос: Which of the following relationships is healthy?

Ответ: You like to talk to someone and listen to him or her.

Результаты модели	Истинное значение (из датасета)
- your you how what	- You feel upset when you are around someone. - You feel that you have to spend time with him or her.

Оценки: $BLEU_1 : 0.25, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.16$

A.1.3

Абзац: There are many wetlands in China and some of them have become the world's important wetlands. The Chinese Yellow Sea Wetlands are among them. They are in Yancheng, Jiangsu Province. They are home of many different kinds of birds and animals. The world's largest Milu Deer Nature Reserve is in them. More than 700 milu deer live freely there. The temperature in the wetlands is usually neither too high nor too low. There is a lot of rain and sunshine, too. They are really good places for wildlife. Offering food and home for some special kinds of animals and birds is not the only reason why we need to protect wetlands. Wetlands are important because they also prevent flood. But some people want to change the wetlands to make more space for farms and buildings. This means there will be less and less space for wildlife. Luckily, more and more people are beginning to realize the important of wetlands and wildlife. Every year, on February 2nd many activities are held to tell people more about wetlands around the world.

Вопрос: The reasons for protecting wetlands are the following except that

Ответ: they will be good farmland

Результаты модели	Истинное значение (из датасета)
- they will be a good place for animals	- they are home of wildlife - they offer food to the wildlife

Оценки: $BLEU_1 : 0.12, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.16$

A.1.4

Абзац: A small boy sat on the street with a hat by his feet. He held up a sign which said : "I am blind, please help ". There were only a few coins in the hat. A man was walking by. He took a few coins from his pocket and dropped them into the hat. He then took the sign, turned it around, and wrote some words. He put the sign back so that everyone who walked by would see the new words. Soon the hat began to fill up. A lot more people were giving money to the blind boy. That afternoon the man who had changed the sign came to see how things were. The boy recognized his footsteps and asked : "Were you the one who changed my sign this morning? What did you write? "The man said : "I only wrote the truth. I said what you said but in a different way. "What he had written was : "Today is a beautiful day but I can not see it. "Do you think the first sign and the second sign were saying the same thing? Of course both signs told people the boy was blind. But the first sign simply told people to help by putting some money in the hat. The second sign told people that they were able to enjoy the day, but the boy could not enjoy it because he was blind. The first sign simply said the boy was blind. The second sign told people they were so lucky that they were not blind. There are at least two lessons we can learn from this simple story. The first is : Be thankful for what you have. Someone else has less. Help where you can. The second is : Be creative. Think differently. There is always a better way!

Вопрос: What was the boy doing on the street?

Ответ: He was begging for money.

Результаты модели	**Истинное значение (из датасета)**
- sign the sign	- He was selling his old hat.

Оценки: $BLEU_1 : 0.0, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.0$

A.1.5

Абзац: In a class this past December, after I wrote some directions on the board for students about their final examination, one young woman quickly took a picture of the board using her smart phone. When I looked in her direction, she apologized : "Sorry. Was it wrong to take a picture? I can't read my own handwriting, "the young woman explained. "It's best if I take a picture of your writing so I can understand the notes. "That remark started a class - wide conversation about taking a picture instead of taking notes. For those in the photo - taking camp, motivations extended beyond their inability to comprehend their own handwriting. Some took pictures of notes because they knew their phone was a safe place to store material. They might lose paper, they reasoned, but they wouldn't lose their phones. Some took photos because they wanted to record exactly the manner in which I had noted information on the board. Others told me that during class they liked to be able to listen to the discussion attentively. Yet the use of cameras as note takers, though it may be convenient, does raise significant questions for the classroom. Is a picture an effective replacement for the process of note - taking? Instructors encourage students to take notes because the act of doing so is more than merely recording necessary information - - it helps prepare the way for understanding. Encouraging students to take notes may be an old - fashioned instructional method, but just because a method has a long history doesn't mean it's out of date. Writing things down engages a student's brain in listening, visual, and kinesthetic learning - - a view supported by a longstanding research. The act of writing down information enables a person to begin committing it to memory, and to process and combine it, establishing the building blocks of learning new concepts. Taking a picture does indeed record the information, but it deletes some of the necessary mental engagement that taking notes employs. So can the two be equally effective? I'm not sure how to measure the effectiveness of either method. For now, I allow students to take notes however they see fit - - handwritten or photographed - - because I figure that some notes, no matter the method of note - taking, are better than none.

Вопрос: Students refuse to take notes by hand because

Ответ: they are more likely to lose notes

Результаты модели	Истинное значение (из датасета)
- how your s a Students students	- they are unable to take notes

Оценки: $BLEU_1 : 0.0, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.0$

A.2 Базовое решение BERT

A.2.1

Абзац: New Zealand is in the Pacific Ocean and it's made up of two islands : the North Island and the South Island. About a thousand years ago, the Maori people came from the islands of the Pacific Ocean to New Zealand and made their home. Since then, people have come from all over the world to live in New Zealand. New Zealand has three official languages : English, Maori and New Zealand Sign Language. Many places in New Zealand have Maori names. "Kia ora "is a Maori language greeting. In 1865, Wellington took Auckland's place and became the capital ( ) of New Zealand, but the largest city is Auckland. Both cities are on the North Island. New Zealand has high mountains, active volcanoes, lakes, rainforests and beautiful sandy beaches, which made it a location for the movie The Lord of the Rings. New Zealand has mild ( ) temperatures and lots of sunshine. January and February are the warmest months of the year, while July is the coldest. New Zealand has many special birds. The best known is the kiwi. The kiwi, about the size of a large chicken, can't fly. It is the symbol of the country and you can find pictures of kiwis on New Zealand stamps and coins. The kiwi is named after its calls - - kewe, kewe. New Zealanders are also called Kiwis, but very. few New Zealanders have ever seen a kiwi bird in the wild! New Zealand is also famous for its sheep. There are millions of sheep living in farms all over New Zealand.,.

Вопрос: We can learn from the passage that Wellington

Ответ: is the capital city of New Zealand

Результаты модели	Истинное значение (из датасета)
- the name of New Zealand	- is on the South Island - is the largest city in New Zealand

Оценки: $BLEU_1 : 0.6, BLEU_2 : 0.39, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.49$

A.2.2

Абзац: Healthy relationships are fun and make you feel good about yourself. The relationships that you make in your teenage years will be a special pan of your life. They will teach you some of the most important lessons about who you are. This may help you understand different kinds of relationships, what makes each relationship special, and how to communicate in a positive way. What makes a relationship healthy? Communication and Sharing : The most important part of any healthy relationship between two people is being able to talk and listen to each other. You and the other person can find out what your common interests are. You can share your feelings with the other person and trust that he or she will be there to listen to you and support you. In healthy relationships, people don't lie. Communication is based on honesty and trust. By listening carefully and sharing your thoughts and feelings with other people, you show them that they play an important part in your life. Respect and Trust : In healthy relationships, you learn to respect and trust important people in your life. Disagreements may still happen, but you learn to stay calm and talk about how you feel. Talking calmly helps you to understand the real reason for not getting along. It makes it much easier to figure out how to fix it. In healthy relationships, working through disagreements often makes the relationship stronger. In healthy relationships, people respect each other for who they are. This includes respecting and listening to yourself and your feelings so you can set boundaries and feel comfortable. You will find that you learn to understand experiences and feelings of others as well as having them understand your experiences and feelings. How do I know that I have a healthy relationship with someone? You know that you are in a healthy relationship with someone because you feel good about yourself when you are around that person. Unhealthy relationships can make you feel sad, angry, scared, or worried. Healthy peer relationships involve an equal amount of giving and taking in the relationship. In unhealthy relationships, there is an unfair balance. You may often feel that you are giving the other person more attention than that he or she gives to you. You should feel safe around the other person and feel that you can trust him or her with your secrets. In a healthy relationship, you like to

Вопрос: Which of the following relationships is healthy?

Ответ: You like to talk to someone and listen to him or her.

Результаты модели	Истинное значение (из датасета)
- nan	- You feel upset when you are around someone. - You feel that you have to spend time with him or her.

Оценки: $BLEU_1 : 0.0, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 1.0$

A.2.3

Абзац: There are many wetlands in China and some of them have become the world's important wetlands. The Chinese Yellow Sea Wetlands are among them. They are in Yancheng, Jiangsu Province. They are home of many different kinds of birds and animals. The world's largest Milu Deer Nature Reserve is in them. More than 700 milu deer live freely there. The temperature in the wetlands is usually neither too high nor too low. There is a lot of rain and sunshine, too. They are really good places for wildlife. Offering food and home for some special kinds of animals and birds is not the only reason why we need to protect wetlands. Wetlands are important because they also prevent flood. But some people want to change the wetlands to make more space for farms and buildings. This means there will be less and less space for wildlife. Luckily, more and more people are beginning to realize the important of wetlands and wildlife. Every year, on February 2nd many activities are held to tell people more about wetlands around the world.

Вопрос: The reasons for protecting wetlands are the following except that

Ответ: they will be good farmland

Результаты модели	Истинное значение (из датасета)
- it is easy to keep the habitats in good weather	- they are home of wildlife - they offer food to the wildlife

Оценки: $BLEU_1 : 0.2, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.26$

A.2.4

Абзац: A small boy sat on the street with a hat by his feet. He held up a sign which said : "I am blind, please help ". There were only a few coins in the hat. A man was walking by. He took a few coins from his pocket and dropped them into the hat. He then took the sign, turned it around, and wrote some words. He put the sign back so that everyone who walked by would see the new words. Soon the hat began to fill up. A lot more people were giving money to the blind boy. That afternoon the man who had changed the sign came to see how things were. The boy recognized his footsteps and asked : "Were you the one who changed my sign this morning? What did you write? "The man said : "I only wrote the truth. I said what you said but in a different way. "What he had written was : "Today is a beautiful day but I can not see it. "Do you think the first sign and the second sign were saying the same thing? Of course both signs told people the boy was blind. But the first sign simply told people to help by putting some money in the hat. The second sign told people that they were able to enjoy the day, but the boy could not enjoy it because he was blind. The first sign simply said the boy was blind. The second sign told people they were so lucky that they were not blind. There are at least two lessons we can learn from this simple story. The first is : Be thankful for what you have. Someone else has less. Help where you can. The second is : Be creative. Think differently. There is always a better way!

Вопрос: What was the boy doing on the street?

Ответ: He was begging for money.

Результаты модели	Истинное значение (из датасета)
- He asked people to the blind boy.	- He was selling his old hat.

Оценки: $BLEU_1 : 0.14, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.16$

A.2.5

Абзац: In a class this past December, after I wrote some directions on the board for students about their final examination, one young woman quickly took a picture of the board using her smart phone. When I looked in her direction, she apologized : "Sorry. Was it wrong to take a picture? I can't read my own handwriting, "the young woman explained. "It's best if I take a picture of your writing so I can understand the notes. "That remark started a class - wide conversation about taking a picture instead of taking notes. For those in the photo - taking camp, motivations extended beyond their inability to comprehend their own handwriting. Some took pictures of notes because they knew their phone was a safe place to store material. They might lose paper, they reasoned, but they wouldn't lose their phones. Some took photos because they wanted to record exactly the manner in which I had noted information on the board. Others told me that during class they liked to be able to listen to the discussion attentively. Yet the use of cameras as note takers, though it may be convenient, does raise significant questions for the classroom. Is a picture an effective replacement for the process of note - taking? Instructors encourage students to take notes because the act of doing so is more than merely recording necessary information - - it helps prepare the way for understanding. Encouraging students to take notes may be an old - fashioned instructional method, but just because a method has a long history doesn't mean it's out of date. Writing things down engages a student's brain in listening, visual, and kinesthetic learning - - a view supported by a longstanding research. The act of writing down information enables a person to begin committing it to memory, and to process and combine it, establishing the building blocks of learning new concepts. Taking a picture does indeed record the information, but it deletes some of the necessary mental engagement that taking notes employs. So can the two be equally effective? I'm not sure how to measure the effectiveness of either method. For now, I allow students to take notes however they see fit - - handwritten or photographed - - because I figure that some notes, no matter the method of note - taking, are better than none.

Вопрос: Students refuse to take notes by hand because

Ответ: they are more likely to lose notes

Результаты модели	**Истинное значение (из датасета)**
- taking	- they are unable to take notes

Оценки: $BLEU_1 : 0.0, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.0$

A.3 Архитектура G+QA

6 эпох простого Генератора и 1 эпоха совместного G+QA

A.3.1

Абзац: New Zealand is in the Pacific Ocean and it's made up of two islands : the North Island and the South Island. About a thousand years ago, the Maori people came from the islands of the Pacific Ocean to New Zealand and made their home. Since then, people have come from all over the world to live in New Zealand. New Zealand has three official languages : English, Maori and New Zealand Sign Language. Many places in New Zealand have Maori names. "Kia ora "is a Maori language greeting. In 1865, Wellington took Auckland's place and became the capital ( ) of New Zealand, but the largest city is Auckland. Both cities are on the North Island. New Zealand has high mountains, active volcanoes, lakes, rainforests and beautiful sandy beaches, which made it a location for the movie The Lord of the Rings. New Zealand has mild ( ) temperatures and lots of sunshine. January and February are the warmest months of the year, while July is the coldest. New Zealand has many special birds. The best known is the kiwi. The kiwi, about the size of a large chicken, can't fly. It is the symbol of the country and you can find pictures of kiwis on New Zealand stamps and coins. The kiwi is named after its calls - - kewe, kewe. New Zealanders are also called Kiwis, but very. few New Zealanders have ever seen a kiwi bird in the wild! New Zealand is also famous for its sheep. There are millions of sheep living in farms all over New Zealand.,.

Вопрос: We can learn from the passage that Wellington

Ответ: is the capital city of New Zealand

Результаты модели	Истинное значение (из датасета)
- is the largest city in New Zealand	- is on the South Island - is the largest city in New Zealand

Оценки: $BLEU_1 : 1.0, BLEU_2 : 1.0, BLEU_3 : 1.0, BLEU_4 : 1.0, ROUGE_L : 1.0$

A.3.2

Абзац: Healthy relationships are fun and make you feel good about yourself. The relationships that you make in your teenage years will be a special pan of your life. They will teach you some of the most important lessons about who you are. This may help you understand different kinds of relationships, what makes each relationship special, and how to communicate in a positive way. What makes a relationship healthy? Communication and Sharing : The most important part of any healthy relationship between two people is being able to talk and listen to each other. You and the other person can find out what your common interests are. You can share your feelings with the other person and trust that he or she will be there to listen to you and support you. In healthy relationships, people don't lie. Communication is based on honesty and trust. By listening carefully and sharing your thoughts and feelings with other people, you show them that they play an important part in your life. Respect and Trust : In healthy relationships, you learn to respect and trust important people in your life. Disagreements may still happen, but you learn to stay calm and talk about how you feel. Talking calmly helps you to understand the real reason for not getting along. It makes it much easier to figure out how to fix it. In healthy relationships, working through disagreements often makes the relationship stronger. In healthy relationships, people respect each other for who they are. This includes respecting and listening to yourself and your feelings so you can set boundaries and feel comfortable. You will find that you learn to understand experiences and feelings of others as well as having them understand your experiences and feelings. How do I know that I have a healthy relationship with someone? You know that you are in a healthy relationship with someone because you feel good about yourself when you are around that person. Unhealthy relationships can make you feel sad, angry, scared, or worried. Healthy peer relationships involve an equal amount of giving and taking in the relationship. In unhealthy relationships, there is an unfair balance. You may often feel that you are giving the other person more attention than that he or she gives to you. You should feel safe around the other person and feel that you can trust him or her with your secrets. In a healthy relationship, you like to

Вопрос: Which of the following relationships is healthy?

Ответ: You like to talk to someone and listen to him or her.

Результаты модели	Истинное значение (из датасета)
- You don't love the other person.	- You feel upset when you are around someone. - You feel that you have to spend time with him or her.

Оценки: $BLEU_1 : 0.12, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.14$

A.3.3

Абзац: There are many wetlands in China and some of them have become the world's important wetlands. The Chinese Yellow Sea Wetlands are among them. They are in Yancheng, Jiangsu Province. They are home of many different kinds of birds and animals. The world's largest Milu Deer Nature Reserve is in them. More than 700 milu deer live freely there. The temperature in the wetlands is usually neither too high nor too low. There is a lot of rain and sunshine, too. They are really good places for wildlife. Offering food and home for some special kinds of animals and birds is not the only reason why we need to protect wetlands. Wetlands are important because they also prevent flood. But some people want to change the wetlands to make more space for farms and buildings. This means there will be less and less space for wildlife. Luckily, more and more people are beginning to realize the important of wetlands and wildlife. Every year, on February 2nd many activities are held to tell people more about wetlands around the world.

Вопрос: The reasons for protecting wetlands are the following except that

Ответ: they will be good farmland

Результаты модели	Истинное значение (из датасета)
- they will be the world's biggest the world's biggest in the world	- they are home of wildlife - they offer food to the wildlife

Оценки: $BLEU_1 : 0.17, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.24$

A.3.4

Абзац: A small boy sat on the street with a hat by his feet. He held up a sign which said : "I am blind, please help ". There were only a few coins in the hat. A man was walking by. He took a few coins from his pocket and dropped them into the hat. He then took the sign, turned it around, and wrote some words. He put the sign back so that everyone who walked by would see the new words. Soon the hat began to fill up. A lot more people were giving money to the blind boy. That afternoon the man who had changed the sign came to see how things were. The boy recognized his footsteps and asked : "Were you the one who changed my sign this morning? What did you write? "The man said : "I only wrote the truth. I said what you said but in a different way. "What he had written was : "Today is a beautiful day but I can not see it. "Do you think the first sign and the second sign were saying the same thing? Of course both signs told people the boy was blind. But the first sign simply told people to help by putting some money in the hat. The second sign told people that they were able to enjoy the day, but the boy could not enjoy it because he was blind. The first sign simply said the boy was blind. The second sign told people they were so lucky that they were not blind. There are at least two lessons we can learn from this simple story. The first is : Be thankful for what you have. Someone else has less. Help where you can. The second is : Be creative. Think differently. There is always a better way!

Вопрос: What was the boy doing on the street?

Ответ: He was begging for money.

Результаты модели	Истинное значение (из датасета)
- He was looking for money.	- He was selling his old hat.

Оценки: $BLEU_1 : 0.33, BLEU_2 : 0.26, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.36$

A.3.5

Абзац: In a class this past December, after I wrote some directions on the board for students about their final examination, one young woman quickly took a picture of the board using her smart phone. When I looked in her direction, she apologized : "Sorry. Was it wrong to take a picture? I can't read my own handwriting, "the young woman explained. "It's best if I take a picture of your writing so I can understand the notes. "That remark started a class - wide conversation about taking a picture instead of taking notes. For those in the photo - taking camp, motivations extended beyond their inability to comprehend their own handwriting. Some took pictures of notes because they knew their phone was a safe place to store material. They might lose paper, they reasoned, but they wouldn't lose their phones. Some took photos because they wanted to record exactly the manner in which I had noted information on the board. Others told me that during class they liked to be able to listen to the discussion attentively. Yet the use of cameras as note takers, though it may be convenient, does raise significant questions for the classroom. Is a picture an effective replacement for the process of note - taking? Instructors encourage students to take notes because the act of doing so is more than merely recording necessary information - - it helps prepare the way for understanding. Encouraging students to take notes may be an old - fashioned instructional method, but just because a method has a long history doesn't mean it's out of date. Writing things down engages a student's brain in listening, visual, and kinesthetic learning - - a view supported by a longstanding research. The act of writing down information enables a person to begin committing it to memory, and to process and combine it, establishing the building blocks of learning new concepts. Taking a picture does indeed record the information, but it deletes some of the necessary mental engagement that taking notes employs. So can the two be equally effective? I'm not sure how to measure the effectiveness of either method. For now, I allow students to take notes however they see fit - - handwritten or photographed - - because I figure that some notes, no matter the method of note - taking, are better than none.

Вопрос: Students refuse to take notes by hand because

Ответ: they are more likely to lose notes

Результаты модели	Истинное значение (из датасета)
- they don't need to pay the students	- they are unable to take notes

Оценки: $BLEU_1 : 0.29, BLEU_2 : 0.0, BLEU_3 : 0.0, BLEU_4 : 0.0, ROUGE_L : 0.31$

A.4 Сравнение

A.4.1 Пример 1

Вопрос: We can learn from the passage that Wellington

Ответ: is the capital city of New Zealand

Истинное значение (из датасета)	- is on the South Island - is the largest city in New Zealand
Baseline DistilBERT	- the
Baseline BERT	- the name of New Zealand
G + QA architecture	- is the largest city in New Zealand

A.4.2 Пример 2

Вопрос: Which of the following relationships is healthy?

Ответ: You like to talk to someone and listen to him or her.

Истинное значение (из датасета)	- You feel upset when you are around someon - You feel that you have to spend time with h or her
Baseline DistilBERT	- your you how what
Baseline BERT	- NaN
G + QA architecture	- You don't love the other person.

A.4.3 Пример 3

Вопрос: The reasons for protecting wetlands are the following except that

Ответ: they will be good farmland

Истинное значение (из датасета)	- they are home of wildlife - they offer food to the wildlife
Baseline DistilBERT	- they will be a good place for animals
Baseline BERT	- it is easy to keep the habitats in good weather
G + QA architecture	- they will be the world's biggest the world's biggest in the world

A.4.4 Пример 4

Вопрос: What was the boy doing on the street?

Ответ: He was begging for money.

Истинное значение (из датасета)	- He was selling his old hat.
Baseline DistilBERT	- sign the sign
Baseline BERT	- He asked people to the blind boy
G + QA architecture	- He was looking for money

A.4.5 Пример 5

Вопрос: Students refuse to take notes by hand because

Ответ: they are more likely to lose notes

Истинное значение (из датасета)	- they are unable to take notes
Baseline DistilBERT	- how your s a Students students
Baseline BERT	- taking
G + QA architecture	- they don't need to pay the students