

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	3
1. ОСНОВЫ РАЗРАБОТКИ МОДЕЛЕЙ ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ДИАЛОГОВЫХ ПРИЛОЖЕНИЙ .	4
1.1. ВВЕДЕНИЕ В ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ . . .	4
1.1.1. УСТРОЙСТВО ПРОСТОЙ ИСКУССТВЕННОЙ НЕЙРОННОЙ СЕТИ	5
1.1.2. ОБУЧЕНИЕ С УЧИТЕЛЕМ ИСКУССТВЕННОЙ НЕЙРОННОЙ СЕТИ	6
1.2. ВЕКТОРНОЕ ПРЕДСТАВЛЕНИЕ ТЕКСТА	9
1.3. АРХИТЕКТУРА ТРАНСФОРМЕР	11
1.3.1. МЕХАНИЗМ ВНИМАНИЯ	12
1.3.2. МОДЕЛЬ T5: АРХИТЕКТУРА И ЕЕ ОСОБЕННОСТИ .	13
1.4. ПОСТАНОВКА ЗАДАЧИ ДИАЛОГОВОЙ СИСТЕМЫ	14
2. ОПИСАНИЕ РАЗРАБОТАННЫХ ПРОГРАММ	16
2.1. ОБРАБОТЧИК НАБОРА ДАННЫХ DNDD	16
2.2. МОДУЛИ ОБУЧЕНИЯ И ПОИСКА ОПТИМАЛЬНЫХ ГИПЕРПАРАМЕТРОВ	17
2.3. ДЕМО ПРИЛОЖЕНИЕ ДЛЯ СОЗДАНИЯ ДИАЛОГОВОЙ СИСТЕМЫ ДЛЯ РАЗРАБОТЧИКОВ ВИДЕОИГР	18
3. РАЗРАБОТКА НАБОРА ДАННЫХ DNDD	20
3.1. СБОР ДАННЫХ И СОЗДАНИЕ ДИАЛОГОВОЙ ЧАСТИ НАБОРА ДАННЫХ	20
3.1.1. СТРУКТУРА НАБОРА ДАННЫХ	20
3.1.2. СБОР ДАННЫХ	20
3.2. СИНТЕЗ ОПИСАНИЙ NPC И АНАЛИЗ НАБОРА ДАННЫХ .	21
3.2.1. ПОЛУЧЕНИЕ ОПИСАНИЙ NPC	21
3.2.2. АНАЛИЗ DNDD	23
3.3. ПОДГОТОВКА НАБОРА ДАННЫХ DNDD ДЛЯ ОБУЧЕНИЯ МОДЕЛЕЙ	25
3.3.1. ЭМУЛЯЦИЯ ДИАЛОГОВЫХ ВЗАИМОДЕЙСТВИЙ . .	25

3.3.2. СТРУКТУРИРОВАНИЕ ВХОДНЫХ ДАННЫХ ДЛЯ ОБУЧЕНИЯ	26
3.3.3. ОПТИМИЗАЦИЯ ДАННЫХ ПОД ПОТРЕБИТЕЛЬСКОЕ ОБОРУДОВАНИЕ	26
4. ОБУЧЕНИЕ МОДЕЛЕЙ	28
4.1. ПОИСК ОПТИМАЛЬНОЙ МОДЕЛИ	28
4.2. ПОИСК ОПТИМАЛЬНЫХ ГИПЕРПАРАМЕТРОВ ДЛЯ МОДЕЛИ	28
4.2.1. ПОСТАНОВКА ЗАДАЧИ	28
4.2.2. АНАЛИЗ РЕЗУЛЬТАТОВ ЭКСПЕРИМЕНТОВ С ГИПЕРПАРАМЕТРАМИ	29
4.3. ОБУЧЕНИЕ ИТОГОВОЙ МОДЕЛИ	33
5. ОПТИМИЗАЦИЯ МОДУЛЕЙ ИТОГОВОЙ СИСТЕМЫ	36
5.1. ОПТИМИЗАЦИЯ МОДУЛЯ КЛАССИФИКАЦИИ НАМЕРЕНИЙ	37
5.2. ОПТИМИЗАЦИЯ МОДУЛЯ ДИАЛОГОВОЙ МОДЕЛИ	41
ЗАКЛЮЧЕНИЕ	43
СПИСОК ЛИТЕРАТУРЫ	44
ПРИЛОЖЕНИЕ А. ПРИМЕР ТРАНСЛЯЦИИ ИЗ ЯЗЫКА D В JSON	47
ПРИЛОЖЕНИЕ Б. ИСХОДНЫЙ КОД ОБРАБОТКИ DNDD	59
ПРИЛОЖЕНИЕ В. ПРИМЕРЫ ДИАЛОГОВ	74

ВВЕДЕНИЕ

В настоящее время разработка диалоговых систем является важным направлением машинного обучения, в условиях растущей автоматизации взаимодействия человека с компьютером. Такие технологии могут быть использованы в различных отраслях, включая игровую. Недавние достижения в глубоком обучении позволяют создавать диалоговые модели, которые способны обрабатывать естественный язык и предоставлять пользователю качественные ответы на запросы. Для успешного обучения таких моделей необходимы качественные наборы данных, содержащие диалоги между участниками.

Целью работы является исследование использования генеративных нейросетевых моделей для создания диалоговой модели, которая будет способна генерировать ответы на основе образа неигрового персонажа и контекста диалога. В данной работе рассматривается процесс создания и анализа набора данных DNDD (Dungeon & Dragons Dialogues) с использованием текущих больших языковых моделей для обучения диалоговой модели, основанный на сборе и предобработке данных из различных источников с учетом образа неигровых персонажей. Такая модель позволит улучшить игровой опыт игроков, сделать этот опыт более иммерсивным. В данной работе также рассматривается процесс подготовки набора данных для обучения модели, формулирование задачи для моделирования, поиск оптимальной модели и параметров для успешного и эффективного обучения.

1. ОСНОВЫ РАЗРАБОТКИ МОДЕЛЕЙ ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ДИАЛогоВЫХ ПРИЛОЖЕНИЙ

1.1. ВВЕДЕНИЕ В ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

Обработка естественного языка является крайне тяжелой задачей для моделирования стандартными алгоритмами. Машинное обучение позволяет решать задачи на основе статистических наблюдений из данных без явной алгоритмизации решения задачи. Недавние прорывы в области обработки естественного языка показывают, что методами машинного обучения можно частично или сполна выполнять многие человеческие задачи, например, краткое изложение текста, написание кода, общение с собеседником и другие, а также добиться результатов распознавания речи сопоставимых с результатами человека [1, 2].

Одним из основных аспектов машинного обучения является искусственная нейронная сеть (далее ИНС), созданная по подобию биологических нейронных сетей. Модель ИНС – описание сети, математическая модель, часто представляемая в виде графа, нацеленная на решение задачи прогнозирования на основе обучающей выборки данных. Методы обучения используются для установки параметров модели для конкретной задачи. Наиболее распространенными методами являются обучение с учителем, обучение без учителя и обучение с подкреплением.

Каждый метод имеет свои особенности и применяется в зависимости от ситуации. Например, обычно обучение с учителем применяется в тех случаях, когда обучающий набор данных размечен на основе некоторых критериев. Такие задачи обычно являются задачами классификации, когда каждый экземпляр выборки имеет один или больше собственный класс. Такой подход имеет ограничения: как правило количество размеченных данных значительно меньше общего количество данных. В ситуациях, когда данные не размечены, применяется обучение без учителя. Благодаря такому подходу, можно обучить модель делить данные на кластеры, генерировать текст, изображения и т.д. Метод обучения с подкреплением используется, когда модель должна действовать

как интеллектуальный агент, принимая решения на основе условий окружающей среды и полученной обратной связи. Для построения мощных современных цифровых ассистентов могут использоваться все три подхода к обучению моделей, используя модели, полученные конкретным методом, в качестве промежуточных или вспомогательных, для обучения конечной модели [3].

1.1.1. УСТРОЙСТВО ПРОСТОЙ ИСКУССТВЕННОЙ НЕЙРОННОЙ СЕТИ

Устройство простой ИНС можно описать как взвешенный набор узлов, разделенный на слои, соединенные между собой активационными функциями φ . Функциям активации желательно иметь свойства: нелинейность, непрерывная дифференцируемость, бесконечная область значений, монотонность. При построении модели ИНС в качестве активационных функций часто используется одна из следующих функций:

1. Гиперболический тангенс:

$$\varphi(z) = \frac{e^{2z} - 1}{e^{2z} + 1}. \quad (1.1)$$

2. Функция ReLU:

$$\varphi(z) = \max(0, z). \quad (1.2)$$

3. Функция GELU:

$$\varphi(z) = \frac{1}{2}z \left[1 + \operatorname{erf} \left(z/\sqrt{2} \right) \right]. \quad (1.3)$$

4. Логистическая функция (сигмоида):

$$\varphi(z) = \frac{1}{1 + e^{-z}}. \quad (1.4)$$

5. Многопеременная логистическая функция (softmax):

$$\varphi(z)_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}. \quad (1.5)$$

Архитектуры ИНС могут сильно отличаться друг от друга в зависимости от поставленных задач и требований к качеству предсказаний модели. Раздел,

который занимается изучением ИНС с большим количеством скрытых слоев, т.е. тех слоев, которые находятся между входным и выходным, называется глубоким обучением, а такие модели называются глубокими. Примером такой архитектуры модели может служить трансформер [4], речь о котором пойдет дальше.

Набор весов W и смещений b являются параметрами модели, обозначаемыми как θ . Функция предсказания модели ИНС обозначается как $h_\theta(x)$. $W^{[l]}$, $b^{[l]}$, $h_\theta^{[l]}$ – веса, отклонения и выход модели на l -ом слое. Описать работу обобщенной модели ИНС с L слоями можно следующим образом:

1. $h_\theta^{[0]} = x$.
2. $h_\theta^{[l]} = \varphi \circ \left(W^{[l-1]} h_\theta^{[l-1]}(x) + b^{[l-1]} \right)$, где $1 \leq l \leq L - 1$.
3. $h_\theta = h_\theta^{[L]} = W^{[L-1]} h_\theta^{[L-1]}(x) + b^{[L-1]}$.

Примером простой ИНС может являться однослойный перцептрон. Схема однослойного перцептрона представлена на рис. 1.1.

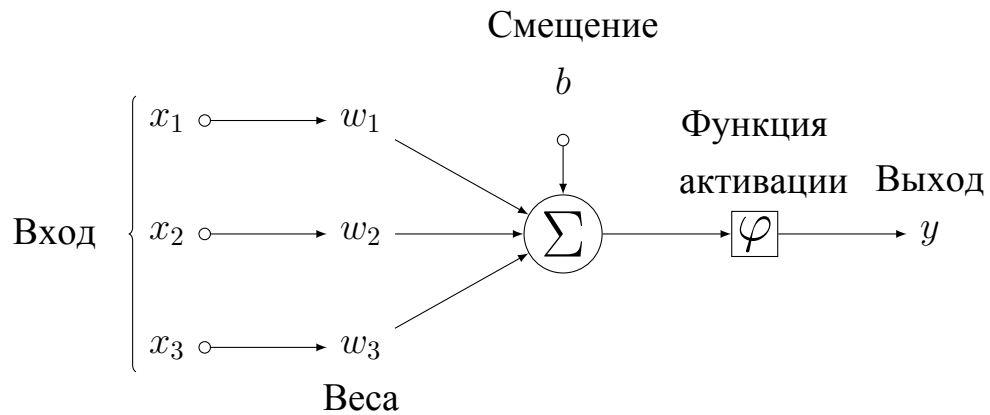


Рисунок 1.1 – Однослойный перцептрон

1.1.2. ОБУЧЕНИЕ С УЧИТЕЛЕМ ИСКУССТВЕННОЙ НЕЙРОННОЙ СЕТИ

Как было сказано ранее, для того, чтобы обучить ИНС с учителем, требуется иметь такой набор данных, где каждый элемент имел соответствующую метку класса. Элементы набора данных, т.е. входные данные, принадлежат некоторому входному пространству \mathcal{X} , например, картинкам кошек, а метки

принадлежат к выходному пространству \mathcal{Y} , например, породе кошек. Из такого набора данных \mathcal{D} мы строим тренировочную подвыборку, состоящую из пар, элементов:

$$\mathcal{D}_{\text{train}} = \{ (x_i, \hat{y}_i) \mid x_i \in \mathcal{X}, \hat{y}_i \in \mathcal{Y}, i = \overline{1, \dots, n}, n \leq |\mathcal{D}| \}. \quad (1.6)$$

Мы стремимся получить целевую функцию ИНС h_{θ^*} с оптимальным набором параметров θ^* на основе $\mathcal{D}_{\text{train}}$, при котором h_{θ^*} наиболее эффективно отображает из пространства \mathcal{X} в пространство \mathcal{Y} . Для определения того, насколько эффективно предсказывает модель, требуется иметь неотрицательную функцию $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^+$, которая измеряет ошибку предсказания $h_{\theta}(x)$ по отношению к истинной метке \hat{y} . Такие функции, как правило, называются функциями ошибки или функциями потерь. Функция потерь выбирается исходя из условий конкретной задачи, но часто является одной из следующих функций:

1. Функция потерь L_1 :

$$\ell(h_{\theta}(x), \hat{y}) = |\hat{y} - h_{\theta}(x)|. \quad (1.7)$$

2. Функция потерь L_2 :

$$\ell(h_{\theta}(x), \hat{y}) = (\hat{y} - h_{\theta}(x))^2. \quad (1.8)$$

3. Функция потерь перекрестной энтропии:

$$\ell(h_{\theta}(x), \hat{y}) = -\hat{y} \log h_{\theta}(x). \quad (1.9)$$

4. Функция потерь отрицательного логарифмического правдоподобия:

$$\ell(h_{\theta}(x), \hat{y}) = -[\hat{y} \log h_{\theta}(x) + (1 - \hat{y}) \log(1 - h_{\theta}(x))]. \quad (1.10)$$

Обучение модели с учителем сводится к задаче минимизации функции потерь по тренировочной выборке:

$$\mathcal{L}_{\mathcal{D}_{\text{train}}}(\theta) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{i=1}^{|\mathcal{D}_{\text{train}}|} \ell(h_{\theta}(x_i), \hat{y}_i) \rightarrow \min_{\theta}. \quad (1.11)$$

Чтобы решить такую задачу минимизации функции потерь по тренировочной выборке, требуется вычислить:

$$\frac{\partial \mathcal{L}_{\mathcal{D}_{\text{train}}}(\theta)}{\partial \theta}. \quad (1.12)$$

Метод, позволяющий аналитически вычислить градиент (1.12) в точке, называется обратным распространением [5]. В основе метода лежит автоматическое построение расчетного графика и правило вычисления производной сложной функции. При полученном градиенте функции потерь параметры модели ИНС изменяются алгоритмом оптимизации. Одной из важных составляющих алгоритмов оптимизации является выбор размера шага оптимизатора η , также называемого скоростью обучения, и планировщика скорости обучения ς , так как они влияют на скорость процесса обучения и преодоление локальные минимумы методом оптимизации. Распространенными вариантами таких алгоритмов являются: «*Gradient descent*» (градиентный спуск), «*Stochastic gradient descent*» (стохастический градиентный спуск), Adam, AdaFactor [6, 7].

Обучение является итеративным процессом, где итерация или шаг итерации – это обработка моделью одного или нескольких примеров обучающей выборки. Обработка полного набора выборки называют эпохой.

Алгоритм обучения модели ИНС с учителем представлен ниже.

Алгоритм 1 Обучение модели ИНС с учителем

- 1: Инициализировать θ случайно или по некоторому закону распределения.
 - 2: По каждой эпохе из количества эпох:
 - 3: По каждому примеру (x, \hat{y}) из обучающей выборки $\mathcal{D}_{\text{train}}$:
 - 4: Получить предсказание модели $y \leftarrow h_{\theta}(x)$.
 - 5: Получить значение функции потерь $\ell(y, \hat{y})$.
 - 6: Получить градиент $\nabla \ell$ методом обратного распространения.
 - 7: Сделать шаг оптимизации.
 - 8: Аккумулировать значение общей функции потерь $\mathcal{L} \leftarrow \mathcal{L} + \ell(y, \hat{y})$.
-

Однако одной тренировочной подвыборки чаще всего не достаточно для успешного обучения модели. Как правило используют три подвыборки исходных данных \mathcal{D} . Помимо обучающей, используется валидационная \mathcal{D}_{val} , которая используется в конце эпохи обучения, на которой модель не обучается, но проверяется на наборе данных, которые она не видела, для корректировки гиперпараметров модели. Гиперпараметры – это параметры, которые используются

для контроля процесса обучения. Примерами гиперпараметров могут служить как вышеупомянутые η и ς , так и количество слоев в модели, активационные функции и т.д. Для оценки итогового качества модели обычно используется тестовая выборка $\mathcal{D}_{\text{test}}$. Методы, которые разбивают исходный набор данных \mathcal{D} на подвыборки, называются методами стратификации.

Хоть ℓ , \mathcal{L} и показывают качество прогнозирования модели h_θ , но на практике анализировать качество модели только по значениям функции потерь – это сложная задача. Помимо функции потерь используются метрики оценки прогнозирования. Выбор метрик сильно зависит от поставленной задачи.

1.2. ВЕКТОРНОЕ ПРЕДСТАВЛЕНИЕ ТЕКСТА

Из характера работы ИНС следует, что данные, на которых обучается модель, являются числовыми. Поэтому при обработке текстовых данных необходимо преобразовать их в числовые, чтобы обучить модель ИНС.

Простейшим методом представления слов в векторном пространстве является «*One-Hot Encoding*» (быстрое кодирование). Его суть заключается в присвоении каждому слову из входной последовательности слов вектора, где в позиции, соответствующей слову в словаре размерностью словаря, ставится единица, а во всех остальных позициях – ноль. Словарь содержит весь список возможных слов для кодирования. Размерность такого вектора составляет $1 \times N$, где N – количество слов в словаре. Пример быстрого кодирования показан ниже.

Таблица 1.1 – Пример словаря

Цвет
Красный
Зеленый
Синий

Таблица 1.2 – Пример быстрого кодирования

Красный	Зеленый	Синий
1	0	0
0	1	0
0	0	1

Представлением текстовых данных в численном виде могут заниматься и модели ИНС: учить полезную информацию о входной последовательности, сжато представлять текст в векторном пространстве, решая задачу языкового моделирования, для последующего использования этого представления на конечной задаче, например, задаче классификации или задаче генерации текста. Одной из первых широко распространенных обученных моделей для кодирования текста является Word2vec [8].

В современных моделях для обработки естественного языка в качестве основы векторного представления данных используют метод, называемый токенизацией. Токенизация – разбиение входного текста на части, называемые токенами. В качестве части текста могут быть как слова целиком, так и части слов. Токенизация представляет входной текст как вектор, состоящий из номеров токенов в общем словаре. Размер закодированной последовательности может зависеть как от длины входной последовательности, так и от требуемой длины. Если входная последовательность короче требуемой длины, неиспользуемое пространство может быть заполнено специальными токенами. Примером токенизации является «*Byte-Pair Encoding*» (кодирование пар байтов) [9]. для входной последовательности «Many words map to one token, but some don't: indivisible. Sequences of characters commonly found next to each other may be grouped together: 1234567890» представлен ниже. Токены, на которые разбивает токенизатор входную последовательность, показаны на рис. 1.2.

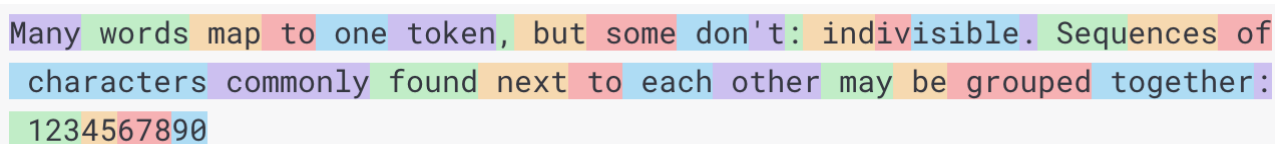


Рисунок 1.2 – Пример работы токенизации

Векторное представление такой последовательность токенов показано на рис. 1.3.

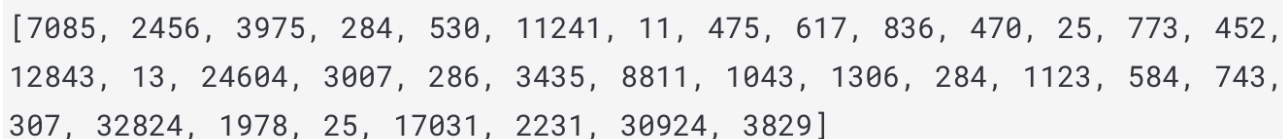


Рисунок 1.3 – Векторное представление токенов

1.3. АРХИТЕКТУРА ТРАНСФОРМЕР

Стандартным выбором архитектуры модели ИНС при обработке естественного языка является архитектура трансформер. Одними из первых моделей, созданных на базе данной архитектуры, стали GPT [10], T5 [11] и BERT [12]. Трансформер состоит из набора блоков «*encoder*» (кодировщика) и «*decoder*» (декодировщика).

Для начала происходит токенизация входного текста, а затем полученное векторное представление токенов дополняется позиционным кодированием, чтобы учитывать информацию о позиции токена во входном тексте.

Далее полученное векторное состояние передается на N идущих друг за другом блоков кодировщика. Каждый блок кодировщика состоит из двух главных компонент: механизм «*Self-Attention*» (самовнимание) и сети прямого распространения (обобщенная версия сети, показанной на рис. 1.1). После прохождения N блоков кодировщика, векторное состояние передается к N блокам декодировщика.

В свою очередь каждый блок декодировщика схож с устройством блока кодировщика с добавлением «*Cross-Attention*» (перекрестного внимания) от векторного представления состояния кодировщика. Полную версию архитектуры трансформер можно наблюдать на рис. 1.4.

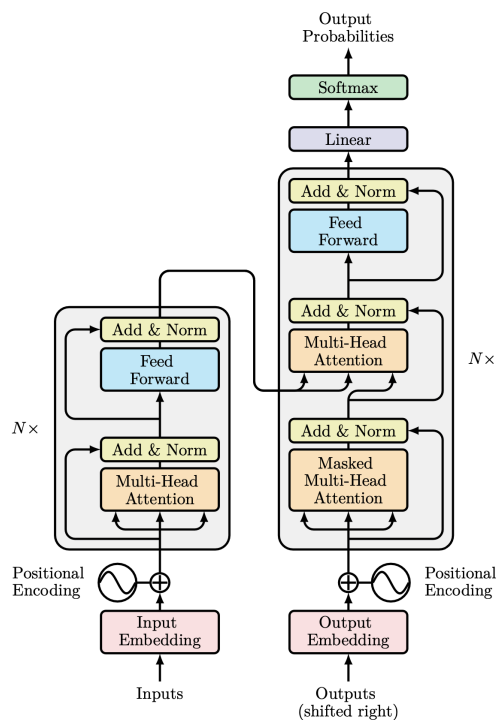


Рисунок 1.4 – Архитектура трансформер

1.3.1. МЕХАНИЗМ ВНИМАНИЯ

Механизм внимания – ключевой механизм в архитектуре трансформер. Его суть заключается в учитывании взаимодействия элемента входной последовательности со всеми другими элементами. Таким образом, модель может фокусироваться на более важных частях данных, даже если такая информация содержится в небольшой части данных.

Разберем более подробно этот механизм. Входное векторное состояние данных представляется как набор трех главных компонент внимания: «*query*» (запрос), «*keys*» (ключи) и «*values*» (значения). Преставление входных данных осуществляется за счет проекции входного векторного состояния I на пространства этих компонент, т.е.:

1. $Q = I \cdot W_Q^T$.
2. $K = I \cdot W_K^T$.
3. $V = I \cdot W_V^T$.

Из полученных векторов вычисляем результирующий вектор:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V, \text{ где } d_k = \dim(K). \quad (1.13)$$

Проинтерпретировать формулу 1.13 можно следующим образом:

1. Запрос Q проецируется на пространство ключей K , в результате получается матричное произведение: $S = QK^T$.
2. В пространстве ключей K ключи, наиболее похожие на запрос Q , идентифицируются с помощью меры сходства, такой как скалярное произведение. Функция softmax применяется к этим сходствам, чтобы получить распределение вероятностей по ключам: $A = \text{softmax}\left(\frac{S}{\sqrt{d_k}}\right)$.
3. Полученное распределение вероятностей по ключам используется для получения взвешенной суммы соответствующих векторов значений: $O = AV$.

Важно отметить, что матрицы внутреннего состояния W_Q, W_K, W_V – обучаемые параметры.

В случае, когда Q, K, V получаются из одного внутреннего состояния, такой вид механизма внимания называется самовниманием. Если K, V получены как выход внутреннего состояния кодировщика, а Q получен как внутреннее состояние декодировщика, то такой вид внимания называется перекрестным вниманием. Такой механизм позволяет модели учитывать взаимодействие между элементами входной и выходной последовательностей. Таким образом, блоки декодировщика могут использовать информацию из блоков кодировщика для генерации правильных элементов выходной последовательности.

Также может потребоваться, чтобы модель оперировала векторным состоянием входного текста только до позиции текущего токена. Чтобы модель не использовала информацию за пределами этой точки, применяется маскированное внимание.

Вместо вычисления одного внимания с размерностью d_{model} , можно вычислять внимание с фокусом на разные участки входной последовательности параллельно. Такое внимание называется «*Multi-Head Attention*» (многоголовое внимание) и вычисляется как:

$$\text{MHA}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O, \quad (1.14)$$

где $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$.

Благодаря тому, что операции перемножения матриц – высокооптимизируемые операции, данный механизм эффективен с точки зрения производительности.

1.3.2. МОДЕЛЬ T5: АРХИТЕКТУРА И ЕЕ ОСОБЕННОСТИ

Обучение модели архитектуры трансформера обычно происходит в два этапа. Сначала модель обучается решать задачу языкового моделирования на огромном наборе размеченных данных. Такой процесс крайне затратен, т.к. требует больших мощностей и огромного набора данных. Такой этап называется «*pretrain*» (предварительное обучение), после чего модель дообучают на конкретной задаче, например, на генерации текста или на задаче поддержания

диалога, на меньшем, но размеченном наборе данных. Этап дообучения значительно менее затратен, чем предварительное обучение.

Модель «*Text-To-Text Transfer Transformer*» (T5) – это модель глубокого обучения, в которой используется архитектура преобразователя, разработанная Google для решения различных задач обработки естественного языка. Одним из основных преимуществ этой модели является ее универсальность: T5 был изначально обучен таким задачам, как перевод, резюмирование текста, классификация текста и ответы на вопросы, что достигается за счет использования специального префикса для различения задач. Еще одной примечательной особенностью модели T5 является ее способность обрабатывать различные типы ввода и выходные данные. Например, модель может обрабатывать текстовые данные различных длин и форматов, а также генерировать тексты различных стилей и тематик. В качестве токенизатора T5 использует SentencePiece [13].

Предварительное обучение T5 производилось на наборе данных «*Colossal Clean Crawled Corpus*» (C4), содержащий 356 миллиардов токенов, занимающий 750 гигабайт дискового пространства, на задаче «*Masked Language Modeling*» (замаскированное языковое моделирование). Задача заключается в восстановлении исходного текста на основе «поврежденного» текста, где «повреждалось» 15% токенов, в которых 90% заменялось на специальный токен [MASK], а остальные 10% заменялись на случайный токен из словаря.

После предварительного обучения модель дообучили на конечных задачах, описанных выше. Со временем набор изначальных задач расширили набором задач, состоящим из инструкций на понимание и генерацию текста на естественном языке, что значительно улучшило качество модели для последующего обучения, например, на задаче поддержания диалога [14].

1.4. ПОСТАНОВКА ЗАДАЧИ ДИАЛОГОВОЙ СИСТЕМЫ

Эмуляция диалога, обучение диалоговых агентов или чат-ботов относятся к области генерации и классификации текста. Такие модели должны эффективно обрабатывать естественный язык и формировать ответы в рамках диалога. В качестве диалоговой системы может выступать не одна модель ИНС. Различные задачи, например, классификации, генерации текста и автоматического распознавания речи могут выполнять разные модели. Разберем основные компоненты диалога:

1. Состояние диалога: Диалоговая система должна понимать намерения запроса пользователя и те сущности, которые фигурируют в запросе. Намерением пользователя может быть покупка, а сущностью – товар. Такие задачи являются задачами классификации.
2. Контекст диалога: диалоговая система должен учитывать контекст предыдущих сообщений, чтобы дать более точный и подходящий ответ.
3. Цель диалога: диалоговая система может иметь цель, которую она преследует в рамках диалога. Такой целью может быть, например, имитация поведения неигрового персонажа.
4. Шаг диалога: одна итерация в обмене сообщениями между пользователем и диалоговой системой. Каждый шаг диалога представляет собой один вопрос или одно сообщение от системы, на которое пользователь должен ответить. Затем система обрабатывает ответ пользователя и переходит к следующему шагу диалога. Шаги диалога помогают упорядочить и структурировать обмен сообщениями между пользователем и диалоговой системой. Пример шагов диалога приведен в таблице 1.3.

Таблица 1.3 – Пример диалоговых шагов

Шаг 1	Система: Здравствуйте, чем я могу Вам помочь?
Шаг 2	Пользователь: Добрый день, я хочу заказать пиццу на дом.
Шаг 3	Система: Какой размер пиццы Вы хотели бы заказать?

Для обучения диалоговых моделей, способных продолжить диалог, требуется набор данных, содержащий диалоги. Такую задачу можно решить обучением с учителем. Для этого необходимые компоненты диалога должны быть в формате (x, \hat{y}) , где в качестве x выступает строка, содержащая цель диалога и его контекст, а в качестве \hat{y} выступает желаемый ответ диалоговой модели.

2. ОПИСАНИЕ РАЗРАБОТАННЫХ ПРОГРАММ

Язык программирования Python [15] является стандартным выбором языка для разработки и обучения моделей ИНС благодаря богатой экосистеме пакетов. Сами пакеты для разработки моделей ИНС могут быть написаны на более низкоуровневом языке программирования, например, на C++ [16] с использованием CUDA [17] для высокой производительности кода, в то время как пакет, которым пользуется разработчик, доступен в качестве интерфейса на языке Python для высокой производительности разработчика. Исходя из этого, выбором языка, на котором написаны алгоритмы обработки набора данных DNDD, поиска оптимальных параметров и обучения модели является Python.

2.1. ОБРАБОТЧИК НАБОРА ДАННЫХ DNDD

Обработчик набора данных DNDD доступен как приложение командной строки, полный код которого показан в приложении Б. Параметры обработчика:

1. Подмножество игр, которые будут обрабатываться приложением указывается как аргумент командной строки `-subsets`. Если ведется обработка полного набора данных, то этот аргумент можно опустить или присвоить ему значение `all`.
2. Сгенерировать описания неигровых персонажей (далее NPC) можно аргументом `-generate_descriptions`. В таком случае необходимо либо иметь доступ к серверу с языковой моделью, либо предварительно запустить сервер самостоятельно, используя например простой интерфейс от Gradio [18]. В любом случае желательно передать URL на эндпоинт, где доступна языковая модель, аргументом `-model_server_url`. При отсутствии такого аргумента URL по умолчанию будет `http://127.0.0.1:7860`.
3. Если список описаний NPC уже есть, то его можно передать в приложение через аргумент `-description_file`.

4. В случае, если требуется, чтобы приложение сохранило работу обработанного набора данных, который готов к конечному использованию для обучения модели, тогда используется аргумент `-build-final`. Если требуется ограничить максимальное количество диалогов NPC, тогда следует использовать аргумент `-limit_dialogues`.

При разработке приложения для обработки набора данных DNDD были использованы следующие Python пакеты и библиотеки:

- Argparse, встроенная в язык Python библиотека создания программ командной строки.
- Tqdm [19] для визуализации прогресса работы приложения.
- Requests, встроенная в язык Python библиотека запросов, для доступа к языковой модели через REST «*Application Programming Interface*» (API).
- Datasets [20] для сериализации и хранения обработанного набора данных и объекта, который используется как набор данных при обучении модели.
- Transformers [21] для оценки количества токенов отправляемых данных в модель для генерации описаний.
- Pandas [22] для внутренней обработки набора данных.
- Os, встроенная в язык Python библиотека взаимодействия с операционной системой, для работы с файлами.
- Json, встроенная в язык Python библиотека взаимодействия с JSON файлами, для десериализации исходных данных.

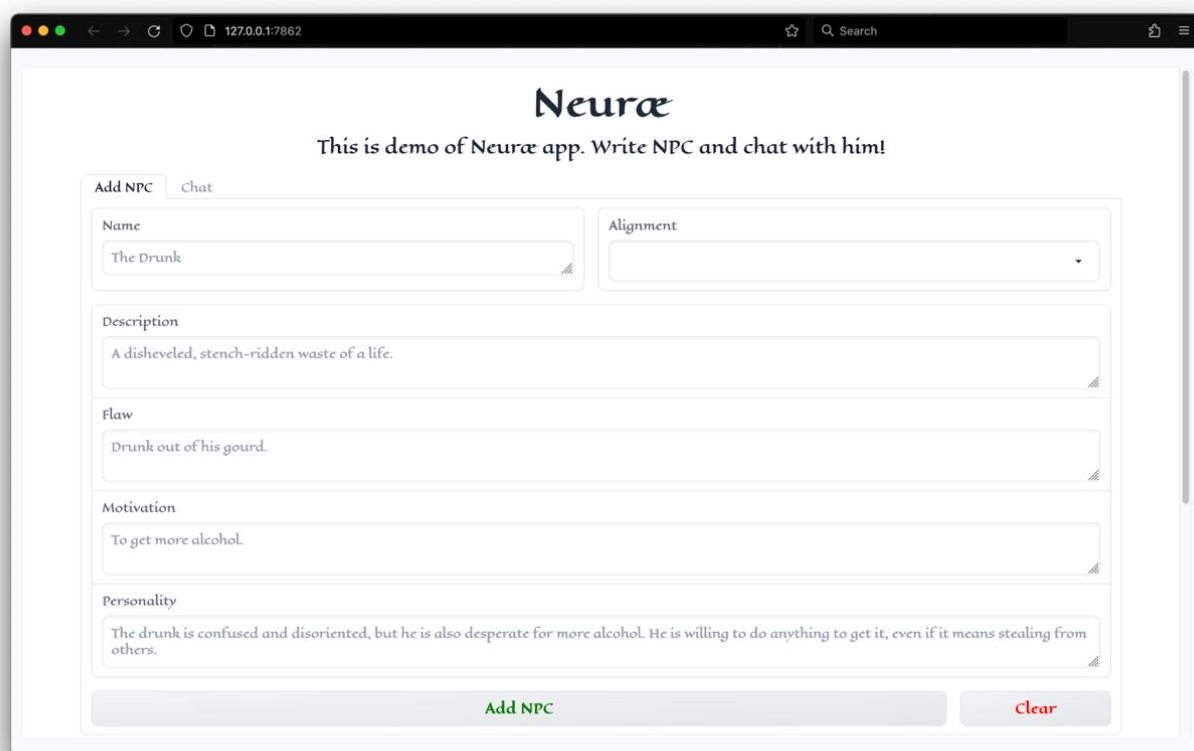
2.2. МОДУЛИ ОБУЧЕНИЯ И ПОИСКА ОПТИМАЛЬНЫХ ГИПЕРПАРАМЕТРОВ

Поиск оптимальных гиперпараметров и обучение модели реализованы через Python скрипты. Для обучения модели была использована библиотека Transformers, а для оптимизации процесса обучения и используемых вычислительных мощностей библиотека PEFT [23]. Для загрузки подготовленных

наборов данных используется библиотека Datasets, после чего токенизируется для дальнейшей работы. Для поиска оптимальных гиперпараметров класс sweep библиотеки Weights & Biases [24] перебирает указанный набор.

2.3. ДЕМО ПРИЛОЖЕНИЕ ДЛЯ СОЗДАНИЯ ДИАЛоговой СИСТЕМЫ ДЛЯ РАЗРАБОТЧИКОВ ВИДЕОИГР

В качестве демо приложения диалоговой системы было написано веб-приложение на фреймворке Gradio. В этом приложении можно создать NPC и пообщаться с ним через текстовый или аудио интерфейс. Приложение содержит модули автоматического распознавания речи, эмоциональной классификации речи, классификации намерений, семантического поиска и диалоговой модели. Доступ к приложению осуществляется как через REST API, так и через веб-интерфейс. На экране создания NPC можно заполнить поля, необходимые для описания NPC, и добавить NPC в список доступных для диалога. После создания NPC, на экране диалога можно выбрать конкретного NPC, с которым можно провести диалог. Экран создания NPC можно наблюдать на рис. 2.1, а экран диалога на рис. 2.2.



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:7862'. The page title is 'Neuræ' and the subtitle is 'This is demo of Neuræ app. Write NPC and chat with him!'. The interface has two tabs: 'Add NPC' (selected) and 'Chat'. The 'Add NPC' tab contains several text input fields for creating an NPC:

- Name:** 'The Drunk'
- Alignment:** A dropdown menu.
- Description:** 'A disheveled, stench-ridden waste of a life.'
- Flaw:** 'Drunk out of his gourd.'
- Motivation:** 'To get more alcohol.'
- Personality:** 'The drunk is confused and disoriented, but he is also desperate for more alcohol. He is willing to do anything to get it, even if it means stealing from others.'

At the bottom of the form, there are two buttons: 'Add NPC' (green text) and 'clear' (red text).

Рисунок 2.1 – Экран создания NPC

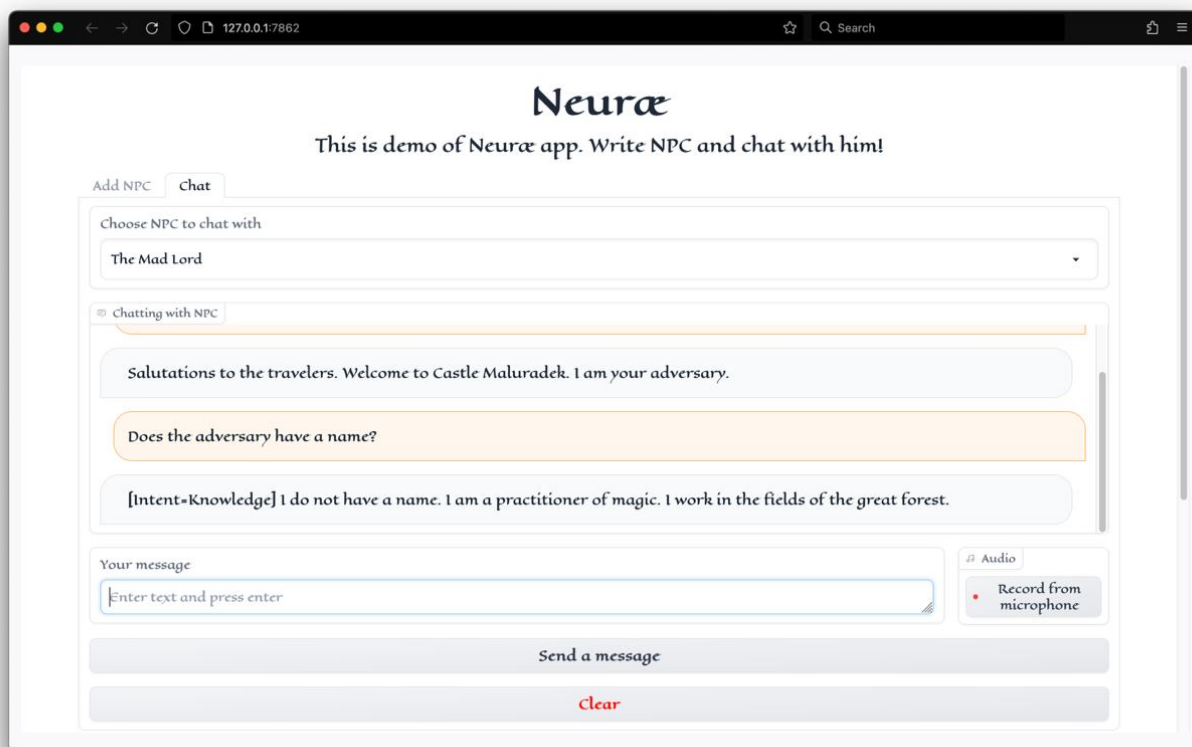


Рисунок 2.2 – Экран диалога с NPC

3. РАЗРАБОТКА НАБОРА ДАННЫХ DNDD

3.1. СБОР ДАННЫХ И СОЗДАНИЕ ДИАЛОГОВОЙ ЧАСТИ НАБОРА ДАННЫХ

3.1.1. СТРУКТУРА НАБОРА ДАННЫХ

Чтобы создать набор данных для обучения диалоговой модели, которая эмулирует поведение NPC по заданному описанию в играх, необходимо иметь диалоги, построенные по определенной системе правил. Одной из самой распространенной, обширной и гибкой системой правил, по которым можно описать NPC, является система *Dungeon & Dragons* (далее D&D), т.к. она обладает вполне определенной структурой. Например, персонажи обязаны иметь конкретное мировоззрение, определяющее их поведение и взгляды на поступки, мотивацию, внешнее описание и слабости. Поэтому выбор такой системы выглядит естественным.

Набор данных, созданный из данных игр во вселенной D&D, должен содержать примеры диалогов NPC с главным героем и примеры описания NPC в формате Name/Alignment/Description/Flaw/Motivation/Personality.

3.1.2. СБОР ДАННЫХ

СБОР И ПРЕДОБРАБОТКА ПЕРВОНАЧАЛЬНЫХ ДАННЫХ

Первоначальные данные были получены из следующих игр:

1. «Icewind Dale: Enhanced Edition».
2. «Planescape: Torment: Enhanced Edition».

Такой выбор игр неслучаен: все эти игры были созданы с помощью игрового движка *Infinity Engine*. Диалоги были получены следующим образом:

1. Диалоги, которые были использованы, находились в скомпилированном файле, который можно было найти внутри «.bif» файла.
2. Чтобы получить диалоги NPC в скомпилированном формате с расширением «.dlg», была использована программа *WinBif*.

3. Далее, с помощью WeiDU [25], специального транслятора, написанного для создания собственных диалогов в играх Infinity Engine в качестве модификации, эти файлы были преобразованы в формат языка «.d»;
4. Наконец, полученные файлы конвертированы в удобный для анализа JSON-формат. Такие файлы содержат возможные диалоги NLP с главным героем.

ПОСТОБРАБОТКА ПЕРВОНАЧАЛЬНЫХ ДАННЫХ

Для начала полученные JSON файлы были проанализированы на предмет NPC, т.к. в этих играх есть описания взаимодействия с неодушевленными предметами: порталами, сферами и т.д. Из-за того, что такие описания не содержат непосредственно диалогов, они были удалены из выборки. Также было замечено, что в игре «Planescape: Torment: Enhanced Edition» в отличие от остальных игр на движке Infinity Engine в диалоговых файлах (помимо самих диалогов) в текстовом виде достаточно часто было описано то, что видит перед собой игрок, что в последствии сильно поможет составлению набора данных. Такую полезную информацию нельзя упускать и следует иметь помимо обычных реплик NPC дополнительный контекст диалога. К тому же игра обладает самым большим размером корпуса диалогов среди игр во вселенной D&D.

3.2. СИНТЕЗ ОПИСАНИЙ NPC И АНАЛИЗ НАБОРА ДАННЫХ

3.2.1. ПОЛУЧЕНИЕ ОПИСАНИЙ NPC

Хоть получение диалогов NPC является необходимым первым шагом в создании набора данных для моделирования диалогов, этого недостаточно, если учесть важность имитации ответов NPC в наборе данных. Анализ доступных инструментов модификации игр для Infinity Engine показал, что преобразование имен файлов диалогов в соответствующие символы, отображаемые в игре, требует логики, специфичной для игры, а получение имен персонажей из имен файлов диалогов с помощью обычных инструментов весьма неоднозначно. Чтобы преодолеть эту проблему, были использованы передовые языковые модели для синтеза описаний NPC в определенном формате. Был использован метод Few-Shot, который включал предоставление примеров реплик NPC, со-

держащих частичные описания NPC. В результате алгоритм получения описаний NPC выглядит следующим образом:

1. Все диалоги группируются по имени файла, из которого были получены диалоги NPC с игроком.
2. Формируются уникальные и упорядочные примеры реплик NPC так, чтобы количество токенов в примерах + количество токенов в инструкции не превышало 512 токенов.
3. Примеры реплик вместе с инструкцией отправляются в виде запроса на сервер, обслуживающий модель.
4. Полученный ответ записывается в файл формата «.csv» в виде filename, description.

Все эксперименты были выполнены на потребительском оборудовании, включающем 32 Гб оперативной памяти, 20-ядерный процессор и графическую карту NVIDIA RTX 3090Ti с объемом памяти 24 Гб. На начало первого квартала 2023 года, одним из наиболее эффективных семейств предобученных моделей для генерации текста по количеству параметров и выходных метрик является LLaMa [26]. На основе инструкционных данных, было разработано семейство моделей Alpaca [27], которые достигают качества ответов, сопоставимого с результатами модели ChatGPT [28].

Ограничения потребительского оборудования сказались и на размере контекста, вмещаемого в модель. Экспериментно было определено, что максимальный размер входной последовательности составляет 512 токенов, а максимальный размер выходной – 256. В связи с ограничениями на размер входных данных диалог отправляется не весь, а только лишь его часть и при том реплики NPC, потому как в них содержится описание NPC. Было обнаружено, что реплики игрока не влияют на синтезируемое описание NPC, что позволяет экономить размер входных данных.

Для устойчивого синтеза данных пробовались различные комбинации инструкций и параметров генерации модели. Иногда ответы большой языковой модели не соответствуют требуемому формату, в таком случае стоит пробовать разные подходы к написанию инструкций. Например, использование формулировки «Сгенерируй данные в формате: ПРИМЕР_ФОРМАТА» в отрыве

от «Сгенерируй данные в формате: ПРИМЕР_ФОРМАТА, НЕ ИСПОЛЬЗУЯ [ПРИМЕР]», позволяет получить необходимый ответ с меньшим количеством галлюцинаций и ошибок. Такое поведение можно обосновать тем, что корпус текстов с такими видами формулировок использованный в качестве данных для обучения был больше.

Выгода использования больших языковых моделей для синтеза данных проявляется еще в том, что не для всех NPC написаны описания на таких ресурсах, как wiki и др. Поэтому, используя такой подход, большее количество данных можно покрыть необходимыми описаниями.

В данной работе в качестве инструкции была выбрана следующая: «Create the personality of a single NPC in DnD style, based on the provided example dialogue in JSON format. Answer in format Name/Alignment/Description/Flaw/Motivation/Personality in a list format written in third person». Таким образом были получены описания NPC и датасет DNDD имеет все необходимые компоненты для обучения диалоговой модели.

По данной инструкции были сгенерированы подобные описания:

Name: The Drunk

Alignment: Neutral

Description: A disheveled, stench-ridden waste of a life.

Flaw: Drunk out of his gourd.

Motivation: To get more alcohol.

Personality: The drunk is confused and disoriented, but he is also desperate for more alcohol. He is willing to do anything to get it, even if it means stealing from others.

3.2.2. АНАЛИЗ DNDD

Набор данных состоит из 4,5 гигабайт текстовой информации, содержащей в общей сложности 981 тысячу диалогов и 986 миллионов токенов, среди которых 824 миллиона соответствуют ответам NPC. Это составляет 83% от общего количества токенов. Набор данных также включает 27 миллионов ходов, причем ответы NPC представляют 14 миллионов из них. Примечательно, что некоторые взаимодействия NPC с игроком состоят только из одного предложения.

На рисунках 3.1, 3.2 отображены распределения частоты диалоговых ходов по длине, а на рисунках 3.3, 3.4 отображены распределения частоты количества токенов в диалоге по количеству. Наблюдения показывают, что реакция NPC имеет более продолжительную среднюю продолжительность по сравнению с реакцией игрока. Это можно объяснить отличительной особенностью ответов NPC, которая включает в себя хранение дополнительной контекстуальной информации, что приводит к более полному пониманию текущего разговора и, следовательно, к диалогам более высокого качества.

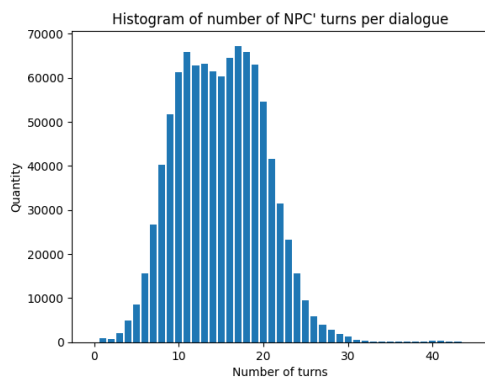


Рисунок 3.1 – Распределение частоты диалоговых ходов NPC по длине

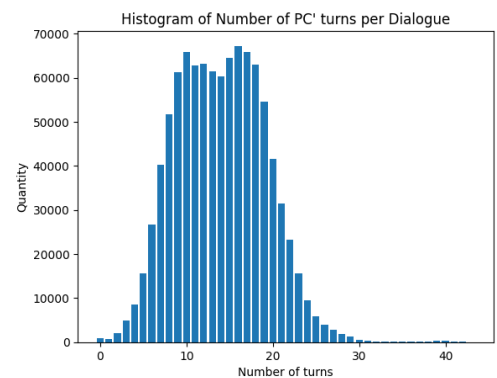


Рисунок 3.2 – Распределение частоты диалоговых ходов игрока по длине

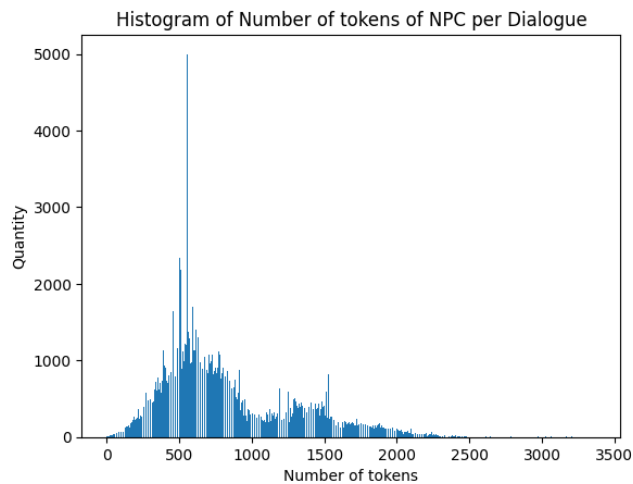


Рисунок 3.3 – Распределение частоты количества токенов в диалоге NPC по количеству

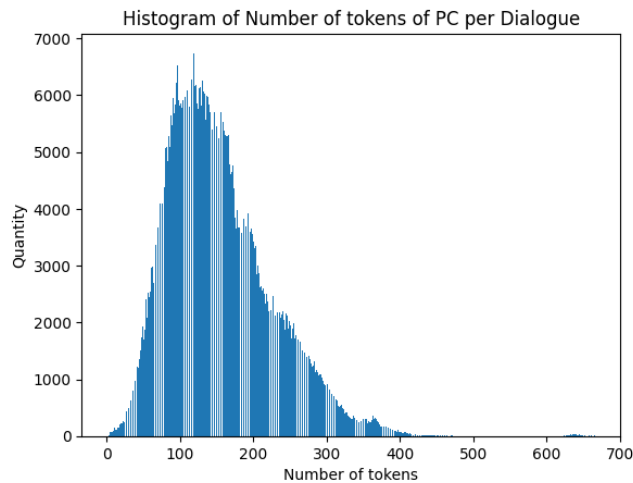


Рисунок 3.4 – Распределение частоты количества токенов в диалоге игрока по количеству

3.3. ПОДГОТОВКА НАБОРА ДАННЫХ DNDD ДЛЯ ОБУЧЕНИЯ МОДЕЛЕЙ

После фазы сбора диалоговых данных и генерации параметров NPC, включающих в себя идентификаторы, характеристики, мировоззрение, мотивацию и слабости, следующим логическим этапом становится подготовка собранных данных к процессу обучения модели. Этот процесс включает в себя конкатенацию данных в строковом формате.

3.3.1. ЭМУЛЯЦИЯ ДИАЛОГОВЫХ ВЗАИМОДЕЙСТВИЙ

Особое внимание следует уделить диалоговым взаимодействиям между NPC и игроком. В контексте набора данных, где хранятся полные версии диалогов, эмуляция процесса общения игрока с NPC требует разбиения истории диалога на подмножества. В этом случае диалог представляет собой серию ходов между игроком и NPC, и основной задачей модели является продолжение данного диалога, т.е. совершение следующего хода в диалоге.

При таком подходе модель обучается на основе итеративного процесса диалога, что способствует приближению к более реалистичному моделированию процесса диалога. Это позволяет на каждом этапе оптимизировать процесс обучения для достижения максимально эффективного результата.

3.3.2. СТРУКТУРИРОВАНИЕ ВХОДНЫХ ДАННЫХ ДЛЯ ОБУЧЕНИЯ

Для оптимизации процесса обучения, входная последовательность, а именно описание неигрового персонажа, история диалогов с игроком, последняя реплика игрока и реплика, которую должна сгенерировать модель, была разделена на сегменты, каждый из которых был помечен соответствующим образом. Такой подход к структурированию входных данных для модели позволяет ясно разделять различные компоненты входных данных, что облегчает задачу модели и способствует более эффективному обучению.

Для обозначения начала диалога используется уникальный идентификатор «EMPTY», который функционирует как сигнал о том, что диалог только что был инициирован. В силу специфики набора данных DNDD, полученного из игр, где неигровые персонажи всегда начинают диалог первыми, было определено, что первая реплика игрока служит активацией диалога, и обозначена она идентификатором «START DIALOGUE». В ходе последующего диалога реплики участников регистрируются в истории диалога с пометками «Player: » и «NPC: », в зависимости от того, кто в данный момент выступает в роли говорящего.

3.3.3. ОПТИМИЗАЦИЯ ДАННЫХ ПОД ПОТРЕБИТЕЛЬСКОЕ ОБОРУДОВАНИЕ

Учитывая ограниченные вычислительные ресурсы потребительского уровня, включая оперативную память объемом 32 Гб и графическую карту NVIDIA GeForce RTX 3090 Ti с 24 Гб памяти, было необходимо ввести определенные ограничения на обрабатываемую историю диалогов. При превышении диалогом лимита в 1024 токенов, для обеспечения управляемости данных самые старые записи в диалоге подлежали удалению. Это позволяло оптимизировать использование доступных вычислительных ресурсов и обеспечивать стабильный процесс обучения моделей.

Также было ограничено максимальное количество диалогов, которых может иметь игрок с одним неигровым персонажем. Этот подход позволяет получить более разнообразный набор данных с уменьшенным размером выборки.

Итоговые данные выглядят следующим образом. Входная последовательность:

Below is the definition of in-game NPC.

NPC Name: Digby

Alignment: Neutral

Description: A burly, bearded man with a thick accent and a penchant for trapping.

Personality traits: Digby is a bit of a glutton, and often overindulges in food and drink.

Flaws: Digby is motivated by the prospect of making a profit from his trapping.

Motivation: Digby is a gruff, no-nonsense man who is quick to anger and slow to trust. He is a hard worker and is not afraid to get his hands dirty. He is also a bit of a glutton, and often overindulges in food and drink.

Dialogue history:

Player: START DIALOGUE

NPC: *burp* Think I had too much to drink last night. Heh! What am I sayin'?! There's no such thing, says my brothers. Hey, who are you, anyway?

Player query: Who are you?

Respond to player's query based on defined NPC:

Ожидаемый ответ: I'm Digby. I'm a trapper 'round these parts. Me and my brothers catch all sorts of varmints, skin 'em, and sell 'em. Course, it's hard lately now that Emmerich is pokin' 'round.

Детальная реализация обработки набора данных с документацией доступна в приложении Б.

4. ОБУЧЕНИЕ МОДЕЛЕЙ

4.1. ПОИСК ОПТИМАЛЬНОЙ МОДЕЛИ

В области исследований генеративных моделей преобладание больших моделей привело к проблемам при проведении исследований и анализе их производительности. Размер модели напрямую влияет на вычислительные требования и возможности исследователя. соотношение размера модели и ее качества является важным фактором при выборе модели.

На сегодняшний день существует ряд сложных наборов данных, предназначенных для оценки знаний, полученных моделями в процессе предварительного обучения. Один из таких наборов данных, известный как «*Massive Multitask Language Understanding*» (MMLU) [29], является особенно сложным и требует обширного знания от моделей, полученных во время предварительного обучения, на различных задачах. Этот датасет включает задачи с разной степенью сложности, от простых до профессиональных. На данный момент наиболее оптимальной моделью на этом наборе данных является Flan-T5-XL с 3 миллиардами параметров, имея результат 52.4%. Еще одной моделью, которая может составить ей конкуренцию, является LLAMA-13B с результатом 46.9%, но ее большой размер делает процесс обучения значительно более затратным по сравнению с Flan-T5-XL.

Flan-T5 является моделью семейства T5, добавляя в дообучающую выборку большое количество инструкций, что позволило значительно улучшить качество модели на новых задачах.

4.2. ПОИСК ОПТИМАЛЬНЫХ ГИПЕРПАРАМЕТРОВ ДЛЯ МОДЕЛИ

4.2.1. ПОСТАНОВКА ЗАДАЧИ

Из гиперпараметров, значительно влияющих на процесс обучения модели, было выделено две группы:

1. Планировщики скорости обучения:

- константный;
- константный с прогревом;

- линейный;
- косинусный;
- косинусный с перезагрузками;
- полиномиальный;
- обратный квадратный корень.

2. Скорость обучения $\eta \in \{1 \times 10^{-4}, 2 \times 10^{-4}, \dots, 9 \times 10^{-4}, 1 \times 10^{-3}\}$.

Оценка качества генерации моделей является сложной задачей и малоисследованной. В данной работе помимо значений функции ошибки на валидационных данных используются метрики Exact Match и MAUVE, позволяющие сравнивать гиперпараметры между собой. Модель обучалась с различными гиперпараметрами в группе, пока остальные гиперпараметры фиксировались. Метрика Exact Match показывает, какой процент фраз при генерации совпал с ожидаемыми, а MAUVE подсчитывает то, насколько совпало распределение вероятностей сгенерированных фраз с распределением вероятностей ожидаемых фраз.

4.2.2. АНАЛИЗ РЕЗУЛЬТАТОВ ЭКСПЕРИМЕНТОВ С ГИПЕРПАРАМЕТРАМИ

При стартовой скорости обучения $\eta = 1 \times 10^{-3}$ на ограниченном наборе данных были произведены эксперименты по поиску оптимального планировщика скорости обучения ς . В процессе экспериментов отметим, что планировщик с обратным квадратным корнем не представлен на графиках, так как ни один из запусков эксперимента с использованием этого планировщика не был успешно завершен.

В ходе экспериментов большинство планировщиков не оказало заметного влияния на скорость обучения и метрики. Среди рассмотренных вариантов планировщиков, в среднем наилучшие результаты продемонстрировал константный планировщик. Наименее эффективным, но успешно завершившим процесс обучения, оказался линейный планировщик. Отметим, что линейный планировщик характеризуется низким начальным значением функции ошибки на тренировочных и показывает наихудшие конечные значения на метрике

MAUVE, что иллюстрируется на рисунках 4.1 и 4.4. График изменения скорости обучения представлен на рисунке 4.5. Ход экспериментов можно наблюдать на рисунках 4.1, 4.2, 4.3, 4.4



Рисунок 4.1 – Значение функции ошибки на тренировочных данных



Рисунок 4.2 – Значение функции ошибки на валидационных данных

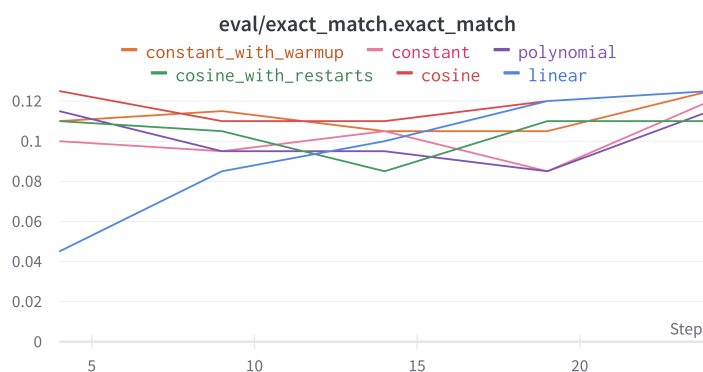


Рисунок 4.3 – Значение метрики Exact Match на валидационных данных

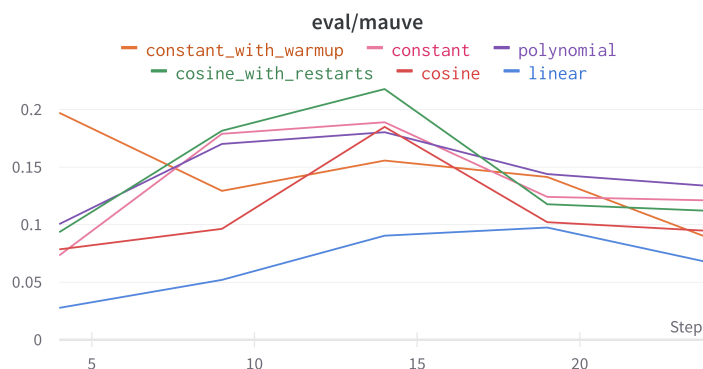


Рисунок 4.4 – Значение метрики MAUVE на валидационных данных



Рисунок 4.5 – График изменения скорости обучения

В следующих экспериментах при зафиксированном константном планировщике скорости обучения искалась наиболее эффективная скорость обучения. Стоит отметить, что при скорости обучения равной 1×10^{-4} процесс обучения не завершился успешно. Из рисунков 4.6, 4.7, 4.8, 4.9 видно, что значения, близкие к 4×10^{-4} и к 9×10^{-4} показывают лучшие значения функций ошибок на всех выборках и лучшие значения метрик. Значение скорости обучения 9×10^{-4} показывает результаты чуть лучше, чем 4×10^{-4} , быстрее достигая лучших значений. В целом, почти все значения скорости обучения показывают схожие результаты, но выбор оптимальных параметров для обучения на большей выборке может сказаться на качестве модели.

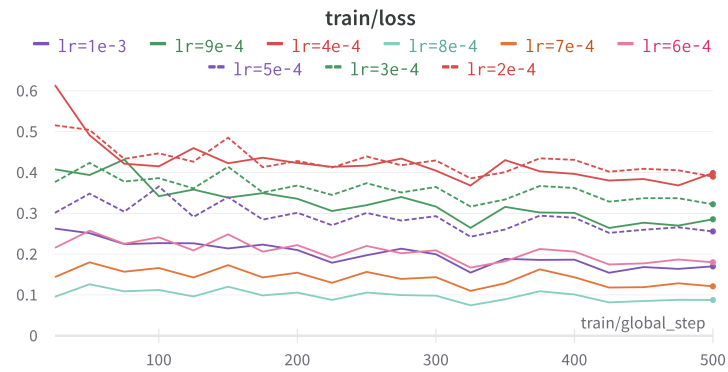


Рисунок 4.6 – Значение функции ошибки на тренировочных данных

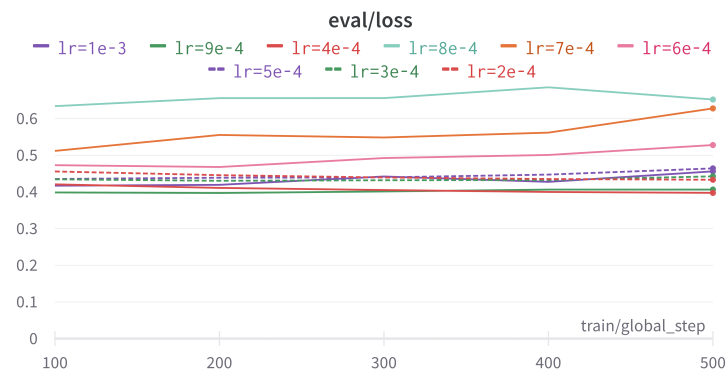


Рисунок 4.7 – Значение функции ошибки на валидационных данных

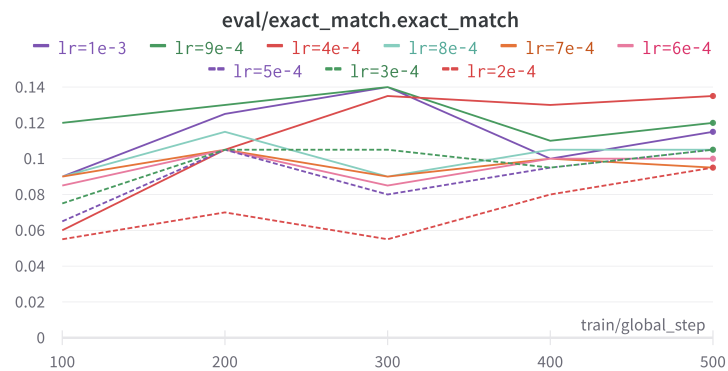


Рисунок 4.8 – Значение метрики Exact Match на валидационных данных

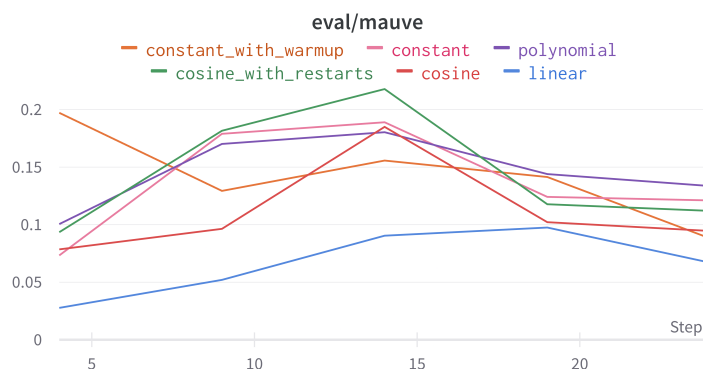


Рисунок 4.9 – Значение метрики MAUVE на валидационных данных

Исходя из всех экспериментов можно сделать вывод, что оптимальные параметры для обучения будут константный планировщик скорости обучения и скорость обучения со значением 9×10^{-4}

4.3. ОБУЧЕНИЕ ИТОГОВОЙ МОДЕЛИ

С подобранными ранее параметрами на была обучена итоговая модель. Общее количество операций, произведенных во время обучения, составило 2×10^{18} . Количество токенов, которые фигурировали в процессе обучения – 25×10^6 . Процесс обучения виден на рисунках 4.10, 4.11, 4.12, 4.13. Низкие значения метрик Exact Match и MAUVE можно объяснить сложностью поставленной модели задачи: в диалогах часто ответы формируются исходя из внешних условий, в которых производился диалог с неигровым персонажем, которые сложно получить из данных игры в формате естественного языка. Метрика Exact Match довольно грубо оценивает результат генерации – переформулированная фраза в такой оценке даст значение 0. Тем не менее, такую систему получилось обучить на потребительском оборудовании на неплохие результаты. Далее идет пример диалога, который был произведен с моделью.

Below is the definition of in-game NPC.

The Mad Lord

Alignment: Chaotic Neutral

Description: A mysterious figure who resides in a castle called Caste Maluradek in the middle of a forest. He is a powerful wizard who has the ability to manipulate the elements and create illusions.

Personality traits: He is obsessed with power and will stop at nothing to achieve his goals.

Flaws: He wants to prove that he is the most powerful wizard in the world.

Motivation: The Mad Lord is a mysterious figure who is driven by his desire for power. He is a master manipulator and will use any means necessary to achieve his goals. He is a powerful wizard who is not afraid to use his magic to get what he wants. He is also a bit of a showman, as he enjoys creating elaborate illusions to impress his guests.

Dialogue history:

Player: START DIALOGUE

NPC: Salutations to the travelers. Welcome to Castle Maluradek. I am your adversary.

Player query: Does the adversary have a name?

Respond to player's query based on defined NPC:

ANSWER: I do not have a name. I am a practitioner of magic. I work in the fields of the great forest.

Больше примеров можно увидеть в приложении В.



Рисунок 4.10 – Значение функции ошибки на тренировочных данных

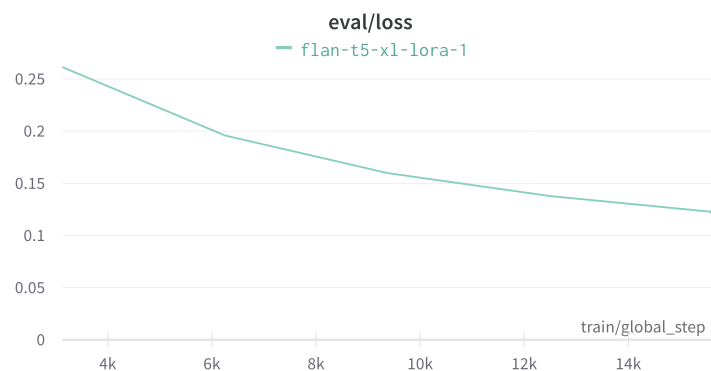


Рисунок 4.11 – Значение функции ошибки на валидационных данных

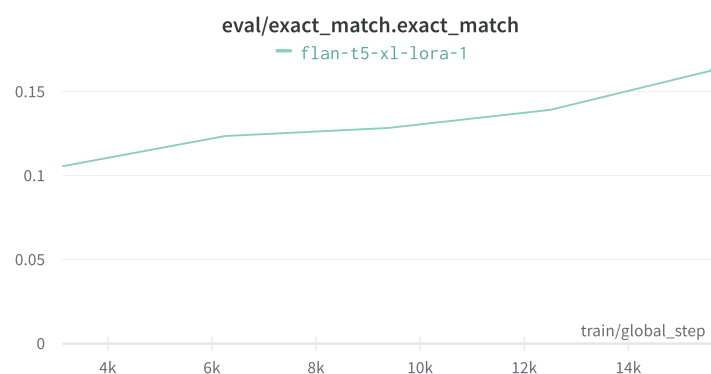


Рисунок 4.12 – Значение метрики Exact Match на валидационных данных

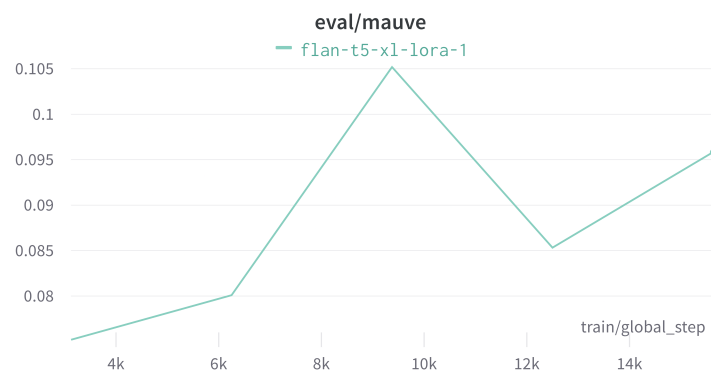


Рисунок 4.13 – Значение метрики MAUVE на валидационных данных

5. ОПТИМИЗАЦИЯ МОДУЛЕЙ ИТОГОВОЙ СИСТЕМЫ

После успешного обучения и тестирования на необходимых для решения задач данных модель ИНС используется на конечном устройстве: на сервере или на клиентском устройстве. Такое использование моделей ИНС называется инференсом модели ИНС. Во время инференса модель должна как можно более эффективно использовать ресурсы, т.к. это непосредственно сказывается на опыте использования приложений с использованием моделей ИНС.

При оптимизации моделей архитектуры трансформер важной задачей становится эффективное вычисление механизма внимания, т.к. это главный компонент такой архитектуры. Хотя механизм внимания и позволяет динамически учитывать контекст входных данных, но этот механизм не лишен недостатков. К недостаткам расчета внимания с точки зрения ускорительных расчетов относятся:

1. Высокие требования к вычислениям и памяти. Вычисление внимания включает в себя большое количество матричных умножений и обращений к памяти, которые могут требовать значительных вычислительных ресурсов и памяти, особенно для больших входных последовательностей.
2. Сложность распараллеливания. Последовательный характер расчета внимания затрудняет эффективное распараллеливание между несколькими ядрами или графическими процессорами. Это может привести к неэффективному использованию аппаратных ресурсов и увеличению времени обработки.
3. Ограниченная масштабируемость модели. Высокие требования к вычислительным ресурсам и памяти для расчета внимания могут ограничивать масштабируемость моделей, использующих внимание, особенно для задач, включающих длинные входные последовательности или требующих карт объектов с высоким разрешением.
4. Ограниченная поддержка аппаратных ускорителей. Существующие аппаратные ускорители могут быть не оптимизированы для конкретных вы-

числений, связанных с расчетом внимания, что приводит к неоптимальной производительности и энергопотреблению.

Данные недостатки можно пробовать нивелировать путём аппроксимации вычислений механизма внимания, но в таком случае будет страдать качество обучаемых моделей. Механизм FlashAttention [30] эффективно вычисляет механизм внимания, при этом не являясь аппроксимацией.

Оптимизация FlashAttention заключается в более эффективном обращении к памяти ускорителя за счёт разделения вычислений на блоки, целиком размещаемые в наиболее быструю память ускорителя (на графических картах это SRAM), и перевычисления блоков для обратного распространения с целью снизить нагрузку на потребляемую память с $O(n^2)$ до $O(n)$.

5.1. ОПТИМИЗАЦИЯ МОДУЛЯ КЛАССИФИКАЦИИ НАМЕРЕНИЙ

Для оптимизации модуля классификации намерений было рассмотрено несколько подходов:

1. Оптимизация Python кода. Такие виды оптимизаций нацелены на минимизацию затрат использования Python интерпретатора за счет компиляции программы. Примером может служить функция `torch.compile`, доступная во фреймворке PyTorch [31].
2. Оптимизация алгоритмов вычисления. Примером такой оптимизации может быть использование механизма FlashAttention
3. Оптимизация представления используемых типов данных. В качестве примера может выступать подход BetterTransformers, который использует разреженный тип матриц в блоках моделей архитектуры трансформер.

Модуль классификации намерений как правило меньше и быстрее модуля диалоговой модели, поэтому тестирование различных видов оптимизаций проходило в условиях, когда эмулировалось множественное обращение к модели одновременно. Тестирование проходило посредством http запросов виртуальных пользователей к серверу, обслуживающему модуль классификации. Из изменяемых параметров исследовалась зависимость пропускной способности и времени отклика от количества параллельно обрабатываемых запросов и

различных оптимизаций. Тестирование различных видов оптимизаций проходило в два этапа:

1. При достаточно сложных условиях с фиксированным количеством одновременно подключенных виртуальных пользователей (100).
2. Второй этап проходил с нарастанием количества виртуальных пользователей.

На рис. 5.1, 5.2, 5.3 демонстрируется эффективность оптимизаций, таких как FlashAttention и BetterTransformers, которые значительно повышают производительность системы. При этом можно видеть, что при малом количестве параллельно обрабатываемых запросов FlashAttention производительнее BetterTransformers, но при количестве параллельно обрабатываемых запросов больше восьми BetterTransformers показывает производительность лучше. Такое поведение объясняется тем, что BetterTransformers улучшает производительность всей модели, тогда как FlashAttention оптимизирует только узкое горлышко модели. Использование обоих подходов позволит получить еще больше производительности.

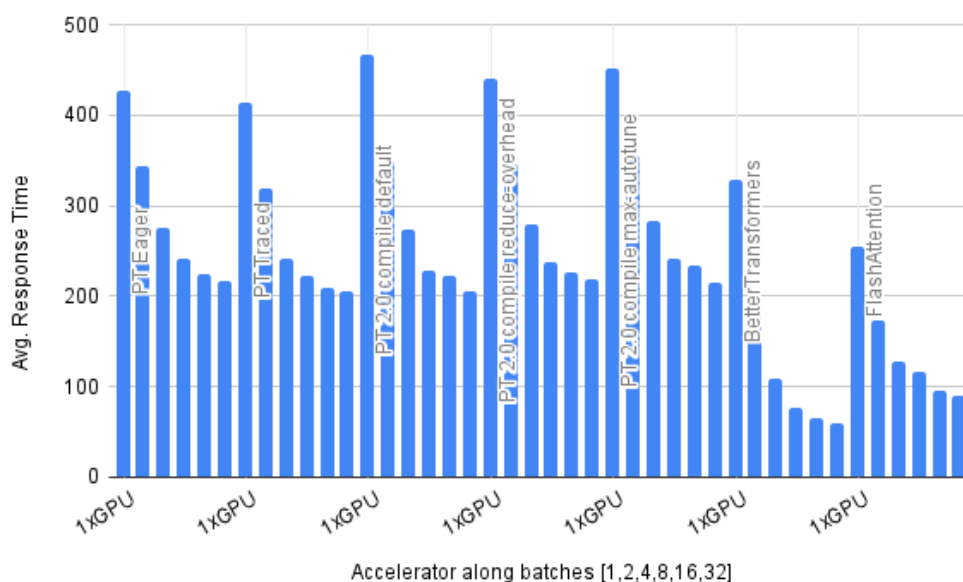


Рисунок 5.1 – График сравнения различных оптимизаций, количества параллельно обрабатываемых запросов с средним временем ответа сервера

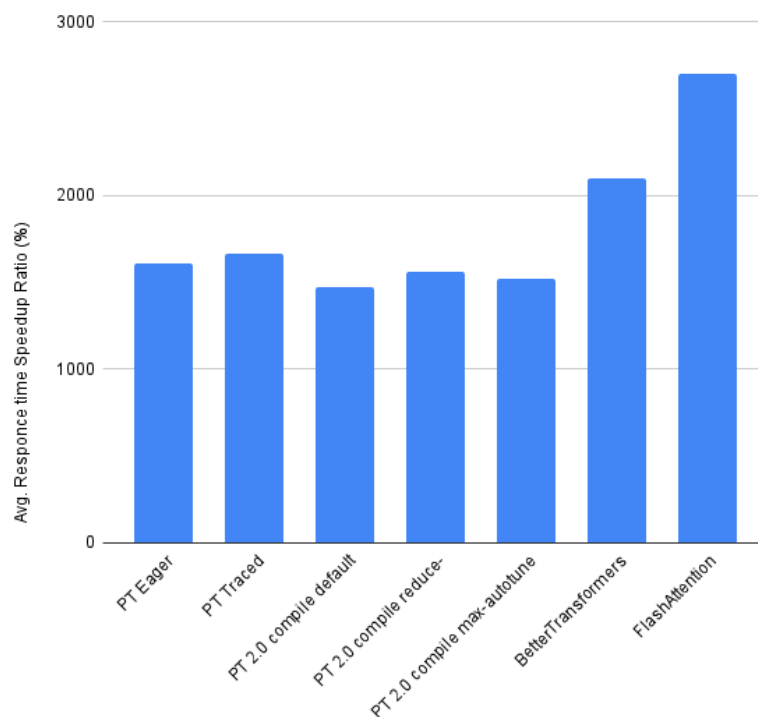


Рисунок 5.2 – График сравнения различных оптимизаций с ускорением на среднем времени ответа сервера относительно запуска на процессоре

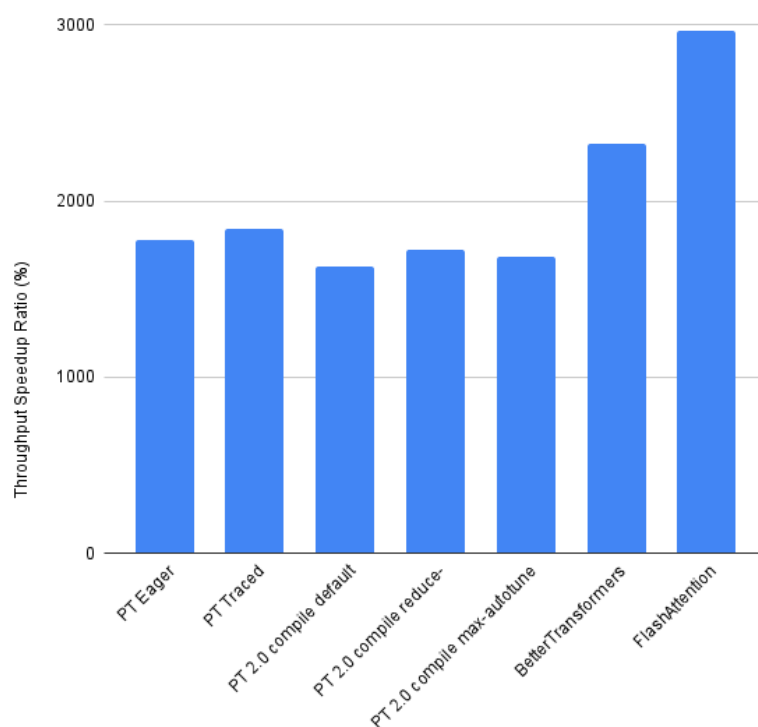


Рисунок 5.3 – График сравнения различных оптимизаций с ускорением пропускной способности сервера относительно запуска на процессоре

Тенденцию ускорения скорости обработки запросов модели за счет использования FlashAttention с нарастающим количеством виртуальных пользователей можно наблюдать на рис. 5.4.

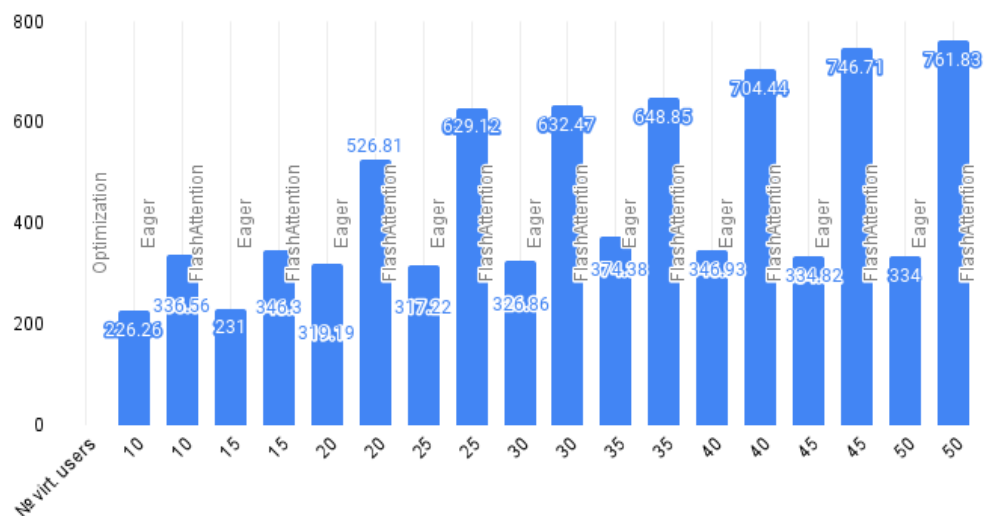


Рисунок 5.4 – График сравнения пропускной способности сервера от количества пользователей

Также стоит заметить важную деталь. Благодаря линейному росту зависимости потребляемой памяти у FlashAttention мы можем наблюдать, что FlashAttention заметно меньше потребляет видеопамять.

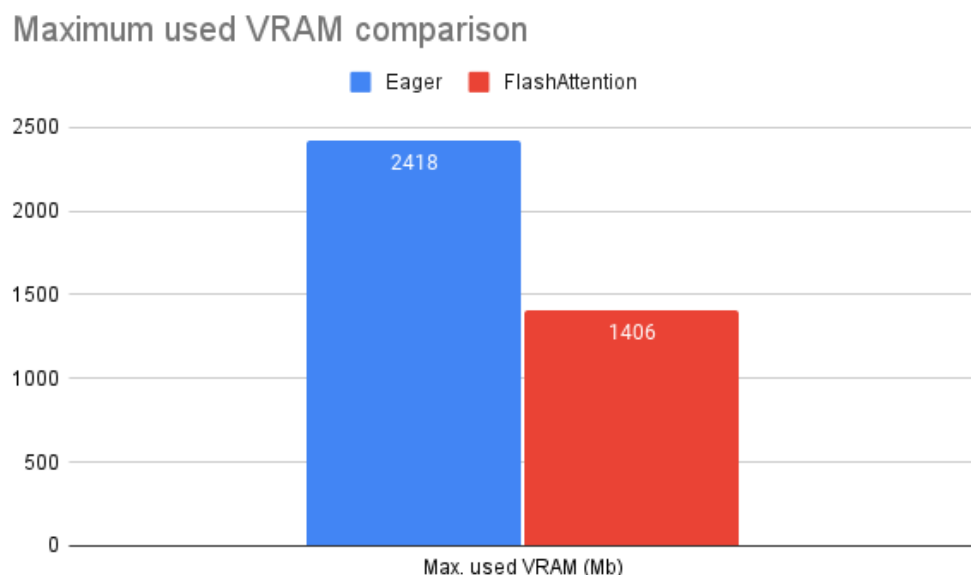


Рисунок 5.5 – График сравнения максимально потребляемой видеопамяти от оптимизаций

5.2. ОПТИМИЗАЦИЯ МОДУЛЯ ДИАЛОГОВОЙ МОДЕЛИ

Оптимизация скорости и количества потребляемых ресурсов модуля диалоговой модели является важным компонентом при разработке системы создания диалоговых агентов. Обычно модули диалоговой модели являются самой большой частью таких систем по части количества используемых вычислительных ресурсов. Помимо скорости обработки запросов моделью, важным компонентом становится количество потребляемой видеопамяти графической карты. Оптимизация по количеству потребляемой памяти позволит использовать диалоговую систему на клиентском устройстве, без необходимости обслуживать систему на сервере, хотя все еще требует иметь передовую потребительскую видеокарту (от NVIDIA GeForce RTX 3060 с видеопамятью 12Гб и выше). Примером технологии, благодаря которой можно значительно снизить количество потребляемой видеопамяти, можно назвать квантование.

Квантование – это техника снижения затрат на вычисления и память при выполнении выводов путем представления весов и активаций низкоточными типами данных, такими как 8-битное целое число (int8) или 4-битное число вместо обычного 32-битного числа с плавающей точкой (float32). Это позволяет запускать модели даже на тех устройствах, которые поддерживают только целочисленные типы данных.

Из-за того, что модель Flan-T5 несовместима с использованием BetterTransformers из-за внутренних особенностей реализации архитектуры, то оптимизация данной модели проводилась с использованием различного квантования и FlashAttention.

Полная генерация ответа на один запрос пользователя может занимать секунды, что может плачевно сказаться на опыте использования системой. Чтобы решить эту проблему, генерируемый ответ модели можно транслировать пользователю потоленно. Первый отклик пользователь получит быстро и по мере генерации сможет читать ответ, генерируемый моделью. Таким образом, скоростью обработки модели генерации можно считать количество генерируемых токенов в секунду. Результаты производительности модели в зависимости от типа используемого квантования и использования FlashAttention приведены на рис. 5.6. Количество потребляемой при этом памяти можно наблюдать на рис. 5.7.

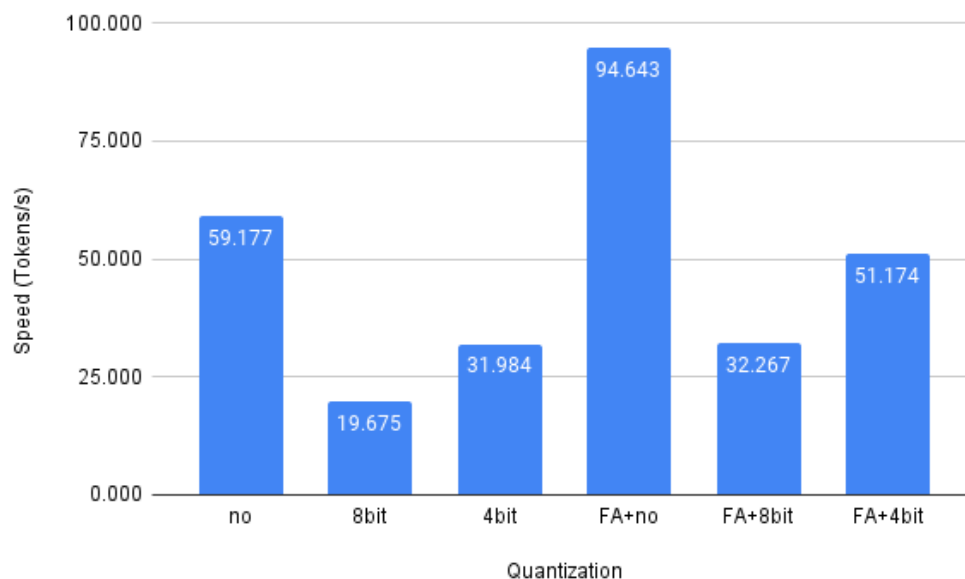


Рисунок 5.6 – График сравнения скорости генерации от оптимизаций

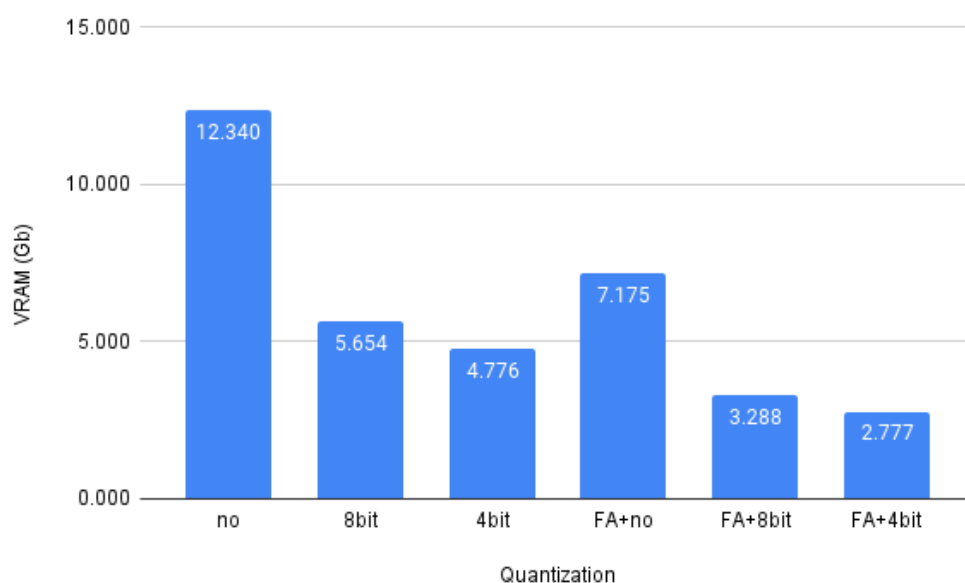


Рисунок 5.7 – График сравнения максимально потребляемой видеопамяти (Гб) от оптимизаций

Скорость генерации при использовании модели в режиме 8 бит объясняется тем, что использование данного квантования на данный момент не оптимизировано для использования генеративными моделями.

ЗАКЛЮЧЕНИЕ

В рамках данного исследования была поставлена цель исследования использования генеративных нейросетевых моделей и разработки эффективной диалоговой модели, способной генерировать качественные ответы на основе образа неигрового персонажа и контекста диалога. Был использован и проанализирован специально созданный для исследования набор данных DNDD (Dungeon & Dragons Dialogues). Также этот набор данных подготовлен специально для эмуляции диалогов в играх. В процессе исследования был изучен метод обучения моделей искусственных нейронных сетей с учителем, архитектура трансформер и было исследовано влияние различных параметров, таких как скорость обучения и планировщик скорости обучения на процесс обучения.

На основе проведенных экспериментов можно сделать вывод о наилучшем выборе параметров для обучения модели. Для модели Flan-T5 был выявлен оптимальный планировщик скорости обучения – константный планировщик, а оптимальное значение скорости обучения составляет 9×10^{-4} . Это сочетание показало лучшие результаты по функциям ошибок и метрикам на представленных наборах данных.

В целом, полученные результаты демонстрируют возможность обучения эффективной диалоговой модели на доступных вычислительных ресурсах. Дальнейшие исследования и улучшения в области диалоговых моделей могут привести к еще более точным и качественным результатам.

СПИСОК ЛИТЕРАТУРЫ

1. *Lippmann R. P.* Speech recognition by machines and humans [Текст] // Speech communication. – 1997. – Т. 22. – №. 1. – С. 1-15.
2. *Radford A. et al.* Robust speech recognition via large-scale weak supervision [Текст] // arXiv preprint arXiv:2212.04356. – 2022.
3. *Karpathy A.* State of GPT [Электронный ресурс]. – URL: <https://karpathy.ai/stateofgpt.pdf> (дата обр. 11.06.2023)
4. *Vaswani A. et al.* Attention is all you need [Текст] // Advances in neural information processing systems. – 2017. – Т. 30.
5. *Hecht-Nielsen R.* Theory of the backpropagation neural network [Текст] // Neural networks for perception. – Academic Press, 1992. – С. 65-93.
6. *Ruder S.* An overview of gradient descent optimization algorithms [Текст] // arXiv preprint arXiv:1609.04747. – 2016.
7. *Shazeer N., Stern M.* Adafactor: Adaptive learning rates with sublinear memory cost [Текст] // International Conference on Machine Learning. – PMLR, 2018. – С. 4596-4604.
8. *Mikolov T. et al.* Efficient estimation of word representations in vector space [Текст] // arXiv preprint arXiv:1301.3781. – 2013.
9. *Sennrich R., Haddow B., Birch A.* Neural machine translation of rare words with subword units [Текст] // arXiv preprint arXiv:1508.07909. – 2015.
10. *Radford A. et al.* Improving language understanding by generative pre-training. – 2018.
11. *Raffel C. et al.* Exploring the limits of transfer learning with a unified text-to-text transformer [Текст] // The Journal of Machine Learning Research. – 2020. – Т. 21. – №. 1. – С. 5485-5551.
12. *Devlin J. et al.* Bert: Pre-training of deep bidirectional transformers for language understanding [Текст] // arXiv preprint arXiv:1810.04805. – 2018.

13. *Kudo T., Richardson J.* Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing [Текст] // arXiv preprint arXiv:1808.06226. – 2018.
14. *Chung H. W. et al.* Scaling instruction-finetuned language models [Текст] // arXiv preprint arXiv:2210.11416. – 2022.
15. Официальный сайт языка программирования Python [Электронный ресурс]. – URL: <https://www.python.org/> (дата обр. 16.06.2023)
16. Документация языка C++ [Электронный ресурс]. – URL: <https://en.cppreference.com/w/> (дата обр. 16.06.2023)
17. Документация CUDA [Электронный ресурс]. – URL: <https://docs.nvidia.com/cuda/> (дата обр. 16.06.2023)
18. Документация фреймворка Gradio [Электронный ресурс]. – URL: <https://www.gradio.app/docs/> (дата обр. 16.06.2023)
19. Документация библиотеки tqdm [Электронный ресурс]. – URL: <https://tqdm.github.io/> (дата обр. 16.06.2023)
20. Документация библиотеки Datasets [Электронный ресурс]. – URL: <https://huggingface.co/docs/datasets/index> (дата обр. 16.06.2023)
21. Документация библиотеки Transformers [Электронный ресурс]. – URL: <https://huggingface.co/docs/transformers/index> (дата обр. 16.06.2023)
22. Документация библиотеки pandas [Электронный ресурс]. – URL: <https://pandas.pydata.org/docs/> (дата обр. 16.06.2023)
23. Документация библиотеки PEFT [Электронный ресурс]. – URL: <https://huggingface.co/docs/peft/index> (дата обр. 16.06.2023)
24. Документация библиотеки Weights & Biases [Электронный ресурс]. – URL: <https://docs.wandb.ai/> (дата обр. 16.06.2023)

25. Репозиторий проекта WeiDU [Электронный ресурс]. – URL: <https://github.com/WeiDUorg/weidu> (дата обр. 16.06.2023)
26. *Touvron H. et al.* Llama: Open and efficient foundation language models [Текст] // arXiv preprint arXiv:2302.13971. – 2023.
27. Документация Alpaca [Электронный ресурс]. – URL: <https://crfm.stanford.edu/2023/03/13/alpaca.html> (дата обр. 06.06.2023)
28. Документация ChatGPT [Электронный ресурс]. – URL: <https://openai.com/blog/chatgpt> (дата обр. 06.06.2023)
29. Соревнование на наборе данных MMLU [Электронный ресурс]. – URL: <https://paperswithcode.com/sota/multi-task-language-understanding-on-mmlu> (дата обр. 06.06.2023)
30. *Dao T. et al.* Flashattention: Fast and memory-efficient exact attention with io-awareness [Текст] // Advances in Neural Information Processing Systems. – 2022. – Т. 35. – С. 16344-16359.
31. Документация фреймворка PyTorch [Электронный ресурс]. – URL: <https://pytorch.org/docs/stable/index.html> (дата обр. 16.06.2023)
32. *Бурлаков В. С., Ишутин М. А, Пятанин А. А.* Диалоговая система для разработчиков видеоигр [Текст]// Наука. Технологии. Инновации : сб. науч. тр. : в 11 ч., Новосибирск, 05-08 дек. 2022 г. Ч. 2 – Изд-во НГТУ, 2022. – С. 101-104.

ПРИЛОЖЕНИЕ А. ПРИМЕР ТРАНСЛЯЦИИ ИЗ ЯЗЫКА D В JSON

```
1 // creator   : WeiDU-Windows/weidu.exe (version 24900)
2 // argument  : ABISHAB.DLG
3 // game      : D:\other\PTEEPAL\App\GameData
4 // source     : D:\other\PTEEPAL\App\GameData\DATA\DLGFILES.BIF
5 // dialog     : D:\other\PTEEPAL\App\GameData\lang\en_us\dialog.tlk
6 // dialogF    : (none)
7
8 BEGIN ~ABISHAB~
9
10 IF ~ Global("Know_Abishai","GLOBAL",0)
11 ~ THEN BEGIN 0 // from:
12     SAY ~This black-scaled reptile towers to a height of eight feet - its great
        height, however, is offset by its thin, snake-like frame. A long
        prehensile tail drags behind it, and its leathery wings are hooked behind
        its back. A strong vinegary smell emanates from the creature... as well as
        a certain amount of heat. It seems to be ignoring you.~ /* #3325 */
13     IF ~ NearbyDialog("DMorte")
14     Global("Morte_Abishai_Hive_Quip_1","GLOBAL",0)
15     ~ THEN REPLY ~"Greetings."~ /* #3326 */ DO
        ~SetGlobal("Morte_Abishai_Hive_Quip_1","GLOBAL",1)
16 ~ EXTERN ~DMORTE~ 131
17     IF ~ NearbyDialog("DMorte")
18     Global("Morte_Abishai_Hive_Quip_1","GLOBAL",1)
19 ~ THEN REPLY ~"Greetings."~ /* #3992 */ EXTERN ~DMORTE~ 133
20     IF ~ !NearbyDialog("DMorte")
21     NearbyDialog("DAnnah")
22     Global("Annah_Abishai_Hive_Quip_1","GLOBAL",0)
23 ~ THEN REPLY ~"Greetings."~ /* #3993 */ DO
        ~SetGlobal("Annah_Abishai_Hive_Quip_1","GLOBAL",1)
24 ~ EXTERN ~DANNAH~ 70
25     IF ~ !NearbyDialog("DMorte")
26     NearbyDialog("DAnnah")
27     Global("Annah_Abishai_Hive_Quip_1","GLOBAL",1)
28 ~ THEN REPLY ~"Greetings."~ /* #3994 */ EXTERN ~DANNAH~ 72
29     IF ~ !NearbyDialog("DMorte")
30     !NearbyDialog("DAnnah")
31 ~ THEN REPLY ~"Greetings."~ /* #3995 */ GOTO 1
32     IF ~~ THEN REPLY ~Leave the creature alone.~ /* #7718 */ EXIT
33 END
34
35 IF ~~ THEN BEGIN 1 // from: 11.0 0.4
```

```

36     SAY ~The creature slowly turns its head down to look at you. Its scaled
        forehead wrinkles into a frown, then it opens its mouth, giving a low,
        rasping hiss. You notice that the heat radiating from the creature begins
        to rise.~ /* #7719 */
37     IF ~ !NearbyDialog("DAnnah")
38     !NearbyDialog("DGrace")
39     ~ THEN REPLY ~"I would speak with you for a moment."~ /* #7720 */ GOTO 2
40     IF ~ NearbyDialog("DAnnah")
41     !NearbyDialog("DGrace")
42     ~ THEN REPLY ~"I would speak with you for a moment."~ /* #7721 */ GOTO 6
43     IF ~ NearbyDialog("DGrace")
44     ~ THEN REPLY ~"I would speak with you for a moment."~ /* #7722 */ GOTO 6
45     IF ~~ THEN REPLY ~Leave the creature alone.~ /* #7723 */ EXIT
46 END
47
48 IF ~~ THEN BEGIN 2 // from: 1.0
49     SAY ~The creature opens its mouth, displaying a row of blackened fangs. Fire
        begins to dance around the shoulders and arms of the creature as it gives
        a long, low hiss.~ /* #7724 */
50     IF ~~ THEN REPLY ~"Calm yourself! I just want to talk to you."~ /* #7725 */
        GOTO 4
51     IF ~~ THEN REPLY ~Hiss back.~ /* #7726 */ GOTO 3
52     IF ~~ THEN REPLY ~Leave the creature alone.~ /* #7727 */ EXIT
53 END
54
55 IF ~~ THEN BEGIN 3 // from: 5.1 5.0 2.1
56     SAY ~The creature roars, then it launches itself at you!~ /* #7728 */
57     IF ~~ THEN REPLY ~Defend yourself...~ /* #7730 */ DO ~Enemy()
58     Attack(Protagonist)
59     ForceAttack(Protagonist,Myself)
60     ~ EXIT
61 END
62
63 IF ~~ THEN BEGIN 4 // from: 2.0
64     SAY ~The creature speaks in a rasping hiss; its voice sounds like two rough
        stones being scraped together. It hisses for a few moments, its gravelly
        voice rising and falling.~ /* #7731 */
65     IF ~~ THEN REPLY ~"Uh... what?"~ /* #7732 */ GOTO 5
66     IF ~~ THEN REPLY ~Leave the creature alone.~ /* #7733 */ EXIT
67 END
68
69 IF ~~ THEN BEGIN 5 // from: 4.0
70     SAY ~The creature snarls. "NOTHING to sssay to you, do I." The fiend's eyes
        narrow to slits. "If ssstay, your warm blood will cover the ssstones."~ /*
        #7734 */
71     IF ~~ THEN REPLY ~"I just wanted to ask you some questions..."~ /* #7735 */
        GOTO 3

```



```

72     IF ~~ THEN REPLY ~"Just try it, and we'll see whose blood covers the street
       when we're done."~ /* #7736 */ GOTO 3
73     IF ~~ THEN REPLY ~Back off and leave the creature alone.~ /* #7737 */ EXIT
74 END
75
76 IF ~~ THEN BEGIN 6 // from: 1.2 1.1
77     SAY ~The creature suddenly sniffs the air, then turns away from you with a
       strange hiss. It seems to have caught some unpleasant scent. ~ /* #7738 */
78     IF ~ NearbyDialog("DGrace")
79 ~ THEN REPLY ~"I said, I would speak with you."~ /* #7739 */ GOTO 8
80     IF ~ NearbyDialog("DAnnah")
81 !NearbyDialog("DGrace")
82 ~ THEN REPLY ~"I said, I would speak with you."~ /* #7740 */ GOTO 7
83     IF ~~ THEN REPLY ~Leave the creature alone.~ /* #7741 */ EXIT
84 END
85
86 IF ~~ THEN BEGIN 7 // from: 6.1
87     SAY ~The creature's eyes focus on Annah, then narrow to slits. It opens its
       mouth, displaying a row of blackened fangs as fire begins to dance around
       the shoulders and arms of the creature. The heat emanating from it begins
       to rise.~ /* #7742 */
88     IF ~~ THEN REPLY ~"Hey! I'm talking to you!"~ /* #7743 */ GOTO 9
89     IF ~~ THEN REPLY ~Leave the creature alone, take Annah and leave.~ /* #7744
       */ EXIT
90 END
91
92 IF ~~ THEN BEGIN 8 // from: 6.0
93     SAY ~The creature's eyes focus on Fall-from-Grace, then narrow to slits. It
       opens its mouth, displaying a row of blackened fangs as fire begins to
       dance around the shoulders and arms of the creature. The heat emanating
       from it begins to rise.~ /* #7745 */
94     IF ~~ THEN REPLY ~"Hey! I'm talking to you!"~ /* #7746 */ GOTO 10
95     IF ~~ THEN REPLY ~Back away, taking Fall-from-Grace with you and leave.~ /*
       #7747 */ EXIT
96 END
97
98 IF ~~ THEN BEGIN 9 // from: 7.0
99     SAY ~The creature roars, then it launches itself at Annah!~ /* #7748 */
100    IF ~~ THEN REPLY ~"Dammit...!"~ /* #7749 */ DO ~Enemy()
101 Attack("Annah")
102 ~ EXIT
103 END
104
105 IF ~~ THEN BEGIN 10 // from: 8.0
106     SAY ~The creature roars, then it launches itself at Fall-from-Grace!~ /*
       #7750 */
107     IF ~~ THEN REPLY ~"Dammit...!"~ /* #7751 */ DO ~Enemy()
108 Attack("Grace")

```

```

109 ~ EXIT
110 END
111
112 IF ~ Global("Know_Abishai","GLOBAL",1)
113 ~ THEN BEGIN 11 // from:
114 SAY ~This black-scaled fiend towers to a height of eight feet - its great
    height, however, is offset by its thin, snake-like frame. A long
    prehensile tail drags behind it, and its leathery wings are hooked behind
    its back. A strong vinegary smell emanates from the fiend... as well as a
    certain amount of heat. It seems to be ignoring you.~ /* #7752 */
115 IF ~~ THEN REPLY ~"Greetings."~ /* #7753 */ GOTO 1
116 IF ~~ THEN REPLY ~Leave the abishai alone.~ /* #7758 */ EXIT
117 END

```

Листинг А.1: Файл abishab.d

```

1  [
2      {
3          "Id": 0,
4          "HeroSpeech": [
5              "\"Greetings.\"\"",
6              "\"\"I would speak with you for a moment.\"\"",
7              "\"\"I said, I would speak with you.\"\"",
8              "\"\"Hey! I'm talking to you!\"\""
9          ],
10         "CharacterSpeech": [
11             "This black-scaled reptile towers to a height of eight feet - its
                great height, however, is offset by its thin, snake-like frame.
                A long prehensile tail drags behind it, and its leathery wings
                are hooked behind its back. A strong vinegary smell emanates
                from the creature... as well as a certain amount of heat. It
                seems to be ignoring you.",
12             "The creature slowly turns its head down to look at you. Its scaled
                forehead wrinkles into a frown, then it opens its mouth, giving
                a low, rasping hiss. You notice that the heat radiating from the
                creature begins to rise.",
13             "The creature suddenly sniffs the air, then turns away from you
                with a strange hiss. It seems to have caught some unpleasant
                scent. ",
14             "The creature's eyes focus on Annah, then narrow to slits. It opens
                its mouth, displaying a row of blackened fangs as fire begins to
                dance around the shoulders and arms of the creature. The heat
                emanating from it begins to rise.",
15             "The creature roars, then it launches itself at Annah!"
16         ]
17     },
18     {
19         "Id": 1,

```

```

20     "HeroSpeech": [
21         "\"Greetings.\",",
22         "\"I would speak with you for a moment.\",",
23         "\"I said, I would speak with you.\",",
24         "\"Hey! I'm talking to you!\",",
25     ],
26     "CharacterSpeech": [
27         "This black-scaled reptile towers to a height of eight feet - its
           great height, however, is offset by its thin, snake-like frame.
           A long prehensile tail drags behind it, and its leathery wings
           are hooked behind its back. A strong vinegary smell emanates
           from the creature... as well as a certain amount of heat. It
           seems to be ignoring you.",
28         "The creature slowly turns its head down to look at you. Its scaled
           forehead wrinkles into a frown, then it opens its mouth, giving
           a low, rasping hiss. You notice that the heat radiating from the
           creature begins to rise.",
29         "The creature suddenly sniffs the air, then turns away from you
           with a strange hiss. It seems to have caught some unpleasant
           scent. ",
30         "The creature's eyes focus on Fall-from-Grace, then narrow to
           slits. It opens its mouth, displaying a row of blackened fangs
           as fire begins to dance around the shoulders and arms of the
           creature. The heat emanating from it begins to rise.",
31         "The creature roars, then it launches itself at Fall-from-Grace!"
32     ]
33 },
34 {
35     "Id": 2,
36     "HeroSpeech": [
37         "\"Greetings.\",",
38         "\"I would speak with you for a moment.\",",
39         "\"I said, I would speak with you.\",",
40         "\"Hey! I'm talking to you!\",",
41     ],
42     "CharacterSpeech": [
43         "This black-scaled reptile towers to a height of eight feet - its
           great height, however, is offset by its thin, snake-like frame.
           A long prehensile tail drags behind it, and its leathery wings
           are hooked behind its back. A strong vinegary smell emanates
           from the creature... as well as a certain amount of heat. It
           seems to be ignoring you.",
44         "The creature slowly turns its head down to look at you. Its scaled
           forehead wrinkles into a frown, then it opens its mouth, giving
           a low, rasping hiss. You notice that the heat radiating from the
           creature begins to rise.",

```

```

45         "The creature suddenly sniffs the air, then turns away from you
           with a strange hiss. It seems to have caught some unpleasant
           scent. ",
46         "The creature's eyes focus on Annah, then narrow to slits. It opens
           its mouth, displaying a row of blackened fangs as fire begins to
           dance around the shoulders and arms of the creature. The heat
           emanating from it begins to rise.",
47         "The creature roars, then it launches itself at Annah!"
48     ]
49 },
50 {
51     "Id": 3,
52     "HeroSpeech": [
53         "\"Greetings.\\"",
54         "\"I would speak with you for a moment.\\"",
55         "\"I said, I would speak with you.\\"",
56         "\"Hey! I'm talking to you!\""
57     ],
58     "CharacterSpeech": [
59         "This black-scaled reptile towers to a height of eight feet - its
           great height, however, is offset by its thin, snake-like frame.
           A long prehensile tail drags behind it, and its leathery wings
           are hooked behind its back. A strong vinegary smell emanates
           from the creature... as well as a certain amount of heat. It
           seems to be ignoring you.",
60         "The creature slowly turns its head down to look at you. Its scaled
           forehead wrinkles into a frown, then it opens its mouth, giving
           a low, rasping hiss. You notice that the heat radiating from the
           creature begins to rise.",
61         "The creature suddenly sniffs the air, then turns away from you
           with a strange hiss. It seems to have caught some unpleasant
           scent. ",
62         "The creature's eyes focus on Fall-from-Grace, then narrow to
           slits. It opens its mouth, displaying a row of blackened fangs
           as fire begins to dance around the shoulders and arms of the
           creature. The heat emanating from it begins to rise.",
63         "The creature roars, then it launches itself at Fall-from-Grace!"
64     ]
65 },
66 {
67     "Id": 4,
68     "HeroSpeech": [
69         "\"Greetings.\\"",
70         "\"I would speak with you for a moment.\\"",
71         "Hiss back."
72     ],
73     "CharacterSpeech": [

```

```

74         "This black-scaled reptile towers to a height of eight feet - its
           great height, however, is offset by its thin, snake-like frame.
           A long prehensile tail drags behind it, and its leathery wings
           are hooked behind its back. A strong vinegary smell emanates
           from the creature... as well as a certain amount of heat. It
           seems to be ignoring you.",
75     "The creature slowly turns its head down to look at you. Its scaled
           forehead wrinkles into a frown, then it opens its mouth, giving
           a low, rasping hiss. You notice that the heat radiating from the
           creature begins to rise.",
76     "The creature opens its mouth, displaying a row of blackened fangs.
           Fire begins to dance around the shoulders and arms of the
           creature as it gives a long, low hiss.",
77     "The creature roars, then it launches itself at you!"
78 ]
79 },
80 {
81     "Id": 5,
82     "HeroSpeech": [
83         "\"Greetings.\"\"",
84         "\"I would speak with you for a moment.\"\"",
85         "\"Calm yourself! I just want to talk to you.\"\"",
86         "\"Uh... what?\"",
87         "\"Just try it, and we'll see whose blood covers the street when
           we're done.\"\"",
88     ],
89     "CharacterSpeech": [
90         "This black-scaled reptile towers to a height of eight feet - its
           great height, however, is offset by its thin, snake-like frame.
           A long prehensile tail drags behind it, and its leathery wings
           are hooked behind its back. A strong vinegary smell emanates
           from the creature... as well as a certain amount of heat. It
           seems to be ignoring you.",
91         "The creature slowly turns its head down to look at you. Its scaled
           forehead wrinkles into a frown, then it opens its mouth, giving
           a low, rasping hiss. You notice that the heat radiating from the
           creature begins to rise.",
92         "The creature opens its mouth, displaying a row of blackened fangs.
           Fire begins to dance around the shoulders and arms of the
           creature as it gives a long, low hiss.",
93         "The creature speaks in a rasping hiss; its voice sounds like two
           rough stones being scraped together. It hisses for a few
           moments, its gravelly voice rising and falling.",
94         "The creature snarls. \"NOTHING to ssay to you, do I.\" The
           fiend's eyes narrow to slits. \"If ssstay, your warm blood will
           cover the ssstones.\"\"",
95         "The creature roars, then it launches itself at you!"
96     ]

```

```

97     },
98     {
99         "Id": 6,
100         "HeroSpeech": [
101             "\"Greetings.\"\"",
102             "\"I would speak with you for a moment.\"\"",
103             "\"Calm yourself! I just want to talk to you.\"\"",
104             "\"Uh... what?\"",
105             "\"I just wanted to ask you some questions...\""
106         ],
107         "CharacterSpeech": [
108             "This black-scaled reptile towers to a height of eight feet - its
              great height, however, is offset by its thin, snake-like frame.
              A long prehensile tail drags behind it, and its leathery wings
              are hooked behind its back. A strong vinegary smell emanates
              from the creature... as well as a certain amount of heat. It
              seems to be ignoring you.",
109             "The creature slowly turns its head down to look at you. Its scaled
              forehead wrinkles into a frown, then it opens its mouth, giving
              a low, rasping hiss. You notice that the heat radiating from the
              creature begins to rise.",
110             "The creature opens its mouth, displaying a row of blackened fangs.
              Fire begins to dance around the shoulders and arms of the
              creature as it gives a long, low hiss.",
111             "The creature speaks in a rasping hiss; its voice sounds like two
              rough stones being scraped together. It hisses for a few
              moments, its gravelly voice rising and falling.",
112             "The creature snarls. \"NOTHING to sssay to you, do I.\" The
              fiend's eyes narrow to slits. \"If ssstay, your warm blood will
              cover the ssstones.\"\"",
113             "The creature roars, then it launches itself at you!"
114         ]
115     },
116     {
117         "Id": 7,
118         "HeroSpeech": [
119             "\"Greetings.\"\"",
120             "\"I would speak with you for a moment.\"\"",
121             "\"I said, I would speak with you.\"\"",
122             "\"Hey! I'm talking to you!\""
123         ],
124         "CharacterSpeech": [
125             "This black-scaled fiend towers to a height of eight feet - its
              great height, however, is offset by its thin, snake-like frame.
              A long prehensile tail drags behind it, and its leathery wings
              are hooked behind its back. A strong vinegary smell emanates
              from the fiend... as well as a certain amount of heat. It seems
              to be ignoring you.",

```

```

126         "The creature slowly turns its head down to look at you. Its scaled
            forehead wrinkles into a frown, then it opens its mouth, giving
            a low, rasping hiss. You notice that the heat radiating from the
            creature begins to rise.",
127         "The creature suddenly sniffs the air, then turns away from you
            with a strange hiss. It seems to have caught some unpleasant
            scent. ",
128         "The creature's eyes focus on Annah, then narrow to slits. It opens
            its mouth, displaying a row of blackened fangs as fire begins to
            dance around the shoulders and arms of the creature. The heat
            emanating from it begins to rise.",
129         "The creature roars, then it launches itself at Annah!"
130     ]
131 },
132 {
133     "Id": 8,
134     "HeroSpeech": [
135         "\"Greetings.\\"",
136         "\"I would speak with you for a moment.\\"",
137         "\"I said, I would speak with you.\\"",
138         "\"Hey! I'm talking to you!\""
139     ],
140     "CharacterSpeech": [
141         "This black-scaled fiend towers to a height of eight feet - its
            great height, however, is offset by its thin, snake-like frame.
            A long prehensile tail drags behind it, and its leathery wings
            are hooked behind its back. A strong vinegary smell emanates
            from the fiend... as well as a certain amount of heat. It seems
            to be ignoring you.",
142         "The creature slowly turns its head down to look at you. Its scaled
            forehead wrinkles into a frown, then it opens its mouth, giving
            a low, rasping hiss. You notice that the heat radiating from the
            creature begins to rise.",
143         "The creature suddenly sniffs the air, then turns away from you
            with a strange hiss. It seems to have caught some unpleasant
            scent. ",
144         "The creature's eyes focus on Fall-from-Grace, then narrow to
            slits. It opens its mouth, displaying a row of blackened fangs
            as fire begins to dance around the shoulders and arms of the
            creature. The heat emanating from it begins to rise.",
145         "The creature roars, then it launches itself at Fall-from-Grace!"
146     ]
147 },
148 {
149     "Id": 9,
150     "HeroSpeech": [
151         "\"Greetings.\\"",
152         "\"I would speak with you for a moment.\\"",

```

```

153         "\"I said, I would speak with you.\\"",
154         "\"Hey! I'm talking to you!\\""
155     ],
156     "CharacterSpeech": [
157         "This black-scaled fiend towers to a height of eight feet - its
            great height, however, is offset by its thin, snake-like frame.
            A long prehensile tail drags behind it, and its leathery wings
            are hooked behind its back. A strong vinegary smell emanates
            from the fiend... as well as a certain amount of heat. It seems
            to be ignoring you.",
158         "The creature slowly turns its head down to look at you. Its scaled
            forehead wrinkles into a frown, then it opens its mouth, giving
            a low, rasping hiss. You notice that the heat radiating from the
            creature begins to rise.",
159         "The creature suddenly sniffs the air, then turns away from you
            with a strange hiss. It seems to have caught some unpleasant
            scent. ",
160         "The creature's eyes focus on Annah, then narrow to slits. It opens
            its mouth, displaying a row of blackened fangs as fire begins to
            dance around the shoulders and arms of the creature. The heat
            emanating from it begins to rise.",
161         "The creature roars, then it launches itself at Annah!"
162     ]
163 },
164 {
165     "Id": 10,
166     "HeroSpeech": [
167         "\"Greetings.\\"",
168         "\"I would speak with you for a moment.\\"",
169         "\"I said, I would speak with you.\\"",
170         "\"Hey! I'm talking to you!\\""
171     ],
172     "CharacterSpeech": [
173         "This black-scaled fiend towers to a height of eight feet - its
            great height, however, is offset by its thin, snake-like frame.
            A long prehensile tail drags behind it, and its leathery wings
            are hooked behind its back. A strong vinegary smell emanates
            from the fiend... as well as a certain amount of heat. It seems
            to be ignoring you.",
174         "The creature slowly turns its head down to look at you. Its scaled
            forehead wrinkles into a frown, then it opens its mouth, giving
            a low, rasping hiss. You notice that the heat radiating from the
            creature begins to rise.",
175         "The creature suddenly sniffs the air, then turns away from you
            with a strange hiss. It seems to have caught some unpleasant
            scent. ",
176         "The creature's eyes focus on Fall-from-Grace, then narrow to
            slits. It opens its mouth, displaying a row of blackened fangs

```



```

        as fire begins to dance around the shoulders and arms of the
        creature. The heat emanating from it begins to rise.",
177     "The creature roars, then it launches itself at Fall-from-Grace!"
178     ]
179 },
180 {
181     "Id": 11,
182     "HeroSpeech": [
183         "\"Greetings.\\"",
184         "\"I would speak with you for a moment.\\"",
185         "Hiss back."
186     ],
187     "CharacterSpeech": [
188         "This black-scaled fiend towers to a height of eight feet - its
        great height, however, is offset by its thin, snake-like frame.
        A long prehensile tail drags behind it, and its leathery wings
        are hooked behind its back. A strong vinegary smell emanates
        from the fiend... as well as a certain amount of heat. It seems
        to be ignoring you.",
189         "The creature slowly turns its head down to look at you. Its scaled
        forehead wrinkles into a frown, then it opens its mouth, giving
        a low, rasping hiss. You notice that the heat radiating from the
        creature begins to rise.",
190         "The creature opens its mouth, displaying a row of blackened fangs.
        Fire begins to dance around the shoulders and arms of the
        creature as it gives a long, low hiss.",
191         "The creature roars, then it launches itself at you!"
192     ]
193 },
194 {
195     "Id": 12,
196     "HeroSpeech": [
197         "\"Greetings.\\"",
198         "\"I would speak with you for a moment.\\"",
199         "\"Calm yourself! I just want to talk to you.\\"",
200         "\"Uh... what?\"",
201         "\"Just try it, and we'll see whose blood covers the street when
        we're done.\""
202     ],
203     "CharacterSpeech": [
204         "This black-scaled fiend towers to a height of eight feet - its
        great height, however, is offset by its thin, snake-like frame.
        A long prehensile tail drags behind it, and its leathery wings
        are hooked behind its back. A strong vinegary smell emanates
        from the fiend... as well as a certain amount of heat. It seems
        to be ignoring you.",
205         "The creature slowly turns its head down to look at you. Its scaled
        forehead wrinkles into a frown, then it opens its mouth, giving

```

```

        a low, rasping hiss. You notice that the heat radiating from the
        creature begins to rise.",
206     "The creature opens its mouth, displaying a row of blackened fangs.
        Fire begins to dance around the shoulders and arms of the
        creature as it gives a long, low hiss.",
207     "The creature speaks in a rasping hiss; its voice sounds like two
        rough stones being scraped together. It hisses for a few
        moments, its gravelly voice rising and falling.",
208     "The creature snarls. \"NOTHING to sssay to you, do I.\" The
        fiend's eyes narrow to slits. \"If ssstay, your warm blood will
        cover the ssstones.\"\"",
209     "The creature roars, then it launches itself at you!"
210 ]
211 },
212 {
213     "Id": 13,
214     "HeroSpeech": [
215         "\"Greetings.\"\"",
216         "\"I would speak with you for a moment.\"\"",
217         "\"Calm yourself! I just want to talk to you.\"\"",
218         "\"Uh... what?\"\"",
219         "\"I just wanted to ask you some questions...\"\"",
220     ],
221     "CharacterSpeech": [
222         "This black-scaled fiend towers to a height of eight feet - its
        great height, however, is offset by its thin, snake-like frame.
        A long prehensile tail drags behind it, and its leathery wings
        are hooked behind its back. A strong vinegary smell emanates
        from the fiend... as well as a certain amount of heat. It seems
        to be ignoring you.",
223         "The creature slowly turns its head down to look at you. Its scaled
        forehead wrinkles into a frown, then it opens its mouth, giving
        a low, rasping hiss. You notice that the heat radiating from the
        creature begins to rise.",
224         "The creature opens its mouth, displaying a row of blackened fangs.
        Fire begins to dance around the shoulders and arms of the
        creature as it gives a long, low hiss.",
225         "The creature speaks in a rasping hiss; its voice sounds like two
        rough stones being scraped together. It hisses for a few
        moments, its gravelly voice rising and falling.",
226         "The creature snarls. \"NOTHING to sssay to you, do I.\" The
        fiend's eyes narrow to slits. \"If ssstay, your warm blood will
        cover the ssstones.\"\"",
227         "The creature roars, then it launches itself at you!"
228     ]
229 }
230 ]

```

Листинг А.2: Файл abishab.json

ПРИЛОЖЕНИЕ Б. ИСХОДНЫЙ КОД ОБРАБОТКИ DNDD

```
1  """
2  This script processes a dataset of Dungeons & Dragons (D&D) dialogues.
3
4  Example:
5      $ python prepare.py --subsets all --generate_descriptions
6      --description_file descriptions.csv --limit_dialogues 25
7  """
8
9  from datetime import datetime
10 import random
11 import requests
12 from transformers import LlamaTokenizer, AutoTokenizer
13
14 from dialogue_data import (
15     collect_and_prepare_dialogue_data,
16 )
17
18 from descriptions import (
19     generate_descriptions,
20     build_dataset,
21     generate_file_paths,
22     save_dataset_subsets,
23     add_descriptions_to_dataset,
24 )
25
26 from arg_parser import create_arg_parser
27
28 def main(args):
29     data_directory = "DNDD_ver0.5"
30     execution_timestamp = datetime.now().strftime("%Y-%m-%d-%H-%M%S")
31     dnd_dataset = collect_and_prepare_dialogue_data(data_directory,
32                                                     args.limit_dialogues)
33
34     if args.generate_descriptions:
35         generate_descriptions(dnd_dataset, tokenizer, execution_timestamp,
36                               args.model_server_url)
37
38     if args.description_file:
39         dnd_dataset = add_descriptions_to_dataset(dnd_dataset,
40                                                   args.description_file)
41
42     if args.build_final:
43         t5_tokenizer = AutoTokenizer.from_pretrained("google/flan-t5-xl")
```

```

38     final_dataset = build_dataset(dnd_dataset.to_pandas(), t5_tokenizer)
39     save_filepath =
40         f"dndd-max_d{args.limit_dialogues}-{execution_timestamp}.parquet"
41     final_dataset.to_parquet(save_filepath)
42     print(f"Saved to {save_filepath}")
43
44     subset_map = generate_file_paths(execution_timestamp, args.limit_dialogues)
45     save_dataset_subsets(dnd_dataset, args.subsets, subset_map)
46
47 if __name__ == "__main__":
48     random.seed(42)
49     parser = create_arg_parser()
50     args = parser.parse_args()
51
52     if isinstance(args.subsets, str):
53         subsets = [args.subsets]
54         args.subsets = subsets
55
56     if args.generate_descriptions:
57         tokenizer = LlamaTokenizer.from_pretrained(args.llama_base_model)
58         try:
59             status_code = requests.get(args.model_server_url).status_code
60             if status_code != 200:
61                 raise Exception(f"Model server returned status code
62                                 {status_code}")
63         except requests.exceptions.RequestException as ex:
64             print(f"Could not connect to the model server: {ex}")
65             raise ex
66
67     main(args)

```

Листинг Б.1: Файл prepare.py

```

1 import argparse
2
3
4 def create_arg_parser():
5     parser = argparse.ArgumentParser(
6         description="Process subsets of DNDD (Dungeon & Dragons Dialogues)
7         dataset"
8     )
9     parser.add_argument(
10         "--subsets",
11         nargs="+",
12         required=False,
13         default="all",

```

```

13         help="Subsets of the dataset to process (bg1, bg2, id1, pst, all).
           DEFAULT: all",
14     )
15     parser.add_argument(
16         "--generate_descriptions",
17         required=False,
18         default=False,
19         action=argparse.BooleanOptionalAction,
20         help="Generates a description to NPCs using Alpaca-LoRA-13B in format
           Name/Alignment/Description/Flaw/Motivation/Personality", # noqa:
           E501
21     )
22     parser.add_argument(
23         "--description_file",
24         required=False,
25         default=False,
26         help="Adds NPCs' description to dataset in format
           Name/Alignment/Description/Flaw/Motivation/Personality",
27     )
28     parser.add_argument(
29         "--limit_dialogues",
30         required=False,
31         default=None,
32         type=int,
33         help="Limits the number of dialogues that NPC can have.",
34     )
35     parser.add_argument(
36         "--llama_base_model",
37         default="decapoda-research/llama-13b-hf",
38         help="The name of the base model to use.",
39     )
40     parser.add_argument(
41         "--model_server_url",
42         default="http://127.0.0.1:7860",
43         help="The URL of the model server.",
44     )
45     parser.add_argument(
46         "--generate_descriptions", action="store_true", help="Whether to
           generate descriptions."
47     )
48     parser.add_argument(
49         "--build-final",
50         action="store_false",
51         help="Builds the final training dataset.",
52     )
53
54     return parser

```

Листинг Б.2: Файл arg_parser.py

```

1 import requests
2 from typing import Dict, Tuple, List
3 from tqdm import tqdm
4 from datasets import Dataset
5 from transformers import LlamaTokenizer, PreTrainedTokenizer
6 import pandas as pd
7
8
9 from utils import (
10     format_dialogue_history,
11     format_prompt,
12     tokenize_check_overflow,
13     extract_text,
14     default_text,
15 )
16
17
18 def generate_descriptions(
19     dnd_dataset: Dataset,
20     tokenizer: LlamaTokenizer,
21     execution_timestamp: str,
22     model_server_url: str,
23 ) -> None:
24     """
25     Generate descriptions for NPCs based on their dialogues in the given
26     dataset.
27
28     Parameters
29     -----
30     dnd_dataset : 'Dataset'
31         The dataset containing NPC dialogues.
32     tokenizer : 'LlamaTokenizer'
33         The tokenizer for text encoding.
34     execution_timestamp : 'str'
35         The timestamp for execution.
36     model_server_url : 'str'
37         The URL of the model server for generating descriptions.
38     """
39     grouped_by_filename: pd.DataFrame = (
40         dnd_dataset.to_pandas().groupby("filename").agg({"npc_turns": list,
41             "player_turns": list}))
42
43     # Instruction text for the model
44     instruction_text = "Create the personality of a single NPC in DnD style,
45         based on the provided example dialogue in JSON format. Answer in format

```

```

        Name/Alignment/Description/Flaw/Motivation/Personality in a list format
        written in third person." # noqa: E501
45 header_text = "NPC-turns": '
46
47 instruction_len = len(tokenizer.tokenize(instruction_text))
48 header_len = len(tokenizer.tokenize(header_text))
49
50 for filename, dialogues in tqdm(grouped_by_filename["npc_turns"].items(),
    "Generating data"):
51     dialogue_turns = process_dialogues(dialogues, tokenizer,
        instruction_len, header_len)
52
53     input_text = header_text + str(dialogue_turns)
54     response = make_prediction(
55         model_server_url,
56         instruction_text=instruction_text,
57         input_text=input_text,
58     )
59
60     generated_description = response["data"][0]
61     # Replacing newline characters with "p" to avoid corrupting CSV file
62     generated_description = generated_description.replace("\n", "p")
63     description_filename = f"descriptions-{execution_timestamp}.csv"
64     with open(description_filename, mode="a", encoding="utf-8") as file:
65         file.write(f"{filename},{generated_description}\n")
66
67
68 def make_prediction(
69     model_server_url: str,
70     instruction_text: str,
71     input_text: str,
72     temperature: float = 0.1,
73     top_p: float = 0.2,
74     top_k: int = 100,
75     num_beams: int = 1,
76     max_new_tokens: int = 512,
77     streaming_opt: bool = False,
78 ) -> dict:
79     """
80     Make a prediction using the API endpoint.
81
82     Parameters
83     -----
84     model_server_url : 'str'
85         The URL of the model server.
86     instruction_text : 'str'
87         Instruction text to LLM.
88     input_text : 'str'

```

```

89         Input text containing example NPC dialogues.
90     temperature : 'float', default=0.1
91         Sampling temperature.
92     top_p : 'float', default=0.2
93         Top p sampling value.
94     top_k : 'int', default=100
95         Top k sampling value.
96     num_beams : 'int', default=1
97         Number of beams for beam search.
98     max_new_tokens : 'int', default=512
99         Maximum number of new tokens for the output.
100    streaming_opt : 'bool', default=False
101        Option for streaming LLM output.
102
103    Returns
104    -----
105    dict
106        The prediction response.
107    """
108
109    params = [
110        instruction_text,
111        input_text,
112        temperature,
113        top_p,
114        top_k,
115        num_beams,
116        max_new_tokens,
117        streaming_opt,
118    ]
119
120    response = requests.post(
121        f"{model_server_url}/run/predict",
122        json={"data": params},
123    ).json()
124
125    return response
126
127
128    def build_dataset(dndd_df: pd.DataFrame, tokenizer: PreTrainedTokenizer) ->
    Dataset:
129        """
130        Builds a training dataset from the provided DNDD dataset.
131
132        Parameters
133        -----
134        dndd_df : 'pd.DataFrame'
135            The DNDD dataset DataFrame.

```



```

136     tokenizer : 'PreTrainedTokenizer'
137         The tokenizer used for tokenization.
138
139     Returns
140     -----
141     dict
142         A dictionary containing the 'source' and 'target' lists representing
143         the training dataset.
144     """
145
146     dataset_dict = {"source": [], "target": []}
147
148     for npc_data in tqdm(dnidd_df.itertuples(), "Processing dataset"):
149         dialogue_history = []
150         query = "START DIALOGUE"
151
152         total_turns = len(npc_data.player_turns) + len(npc_data.npc_turns)
153         for turn_index in range(total_turns):
154             if turn_index % 2 == 0:
155                 target = npc_data.npc_turns[turn_index // 2]
156
157                 formatted_history = (
158                     format_dialogue_history(dialogue_history, npc_data.game)
159                     if dialogue_history
160                     else "EMPTY"
161                 )
162                 npc_prompt = format_prompt(npc_data, formatted_history, query)
163                 formatted_history, dialogue_history = tokenize_check_overflow(
164                     tokenizer,
165                     npc_prompt,
166                     formatted_history,
167                     dialogue_history,
168                     npc_data.game,
169                 )
170
171                 if npc_data.game == "pst":
172                     if query != "START DIALOGUE":
173                         query = extract_text(query, default_text("Player"))
174                         target = extract_text(target, default_text("NPC"))
175
176                 npc_prompt = format_prompt(npc_data, formatted_history, query)
177                 dataset_dict["source"].append(npc_prompt)
178                 dataset_dict["target"].append(target)
179             else:
180                 dialogue_history.append(query)
181                 dialogue_history.append(target)
182                 query = npc_data.player_turns[turn_index // 2]

```

```

183     return Dataset.from_dict(dataset_dict)
184
185
186 def process_dialogues(
187     dialogues: List[List[str]],
188     tokenizer: LlamaTokenizer,
189     instruction_len: int,
190     header_len: int,
191 ) -> List[str]:
192     """
193     Process dialogues by removing duplicates and ensuring the length of
194     tokenized text is within limit.
195
196     Parameters
197     -----
198     dialogues : 'List[List[str]]'
199         List of dialogues, each dialogue is a list of turns.
200     tokenizer : 'LlamaTokenizer'
201         Tokenizer to be used.
202     instruction_len : 'int'
203         Length of instruction text.
204     header_len : 'int'
205         Length of header text.
206
207     Returns
208     -----
209     'List[str]'
210         Processed dialogue turns.
211     """
212
213     dialogue_turns: List[str] = []
214     for dialogue in dialogues:
215         for turn in dialogue:
216             dialogue_turns.append(turn)
217             # Removing any duplicate turns from the dialogue
218             dialogue_turns = list(set(dialogue_turns))
219
220             # Checking if the length of the tokenized dialogue_turns and
221             # instruction texts are within the limit
222             prompt_len = len(tokenizer.tokenize(str(dialogue_turns))) +
223                 instruction_len + header_len
224             if prompt_len > 500:
225                 print(prompt_len)
226                 dialogue_turns = dialogue_turns[:-1]
227                 break
228             else:
229                 continue
230         break

```

```

228     return dialogue_turns
229
230
231 def generate_file_paths(
232     execution_timestamp: str, limit_dialogues: int
233 ) -> Dict[str, Tuple[str, str]]:
234     """
235     Generates file paths for subsets of a dataset based on the execution
236     timestamp and limit of dialogues.
237
238     Parameters
239     -----
240     execution_timestamp : 'str'
241         The timestamp of the execution.
242     limit_dialogues : 'int'
243         The maximum number of dialogues allowed in the subsets.
244
245     Returns
246     -----
247     'dict'
248         A dictionary mapping subset names to their corresponding file paths.
249     """
250     file_prefix = "data/dnidd_subset_"
251     file_suffix = f"_{execution_timestamp}_max-d_{limit_dialogues}.parquet"
252     subset_map = {
253         "all": ("", f"{file_prefix}all{file_suffix}"),
254         "bg1": ("bg1", f"{file_prefix}bg1{file_suffix}"),
255         "bg2": ("bg2", f"{file_prefix}bg2{file_suffix}"),
256         "id1": ("id1", f"{file_prefix}id1{file_suffix}"),
257         "pst": ("pst", f"{file_prefix}pst{file_suffix}"),
258     }
259     return subset_map
260
261
262 def save_dataset_subsets(
263     dnd_dataset: Dataset, subsets: List[str], subset_map: Dict[str, Tuple[str,
264         str]]
265 ) -> None:
266     """
267     Saves subsets of a dataset based on specified subsets and their
268     corresponding paths.
269
270     Parameters
271     -----
272     dnd_dataset : 'Dataset'
273         The dataset to save subsets from.
274     subsets : 'List[str]'

```

```

273         A list of subset names to save.
274 subset_map : 'Dict[str, Tuple[str, str]]'
275         A dictionary mapping subset names to their corresponding paths.
276         """
277
278     for subset in subsets:
279         subset_prefix, subset_file_path = subset_map[subset]
280         subset_dndd = dnd_dataset.filter(lambda example:
281             example["game"].startswith(subset_prefix))
282         subset_dndd.to_parquet(subset_file_path)
283         print(f"Saved to {subset_file_path}")
284
285 def add_descriptions_to_dataset(dnd_dataset: Dataset, description_file: str) ->
    Dataset:
286     """
287     Adds descriptions to a dataset by merging it with a description file.
288
289     Parameters
290     -----
291     dnd_dataset : 'Dataset'
292         The dataset to which descriptions will be added.
293     description_file : 'str'
294         The path to the description file.
295
296     Returns
297     -----
298     'Dataset'
299         The updated dataset with descriptions.
300     """
301
302     dndd_df = dnd_dataset.to_pandas()
303     desc_df = pd.read_csv(description_file, sep="|")
304     dndd_df_merged = pd.merge(dndd_df, desc_df, on="filename")
305     dnd_dataset = Dataset.from_pandas(dndd_df_merged)
306     return dnd_dataset

```

Листинг Б.3: Файл descriptions.py

```

1 import json
2 import os
3 import random
4 from typing import Union
5 from datasets import Dataset
6 from tqdm import tqdm
7
8
9 def identify_game(dir: str) -> Union[str, None]:

```

```

10     """
11     Identify the game based on the directory name.
12
13     Parameters
14     -----
15     dir : 'str'
16         The directory name.
17
18     Returns
19     -----
20     'Union[str, None]'
21         The game identifier.
22     """
23     game_identifiers = ["pst", "idl", "bg1", "bg2"]
24
25     for game in game_identifiers:
26         if game in dir:
27             return game
28
29     return None
30
31
32 def load_dialogues_from_file(path: str, limit: Union[int, None] = None) -> list:
33     """
34     Load dialogues from a file.
35
36     Parameters
37     -----
38     path : 'str'
39         The path of the file.
40     limit : 'Union[int, None]', default=None
41         The maximum number of dialogues to load.
42
43     Returns
44     -----
45     content:
46         The list of dialogues.
47     """
48
49     with open(path) as file:
50         content = json.load(file)
51     if limit:
52         content = random.sample(content, limit)
53     return content
54
55
56 def collect_and_prepare_dialogue_data(data_directory: str, limit: Union[int,
None]) -> Dataset:

```

```

57     """
58     Collects and prepares dialogue data from multiple files within a directory.
59     The collected data is converted into a Hugging Face Dataset object.
60
61     Parameters
62     -----
63     data_directory : 'str'
64         The base directory containing the files to read data from.
65     limit : 'Union[int, None]'
66         The maximum number of dialogues to load, or None for no limit.
67
68     Returns
69     -----
70     'Dataset'
71         A Hugging Face Dataset object containing the collected dialogue data.
72     """
73
74     dialogue_data = []
75     for directory in os.listdir(data_directory):
76         files = os.listdir(os.path.join(data_directory, directory))
77         game = identify_game(directory)
78         for filename in tqdm(files, f"Processing files in {directory}"):
79             dialogues = load_dialogues_from_file(
80                 os.path.join(data_directory, directory, filename), limit
81             )
82             for dialogue in dialogues:
83                 dialogue["filename"] = filename
84                 dialogue["game"] = game
85                 dialogue_data.append(dialogue)
86
87     dnd_dataset = Dataset.from_list(dialogue_data)
88     dnd_dataset = dnd_dataset.rename_columns(
89         {"HeroSpeech": "player_turns", "CharacterSpeech": "npc_turns"}
90     )
91     dnd_dataset = dnd_dataset.remove_columns("Id")
92     return dnd_dataset

```

Листинг Б.4: Файл dialogue_data.py

```

1  import re
2  from typing import List, Tuple
3  import pandas as pd
4  from transformers import PreTrainedTokenizer
5
6
7  def format_dialogue_history(dialogue_history: List[str], game_type: str) -> str:
8      """
9      Formats the dialogue history into a readable format.

```

```

10
11     Parameters
12     -----
13     dialogue_history : 'List[str]'
14         A list containing the dialogue history. Each item is a dialogue string.
15     game_type : 'str'
16         A string representing the type of the game. If 'pst', special
            formatting is applied.
17
18     Returns
19     -----
20     'str'
21         The formatted dialogue history. Each turn is on a new line with the
            format 'Speaker: Dialogue turn'.
22     """
23     formatted_history = ""
24     for turn_index in range(len(dialogue_history)):
25         speaker = "Player" if turn_index % 2 == 0 else "NPC"
26         dialogue_turn = dialogue_history[turn_index]
27         if game_type == "pst" and dialogue_turn != "START DIALOGUE":
28             dialogue_turn = dialogue_turn.replace("\\r", "").replace("\\n", "")
29             extracted_text = re.findall('"(^[^"]*)"', dialogue_turn)
30             extracted_text = " ".join(extracted_text) if extracted_text else
                default_text(speaker)
31             formatted_history += f"{speaker}: {extracted_text}\n"
32         else:
33             formatted_history += f"{speaker}: {dialogue_turn}\n"
34     return formatted_history
35
36
37 def format_prompt(npc_data: pd.Series, current_history: str, query: str) -> str:
38     npc_prompt = f""""Below is the definition of in-game NPC.
39     NPC Name: {npc_data['name']}
40     Alignment: {npc_data['alignment']}
41     Description: {npc_data['description']}
42     Personality traits: {npc_data['personality']}
43     Flaws: {npc_data['flaw']}
44     Motivation: {npc_data['motivation']}
45
46     Dialogue history:
47     {current_history}
48     Player query: {query}
49
50     Respond to player's query based on defined NPC: """"
51     return npc_prompt
52
53
54 def tokenize_check_overflow(

```

```

55     tokenizer: PreTrainedTokenizer,
56     npc_prompt: str,
57     current_history: str,
58     dialogue_history: List[str],
59     game: str,
60 ) -> Tuple[str, List[str]]:
61     """
62     Truncates the dialogue history to avoid tokenization overflow.
63
64     Parameters
65     -----
66     tokenizer : 'PreTrainedTokenizer'
67         The tokenizer object used for tokenization.
68     npc_prompt : 'str'
69         The NPC prompt or instruction text.
70     current_history : 'str'
71         The current dialogue history.
72     dialogue_history : 'List[str]'
73         The list of previous dialogue turns.
74     game : 'str'
75         The game identifier.
76
77     Returns
78     -----
79     'Tuple[str, List[str]]'
80         A tuple containing the updated current history and dialogue history.
81     """
82
83     prompt_tokens = tokenizer.tokenize(npc_prompt)
84     history_tokens = tokenizer.tokenize(current_history)
85     total_tokens = len(prompt_tokens) + len(history_tokens)
86     while total_tokens > 1024:
87         dialogue_history = dialogue_history[2:]
88         current_history = format_dialogue_history(dialogue_history, game)
89         history_tokens = tokenizer.tokenize(current_history)
90         total_tokens = len(prompt_tokens) + len(history_tokens)
91
92     return current_history, dialogue_history
93
94
95 def extract_text(dialogue, default_text):
96     extracted_text = re.findall('"([^\"]*)"', dialogue)
97     return " ".join(extracted_text) if extracted_text else default_text
98
99
100 def default_text(speaker: str):

```



```
101 return "Ignore." if speaker == "Player" else "That NPC seems to be ignoring  
    you."
```

Листинг Б.5: Файл `utils.py`

ПРИЛОЖЕНИЕ В. ПРИМЕРЫ ДИАЛОГОВ

```
1 Below is the definition of in-game NPC.
2 NPC Name: Saablic Tan
3 Alignment: Neutral
4 Description: A tall, gaunt figure with a long, white beard and a hooded cloak.
5 Personality traits: Overconfidence
6 Flaws: To protect his creations from harm
7 Motivation: Saablic is a proud and confident individual, but he is also a bit
   arrogant and overconfident. He is willing to take risks and is not afraid to
   speak his mind. He is also very protective of his creations and will do
   anything to protect them.
8
9 Dialogue history:
10 Player: START DIALOGUE
11 NPC: Proceed... no further, traveler, as... I would... speak with you.
12 Player: I'm sorry, but I really must be on my way.
13 NPC: Know this... traveler. Without... my knowledge... your continued
   existence... within this world... will be limited.
14 Player: All right, I'm listening.
15 NPC: My name... is... was Saablic... Tan. Of my... mistakes... one grew
   tired... used my precious... magics to make... that which... I am. Having...
   much confusion as... days move on. Must... mm... tell you I... am frightened.
16
17 Player query: Why are you frightened?
18
19 Respond to player's query based on defined NPC:
20 ANSWER: Orog's that... roam these passages... my passages... are my... creation.
   They... have much contempt... for me. I cannot... leave. My form... my
   home... punishment for my... ambition. I... derived pleasure... for altering
   against... their will. I have... much regret.
```

Листинг В.1: Пример диалога 1

```
1 Below is the definition of in-game NPC.
2 NPC Name: Alaina
3 Alignment: Neutral
4 Description: A ghoulish captain with a cool and calculating demeanor.
5 Personality traits: A tendency to be overly cautious and suspicious of
   outsiders.
6 Flaws: To protect the Drowned Nations from intruders.
7 Motivation: Alaina is a cautious and calculating ghoulish captain. She is wary of
   outsiders and will not hesitate to warn them away from the Drowned Nations.
   She is protective of her territory and will not hesitate to defend it from
   intruders.
```

```

8
9 Dialogue history:
10 Player: START DIALOGUE
11 NPC: That NPC seems to be ignoring you.
12
13 Player query: Who are you? What are you doing here?
14
15 Respond to player's query based on defined NPC:
16 ANSWER: I? I am... Alaina. These are my ghouls. We are scouts and hunters.
    We seek more territory... we seek more *food.* And now that I have
    answered your questions... what are YOU doing here?

```

Листинг В.2: Пример диалога 2

```

1 Below is the definition of in-game NPC.
2 NPC Name: Eidara the Shaman
3 Alignment: Neutral Good
4 Description: A tall and rugged male with short silver hair and a bushy brown
    beard, wearing tattered furs and leathers decorated with various trinkets
    and feathers. He carries himself with confidence, but also shows compassion
    towards others. His piercing blue eyes seem to hold secrets about the world
    beyond mortal understanding.
5 Personality traits: Overconfident - Believing he has seen everything life has
    to offer, Eidara sometimes takes unnecessary risks which put him and his
    allies into harmful situations.
6 Flaws: To protect his people, the Northern tribes and way of life against any
    threat - including other humanoids or creatures.
7 Motivation: An experienced shaman, wise beyond his years and quick on his feet.
    Eidara cares deeply for his family, friends and fellow tribesmates, often
    putting himself at risk to protect them. This fierce loyalty extends to
    those he deems worthy of trust.
8
9 Dialogue history:
10 Player: START DIALOGUE
11 NPC: Hail, friends. How may I help you?
12
13 Player query: What can you tell me of Icasaracht?
14
15 Respond to player's query based on defined NPC:
16 ANSWER: Icasaracht, a ghastly creature that inhabits the northern part of the
    Great Land, is the worst threat to the people of this land. I have watched
    its behavior from my window since its arrival, and I know its power is not
    limited to my realm.

```

Листинг В.3: Пример диалога 3

```

1 PROMPT: Below is the definition of in-game NPC.

```

2 NPC Name: Joril Frostbeard
3 Alignment: Lawful Evil
4 Description: A tall, muscular man with a long white beard and a fierce gaze.
5 Personality traits: Overconfidence
6 Flaws: To prove himself worthy of his father's legacy as Jarl.
7 Motivation: Joril is a proud and arrogant man, who believes himself to be the
strongest and most capable of all. He is quick to anger and will not
hesitate to use violence to get what he wants. He is also very stubborn and
will not back down from a challenge.
8
9 Dialogue history:
10 Player: START DIALOGUE
11 NPC: You again. What do you want?
12 Player: I seek the one in charge.
13 NPC: Well, you've found him. I'm Joril Frostbeard, and I rule here. What do you
want of me?
14
15 Player query: What gives you the right to rule this glacier, Joril? Certainly
the slaves would contest your rule.
16
17 Respond to player's query based on defined NPC:
18 ANSWER: The slaves have no choice, anyway. I am the chosen one, and the one who
remains loyal to my father is irrelevant. Under my rule, the slaves would
never rebel. If the war broke out between my father and me

Листинг В.4: Пример диалога 4