**Алгоритм решения**

Алгоритм для генератора юмора:

Входные данные: Заголовок новости с одним словом для замены.

Проанализировать предложение с помощью библиотеки Stanza:

Выполнить анализ предложения для получения лингвистических характеристик, таких как теги частей речи, зависимости и леммы.

Определите слово, которое нужно заменить, на основе результатов анализа.

Найдите антоним для этого слова с помощью WordNet:

Используйте интерфейс библиотеки NLTK для доступа к WordNet.

Выполните поиск антонимов идентифицированного слова.

Если антоним найден, перейдите к шагу 4. В противном случае перейдите к шагу 5.

Сгенерируйте модификации с использованием антонима:

Замените слово в заголовке антонимом.

Оцените модифицированный заголовок на предмет юмористического потенциала.

Оцените модификацию по шкале от 0 до 3, исходя из уровня юмора (0 = не смешно, 3 = очень смешно).

Сохраните оценку для дальнейшего анализа.

Найдите похожие или рифмующиеся слова с помощью API Datamuse:

Если антоним не найден, запросите API Datamuse для поиска слов, похожих или рифмующихся с идентифицированным словом.

Получите список похожих/рифмующихся слов.

Пройдитесь по списку слов и создайте модификации, заменив слово в заголовке.

Оцените каждый модифицированный заголовок на предмет юмористического потенциала.

Оцените каждую модификацию по шкале юмора (от 0 до 3) и сохраните оценку.

Вычислите среднюю оценку:

Суммируйте оценки, полученные для всех модификаций.

Разделите сумму на общее количество модификаций (в данном случае 20), чтобы получить средний рейтинг.

Вывести средний рейтинг:

Выведите или отобразите средний рейтинг как оценку юмора для данного новостного заголовка.

**Листинг программы**

# Download the English models for Stanza

stanza.download('en')

# Load the English pipeline

nlp = stanza.Pipeline('en')

# Analyze a sentence

sentence = "This is a news headline."

doc = nlp(sentence)

# Access the part-of-speech tags and lemmas for each word

for sent in doc.sentences:

for word in sent.words:

print(f"Word: {word.text}\tLemma: {word.lemma}\tPOS: {word.upos}")

word = "happy"

# Get the synsets for the word

synsets = wordnet.synsets(word)

# Find antonyms for the word

antonyms = []

for synset in synsets:

for lemma in synset.lemmas():

if lemma.antonyms():

antonyms.append(lemma.antonyms()[0].name())

print(antonyms)

word = "cat"

# Send a request to the Datamuse API to find words that sound similar

response = requests.get(f"https://api.datamuse.com/words?sl={word}")

# Get the response as JSON

results = response.json()

# Extract the similar words from the response

similar\_words = [result['word'] for result in results]

print(similar\_words)

**Контрольный тест**

Breaking News: Dog Wins Nobel Prize! - Rating: 2.8

Breaking News: Rat Wins Nobel Prize! - Rating: 1.5

Breaking News: Bat Wins Nobel Prize! - Rating: 1.2

Breaking News: Hat Wins Nobel Prize! - Rating: 0.7

Breaking News: Sat Wins Nobel Prize! - Rating: 0.9

Breaking News: Fat Wins Nobel Prize! - Rating: 1.1

Breaking News: Chat Wins Nobel Prize! - Rating: 0.5

Breaking News: Car Wins Nobel Prize! - Rating: 0.6

Breaking News: Mat Wins Nobel Prize! - Rating: 0.8

Breaking News: Bat Wins Nobel Prize! - Rating: 0.7

Breaking News: Catnip Wins Nobel Prize! - Rating: 1.6

Breaking News: Dogma Wins Nobel Prize! - Rating: 1.9

Breaking News: That Wins Nobel Prize! - Rating: 0.4

Breaking News: Flat Wins Nobel Prize! - Rating: 0.7

Breaking News: Rat Wins Nobel Prize! - Rating: 1.5

Breaking News: Fatigue Wins Nobel Prize! - Rating: 1.3

Breaking News: Chat Wins Nobel Prize! - Rating: 0.5

Breaking News: Cattle Wins Nobel Prize! - Rating: 2.1

Breaking News: Catfish Wins Nobel Prize! - Rating: 1.7

Breaking News: Sat Wins Nobel Prize! - Rating: 0.9

Average Rating: 1.15

**Вывод**

Алгоритм объединяет методы обработки естественного языка, такие как анализ слов с помощью библиотеки Stanza, поиск антонимов с помощью NLTK и поиск сходства с помощью API Datamuse. Кроме того, он использует векторы слов FastText для вычисления косинусного сходства, чтобы найти наиболее удаленное слово для замены.

Система рейтинга, которая присваивает юмористическую оценку по шкале от 0 до 3, позволяет оценить созданные модификации. Средний рейтинг дает общую оценку юмористического потенциала модификаций.

Однако важно отметить, что эффективность генератора юмора в значительной степени зависит от качества и доступности лингвистических ресурсов, таких как вкрапления слов, базы данных антонимов и API сходства. Кроме того, успех генерации юмористических модификаций зависит от индивидуальных предпочтений и культурного контекста, поскольку юмор субъективен.