## **Отчет по задаче генерации описания к видео**

### **1. Название задачи и участники**

**Задача:** Разработка модели генерации текстового описания к короткому видео

**Участники проекта:** Хачатрян Саргис, Ткаченко Арэс

### **2. Краткое описание и постановка задачи**

Цель проекта — создать модель, способную генерировать осмысленные текстовые описания (caption) к коротким видеороликам.  
 На вход подаётся видеофайл, из которого извлекаются кадры. Модель должна выдать на выходе текст, описывающий происходящее.

### **3. Датасет и его проблемы**

Датасет состоит из 603 коротких видеофайлов (.mp4) и соответствующих текстовых описаний (caption). Каждое видео описано предложением на английском языке, отражающем его содержание.

* **Много длинных описаний:** из-за чего модель будет обучаться дольше
* **Малое-кол-во данных:** модель может не научится поставленной задаче
* **Данные - видео:** т.е. их нужно как-то разбивать на кадры

### **4. Предобработка данных**

#### **Текст:**

* Использована токенизация nltk.word\_tokenize и лемматизация WordNetLemmatizer.
* Добавлены специальные токены: <PAD>, <SOS>, <EOS>, <UNK>.
* Осуществлена последовательная подготовка:
  + Преобразование текста в список токенов
  + Построение словаря с Tokenizer из Keras
  + Преобразование токенов в ID и паддинг до фиксированной длины

#### **Видео:**

* Из каждого видео извлекались **N = 16** кадров, равномерно распределённых по длине ролика.
* Использован cv2.VideoCapture для извлечения кадров.
* Для каждого кадра применялись преобразования
* Изображения преобразовывались в тензоры и объединялись в batch.

### **5. Описание моделей**

Разработаны модели OptimusPrime и Bumblebee:

### **OptimusPrime (Transformer-based)**

**Архитектура:**

* **Видео энкодер:** r3d\_18 (3D ResNet-18), обученный на видеоданных. Он извлекает пространственно-временные признаки из видео.
* **Проекция:** Линейный слой, проецирующий выход ResNet в пространство размерности d\_model.
* **Текст энкодер:** Transformer Encoder, обрабатывающий текст и встроенную видеоинформацию.
* **Декодер:** Transformer Decoder, генерирующий токены на основе скрытых состояний.
* **Генерация:** генерация последовательности с использованием temperature sampling.

**Особенности:**

* Использует **всю видеопоследовательность как 3D-объект**.
* Позволяет учесть **контекст всей последовательности** через self-attention.
* Гибкость архитектуры и масштабируемость.

### **Bumblebee (LSTM-based)**

**Архитектура:**

* **Видео энкодер:** ResNet50, применяемый **по кадру**, с последующим усреднением по времени.
* **Проекция:** Преобразует CNN-вектор в d\_model вектор.
* **Текст энкодер:** LSTM, который обрабатывает объединение видеофичи и токенов исходного текста.
* **Декодер:** LSTM, генерирующий последовательность на основе encoder hidden state.
* **Генерация:** Пошаговая, LSTM-декодером.

**Особенности:**

* Простая и проверенная архитектура для последовательных данных.
* **Быстрее и легче**, чем Transformer.

### **6. Процесс обучения и метрики**

* **Фреймворк:** PyTorch
* **Метод:** Teacher Forcing с перекрытием входа и цели
* **Функция потерь:** CrossEntropyLoss
* **Оптимизатор:** Adam
* **Метрика:** BLEU Score (сравнение генерации с эталонными описаниями)

### **7. Сравнение моделей**

|  |  |  |
| --- | --- | --- |
| **Модель** | **Параметры** | **BLEU** |
| Bumblebee | d\_model=512, LSTM(2 сл.) | 0.009 |
| OptimusPrime | d\_model=512, Transformer(6 сл.) | 0.007 |

### **8. Выводы и улучшения**

* **Низкие значения BLEU**:  
   Полученные значения BLEU — 0.009 для модели OptimusPrime и 0.007 для Bumblebee — указывают на то, что качество сгенерированных описаний видео на текущем этапе остаётся очень низким. Возможно причина кроется в малом кол-ве данных и небольшом числе эпох обучения. Т.е. стоит обучать модели дольше.
* **Оптимус немного лучше**:  
   Хотя обе модели показывают слабые результаты, OptimusPrime немного опережает Bumblebee, что может говорить о более удачной архитектуре или более качественном использовании признаков. Это говорит о том, что возможно архитектура трансформер будет лучше для решения этой задачи.
* **Потенциальные улучшения**:
  + Аугментация и сэмплирование данных
  + Выбор большего числа кадров для анализа/Выбор ключевых кадров