УДК 004.8, 004.4, 004.9, 778.582

АВТОМАТИЗАЦИЯ СОРТИРОВКИ МАТЕРИАЛОВ ПО ТЕКСТУ СЦЕНАРИЯ ДЛЯ ВИДЕОМОНТАЖА

А. Д. Неманов[1] [0009-0004-1840-2347], И. С. Шахова[2] [0000-0003-1591-5767]

[1, 2] *Институт информационных технологий и интеллектуальных систем, Казанский федеральный университет, г. Казань*

[1]andrewoch@yandex.ru, [2]is@it.kfu.ru

Аннотация

Процесс видеомонтажа включает множество трудоемких операций по сортировке и подготовке материалов, что требует значительных временных затрат. В статье описана разработка программного решения для автоматизации этих процессов с использованием технологии машинного обучения. Основное внимание уделено созданию системы, способной классифицировать и сортировать медиафайлы по тексту сценария, тем самым повышая эффективность подготовки материалов к монтажу. Система включает модули распознавания речи, классификации аудио и видео, а также алгоритмы определения соответствия сценарию. Тестирование показало, что предложенная система правильно классифицирует медиафайлы в большинстве случаев, что позволяет существенно сократить время на черновой монтаж.

Ключевые слова*: видеомонтаж, автоматизация, машинное обучение, распознавание речи, классификация аудио, классификация видео, coreml, параллельные вычисления, сценарий, soundex, tf-idf, косинусное сходство, обработка естественного языка*

ВВЕДЕНИЕ

Монтаж является неотъемлемой частью производства видео и состоит из таких процессов, как выбор и соединение подходящих видеоматериалов, коррекция цвета, добавление специальных эффектов, аудиодорожек и музыки.

В процессе съемок у авторов накапливается значительное количество материала. При этом, как правило, для улучшения качества звука голоса людей запи-

сываются на отдельные микрофоны, из-за чего в результате монтажеру предоставляются как видеофайлы, так и звуковые файлы. Среди них содержатся неудачные дубли, тестовые записи и записи с плохим качеством звука.

На раннем этапе обработки материалов видеомонтажеру необходимо провести так называемый «черновой» монтаж, то есть собрать в монтажной программе примерный макет видеопроекта и проверить, что нужно переснять, каких кадров не хватает или какие кадры не «клеятся» на монтаже [1]. Этот процесс подразумевает следующие действия: отсечь бракованный материал, отобрать все удачные дубли, найти звуковые файлы, соответствующие видеоряду, перепроверить материал на наличие технического брака и сделать заключение о готовности приступить к монтажу. Для этого необходимо вручную прослушивать и сортировать файлы, что сопровождается значительными трудозатратами.

Хотя создатели кино редко показывают статистику по итогам своих съемочных процессов, известно, что обычный съемочный день длится от 8 до 12 часов. При этом длина отснятого удачного материала, который потом попадает в итоговый вариант, составляет 2–3 минуты в кино, 6–7 минут в простых разговорных сериалах. Количество дублей на каждый кадр может разниться от 2–3 до нескольких десятков [2, 3]. При этом нужно учитывать, что запись ведется на несколько устройств и что специалисту необходимо пересмотреть материалы в среднем по 2 раза, чтобы убедиться в качестве отобранных материалов. Так, для минутного ролика черновой монтаж может длиться до 20 минут, а для полнометражного фильма может потребоваться 240 часов.

Чтобы упростить себе задачу, съемочная команда переименовывает файлы по заранее оговоренному шаблону, отмечая удачные дубли, иногда даже находясь непосредственно на месте съемок. Однако эти действия занимают больше времени, что создает риск отклониться от графика [4].

Решением обозначенной проблемы является инструмент, который позволит автоматизировать сортировку видео и аудио, а затем, следуя предоставленному сценарию, определить самые удачные дубли с точки зрения соответствия сценарию.

Для проведения обзора альтернативных решений были выделены следующие критерии: возможность сортировки по сценарию по тексту речи, звуку и описанию картинки, проверка соответствия сценарию, автоматизация сортировки.

Одним из альтернативных программных решений является программа для монтажа видео Adobe Premiere Pro[1]. Она не имеет функционала автоматической сортировки файлов, но предоставляет возможность вручную организовывать иерархию файлов в проекте.

Еще одним похожим решением является программа Davinci Resolve[2]. Она также не имеет возможности автоматизированной сортировки по сценарию, однако предоставляет модуль распознавания речи в аудиофайлах, что позволяет пользователю искать нужные файлы по тексту речи.

Самым близким к необходимому решению является AVID Media Composer. Это программа, которая специализируется на подготовке видеофрагментов к монтажу, сортировке, а также черновому монтажу [5]. В ней имеется функционал добавления сценария, расшифровки речи и поиска по тексту речи. Также файлы можно автоматически отсортировать по сценарию. Однако это работает только для фраз, по сценам экспозиции файлы сортировать нельзя.

СТАНДАРТ ОФОРМЛЕНИЯ СЦЕНАРИЯ

В киноиндустрии приняты определенные стандарты, по которым оформляется сценарий. Существует несколько стандартов с разными вариантами оформления, но их объединяет одно: сценарии состоят из логических блоков, позволяющих упростить их понимание и контролировать процесс съемок, отмечая отснятые части [6].

Среди таких блоков можно выделить два:

1) диалоги;
2) описание действия (далее сцены-экспозиции).

Блоки диалогов состоят из последовательности фраз. Фразы, в свою очередь, состоят из имени персонажа и текста речи этого персонажа. Так как в некоторых случаях съемка диалога происходит в несколько дублей или даже по одной фразе на дубль, было принято решение впоследствии работать именно с отдельной фразой.

Сцены-экспозиции состоят из описания какого-либо действия без речи персонажей. Предполагается, что текстовый файл сценария соответствует стандарту

[1] https://www.adobe.com/products/premiere.html

[2] https://www.blackmagicdesign.com/products/davinciresolve

киноиндустрии, чтобы имелась возможность программным путём разбить сценарий на логические блоки (рис. 1).

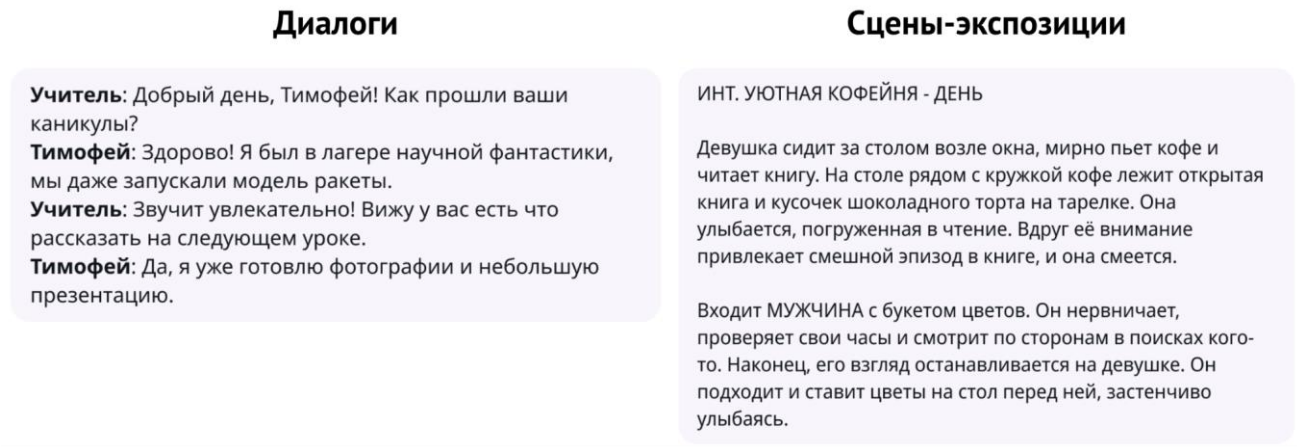


Рис. 1. Пример логических блоков

ПРОГРАММНЫЕ МЕХАНИЗМЫ СОРТИРОВКИ ДИАЛОГОВ ПО СЦЕНАРИЮ

Определить фразу, произнесенную персонажем, можно, сравнивая текст расшифровки речи в файле и текст фразы в сценарии. Однако в действительности существует ряд проблем, для решения которых требуется усложнить этот механизм:

1) Расшифровка речи сопровождается погрешностью в распознавании – могут распознаваться другие слова, близкие по звучанию.
2) В файле помимо самой фразы могут записываться и служебные разговоры. У профессионалов это может быть фраза «Камера, мотор!», а у начинающих могут быть записаны случайные диалоги за кадром, которые по длине не отличаются от самой фразы текста. Алгоритм определения фразы должен обладать возможностью игнорировать эти части при сравнении.
3) В одном файле может быть записано сразу несколько дублей одной сцены.

Первую проблему можно решить, преобразуя слова в расшифровке речи и тексте сценария в фонетические коды и сравнивая их. Это позволит опираться на произношение слов, а не на то, как они пишутся.

Вторая и третья проблемы решаются сегментацией аудиофайла во время расшифровки речи и определением сегментов, подходящих или неподходящих по сценарию. Это также может помочь при черновом монтаже, ведь если автоматически отметить часть файла, подходящую по сценарию, то можно обращать внимание только на эту часть, не затрачивая время на остальные.

Таким образом, порядок работы механизма сортировки диалога по сценарию выглядит следующим образом: у медиафайла производится расшифровка речи с временными отметками для каждого сегмента. Затем у этой расшифровки все слова преобразуются в фонетические коды. Такое же преобразование происходит и у текста каждой фразы в сценарии. После этого алгоритм сравнивает фонетические коды файла с кодами каждой фразы, определяя наиболее подходящую фразу и указывая, какие сегменты в файле подходят к этой фразе, а какие нет (рис. 2).

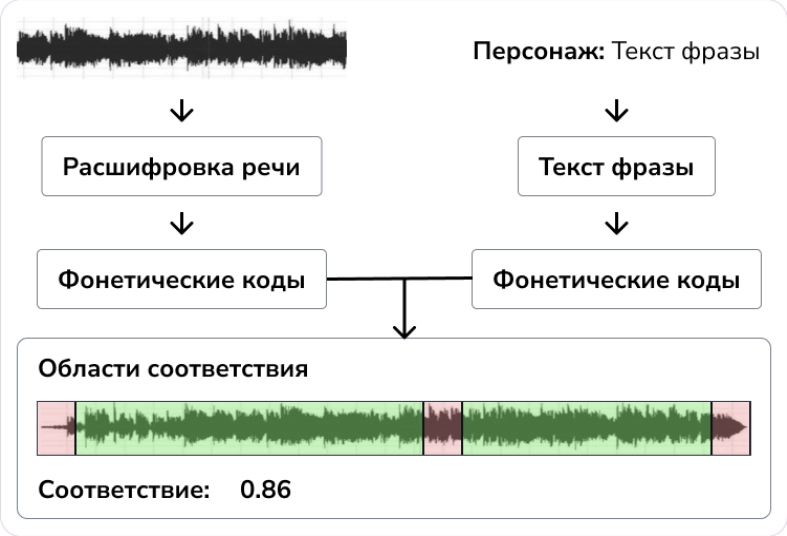


Рис. 2. Схема работы механизма сортировки диалогов по сценарию

При реализации расшифровки речи необходимо было балансировать между точностью и скоростью, так как цель предложенного инструмента — сэкономить специалисту время, которое он тратит на сортировку видео- и аудиофайлов.

Для решения данной задачи была выбрана модель расшифровки речи Whisper от OpenAI [7, 8]. За счёт архитектуры, учитывающей контекст предложения, выбранная модель показывает точность, достаточную для решения поставленной задачи даже в случае сниженного качества обрабатываемых аудиозаписей. Кроме того, она поддерживает несколько языков, что обеспечивает перспективы адаптации программного решения под эти языки. Была выбрана модель small, третья по величине.

При выборе подходящей библиотеки, использующей Whisper для расшифровки и имеющей поддержку сегментации, были рассмотрены два варианта: SwiftWhisper [9] и WhisperKit [10]. Главным преимуществом WhisperKit является то, что эта библиотека позволяет сегментировать аудиофайл по одному слову в сегменте. Благодаря этому можно избежать ситуаций, когда в один сегмент могли попасть как часть речи по сценарию, так и часть посторонних разговоров. При этом временные границы сегментов с использованием WhisperKit определяются без погрешностей точнее по сравнению со SwiftWhisper (рис. 3).

Рис. 3. Сравнение сегментации библиотек расшифровки речи

За счет оптимизации под ARM-процессоры распознавание речи с WhisperKit работает быстрее, чем реализация SwiftWhisper на C++, и, кроме того, имеется возможность распознавать несколько файлов одновременно.

Ещё одним преимуществом WhisperKit является то, что можно распознавать несколько файлов одновременно. Был разработан модуль расшифровки, который имеет собственную очередь задач DispatchQueue [11], в которые можно добавлять файлы на расшифровку. Очередь контролируется с помощью DispatchSemaphore, который пропускает на расшифровку файлы по количеству

ядер процессора устройства. Таким образом получается избежать переполнения оперативной памяти устройства, даже если в очереди находится несколько сотен файлов, ведь одновременно распознаваться будет только определенное их количество. К тому же этот модуль не блокирует работу остального приложения и позволяет добавлять файлы в очередь, даже если в ней уже находятся файлы. При добавлении в очередь файлы сортируются по длительности, чтобы более короткие распознавались первыми и быстрее покидали очередь, освободив память. Для обработки ошибок и лучшего контроля большого количества асинхронных процессов был использован Combine [12].

Перед сравнением было решено конвертировать слова в фонетические коды Soundex [13]. Был реализован модуль, который преобразует строку в массив фонетических кодов с использованием инструмента RussianSoundex из библиотеки Fonetika [14]. Для оптимизации уже преобразованные слова добавлялись в словарь, чтобы повторно использовать полученный код. Также при загрузке сценария все его блоки диалогов сразу обрабатываются, и заготовленные коды для каждой сцены сохраняются для дальнейшей работы.

Был разработан алгоритм классификации фразы, который работает следующим образом: при сравнении с каждой фразой в массиве сегментов расшифровки сразу подсчитывается, какие сегменты подряд подходят по сценарию, а какие нет. Таким образом формируется чередующаяся последовательность из наборов подходящих и неподходящих сегментов. В процессе составляется список из начальных и конечных индексов каждого набора, а также длины этого набора.

После составления списков наборов сегментов для каждой фразы выбирается список с самым длинным подходящим набором, и фраза, к которой был создан этот набор, выбирается как подходящая для файла. После этого по полученному списку индексов происходит склеивание исходных сегментов, и новый набор сегментов заменяет исходный.

С таким подходом удалось реализовать алгоритм, который для обработки одного файла имеет временную сложность O(N×P×M), где:

- N — количество слов в расшифровке файла,
- P — количество фраз в сценарии,
- M — средняя длина фразы в сценарии.

Было проведено тестирование алгоритма, в результате которого выявилась следующая проблема: если в середине расшифровки речь распознается плохо и даже фонетические коды у слов отличаются, то в результате получается два подходящих сегмента, разделенные неподходящим сегментом из проблемных слов. Однако, поскольку общее сходство по сценарию считается как сходство самого длинного подходящего сегмента, из-за разделенных сегментов алгоритм выдаёт значение, которое отличается от фактического минимум вполовину, что может привести к неправильной классификации.

Поэтому алгоритм был дополнен механизмом подсчёта сквозной длины совпадения с фразой: при нахождении подходящего сегмента определяется, на каком слове фразы этот сегмент закончился. Затем при нахождении следующего подходящего сегмента определяется слово фразы, с которого новый сегмент начался. Вычисляется разрыв между конечным словом фразы из прошлого сегмента и начальным словом фразы из нового сегмента. Если разница между этим разрывом и длиной неподходящего сегмента между подходящими не превышает два слова, то считается, что новый подходящий сегмент продолжает фразу из первого сегмента, пропустив столько слов фразы, сколько слов в неподходящем сегменте (рис. 4).

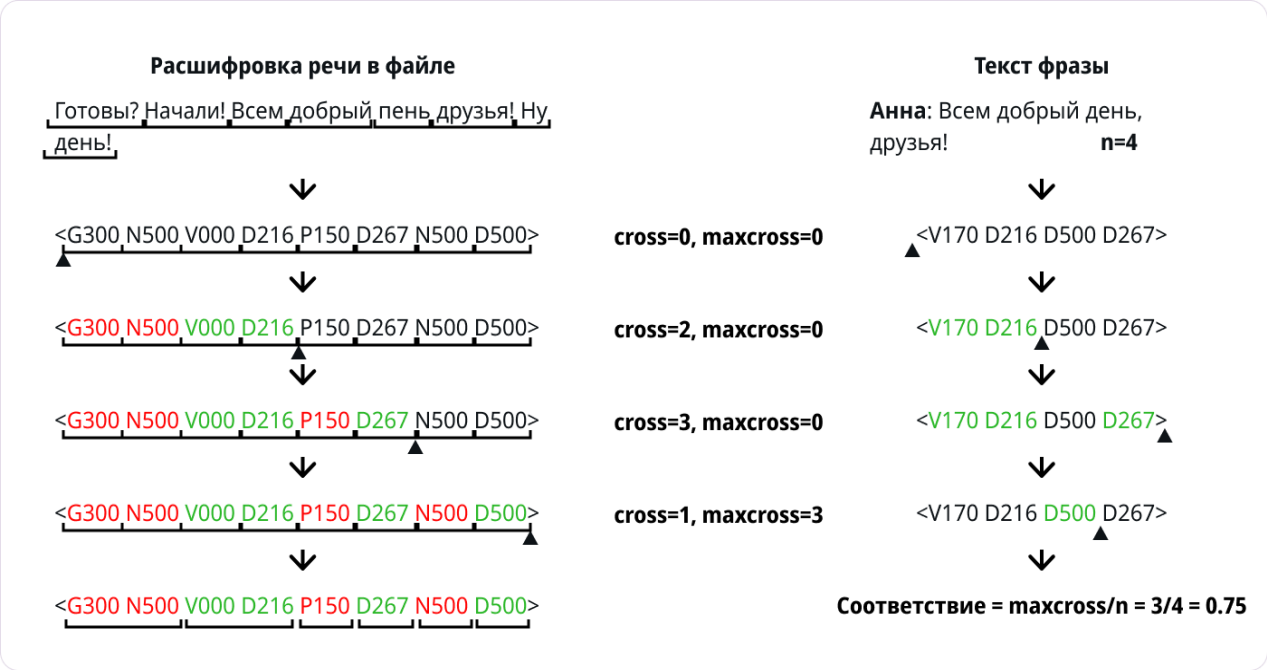


Рис. 4. Определение соответствия фразе с подсчетом сквозной длины

Общее соответствие файла сценарию считается как сумма длин сегментов, продолжающих друг друга. Теперь алгоритм при выборе подходящей фразы опирается именно на максимальную сквозную длину.

СТАНДАРТ ОФОРМЛЕНИЯ СЦЕНЫ ЭКСПОЗИЦИИ

Для сортировки по сценам-экспозициям было необходимо классифицировать изображение и звук в видеозаписи. Чтобы определить предметную область, которую необходимо покрыть классификацией, были изучены стандарты оформления сценариев. В голливудском формате сцена экспозиции выглядит следующим образом: сверху пишутся ИНТ (интерьер) либо НАТ (натура), затем название локации и время суток. После чего пишется текст сцены (рис. 5) [15].

ИНТ. УЮТНАЯ КОФЕЙНЯ - ДЕНЬ

Девушка сидит за столом возле окна, мирно пьет кофе и читает книгу. На столе рядом с кружкой кофе лежит открытая книга и кусочек шоколадного торта на тарелке. Она улыбается, погруженная в чтение. Вдруг её внимание привлекает смешной эпизод в книге, и она смеется.

Входит МУЖЧИНА с букетом цветов. Он нервничает, проверяет свои часы и смотрит по сторонам в поисках кого-то. Наконец, его взгляд останавливается на девушке. Он подходит и ставит цветы на стол перед ней, застенчиво улыбаясь.

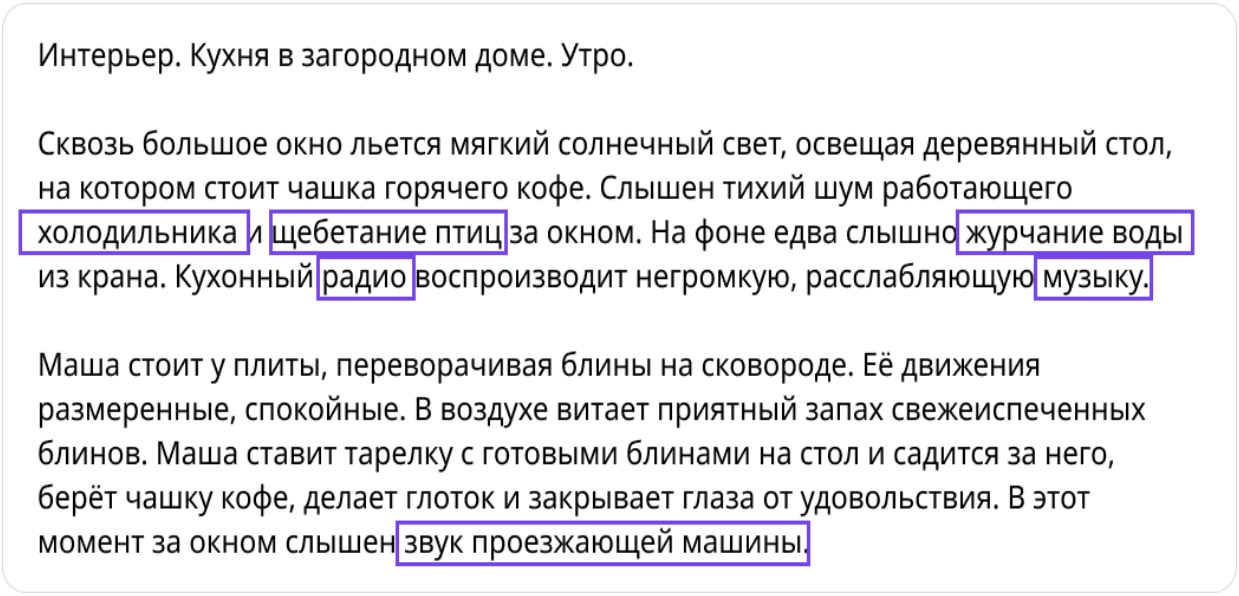
Рис. 5. Пример сцены экспозиции

Опираясь на этот стандарт, необходимо было классифицировать медиафайлы и тексты блоков экспозиции, а затем найти способ сравнить определенные классы и на основе этого подобрать самую подходящую сцену для файла.

ПРОГРАММНЫЕ МЕХАНИЗМЫ КЛАССИФИКАЦИИ АУДИОФАЙЛОВ

Классификация аудио необходима, чтобы определять объекты на видео, которые не присутствуют в кадре, а находятся вне зоны видимости камеры. Часто в сценариях прописывается краткое описание атмосферы сцены, в том числе через

звуки, что является возможностью для автоматизированной классификации сцены по звуку (рис. 6).


Интерьер. Кухня в загородном доме. Утро.

Сквозь большое окно льется мягкий солнечный свет, освещая деревянный стол, на котором стоит чашка горячего кофе. Слышен тихий шум работающего холодильника и щебетание птиц за окном. На фоне едва слышно журчание воды из крана. Кухонный радио воспроизводит негромкую, расслабляющую музыку.

Маша стоит у плиты, переворачивая блины на сковороде. Её движения размеренные, спокойные. В воздухе витает приятный запах свежеиспеченных блинов. Маша ставит тарелку с готовыми блинами на стол и садится за него, берёт чашку кофе, делает глоток и закрывает глаза от удовольствия. В этот момент за окном слышен звук проезжающей машины.

Рис. 6. Пример описания атмосферы сцены через звуки

Для классификации аудио был выбран SoundAnalysis Framework [16] из-за своей оптимизации для языка Swift и широкого набора определяемых классов – он содержит 303 класса аудио. Был реализован модуль классификации файлов с поддержкой многопоточности. SoundAnalysis классифицирует аудио по сегментам, поэтому для общей уверенности в классе считается средняя уверенность по всем сегментам. После проведения тестов был установлен порог уверенности, равный 0,3.

Также был реализован механизм, который самостоятельно определяет, к какому типу сцены отнести файл и по какому алгоритму его сортировать. В модели классификации аудио есть класс speech, и если уверенность в нём больше заданного порога, то файл будет считаться файлом диалога и сортироваться по методу, описанному в предыдущем разделе. Иначе файл считается файлом сцены экспозиции и обрабатывается по правилам классификации таких файлов. Было проведено тестирование на наборе данных из 421 аудиофайла и 218 видеофайлов, после чего для аудиофайлов был установлен порог уверенности 30%, а для видеофайлов – 60%.

ПРОГРАММНЫЕ МЕХАНИЗМЫ ОБНАРУЖЕНИЯ ОБЪЕКТОВ НА ВИДЕОЗАПИСИ

Нахождение объектов на видеозаписи является ключевым способом классификации файлов, так как определяет, что находится в кадре, и оно по определению включается в сцену экспозиции, даже если у неё не будет описания других пунктов.

Для нахождения объектов на видео было решено выбрать оптимизированную для языка программирования Swift модель YOLOv8 [17], так как эта модель специализируется на анализе изображений со скоростью, достаточной, чтобы обнаруживать объекты в реальном времени. В разработанном программном решении была использована сконвертированная в CoreML модель YOLOv8X, которая обеспечивает лучшую точность среди предобученных моделей.

Так как модель работает только с изображениями, был разработан модуль, который с помощью библиотеки Vision выбирает кадр из каждой секунды видео, находит объекты на выбранных изображениях с использованием YOLO и потом оценивает степень уверенности относительно всех кадров.

Этот модуль также подсчитывает количество объектов одного класса в кадре, присваивая файлу дополнительные классы с количественным параметром. К примеру, пусть в видео есть два действующих лица. Модуль определяет класс «person» с уверенностью 100%, а класс «2 person» — с уверенностью 85%. Это необходимо для более точного сравнения со сценарием, чтобы фильтровать сцены с другим количеством людей, определять правильное количество объектов в кадре и так далее.

ПРОГРАММНЫЕ МЕХАНИЗМЫ КЛАССИФИКАЦИИ ДЕЙСТВИЙ НА ВИДЕОЗАПИСИ

Результаты определения объектов на видеозаписи соответствуют существительным в описании сцены экспозиции, в то время как результаты классификации действий должны охватывать глаголы. Поэтому для успешного определения сцены это также необходимый пункт.

Классификация действий на видео происходит с использованием библиотеки Visual Action Kit [18]. Однако была выявлена следующая проблема: библиотека принимает на вход и распознаёт только видео длительностью не более 300

кадров (6–12 секунд в зависимости от частоты кадров). Поэтому она была доработана механизмом разделения входного файла на сегменты длиной 300 кадров и последовательной обработки каждого из них. Это позволило распознавать видео любой длины. Уверенность в классе для всего видео определялась как средняя уверенность по всем сегментам.

ПРОГРАММНЫЕ МЕХАНИЗМЫ КЛАССИФИКАЦИИ ВРЕМЕНИ СУТОК

Все вышеперечисленные механизмы классификации нужны для непосредственной работы алгоритма сравнения, поскольку их результаты описывают действие, происходящее в сцене. Однако количество уникальных классов, по которым сможет определиться сцена экспозиции, сильно ограничено. Было необходимо дополнить это количество так, чтобы новые классы позволяли сужать круг выбора подходящей сцены, что повысило бы точность определения.

Для решения данной задачи предложен программный механизм, основанный на определении классов локаций и времени суток. При оформлении сценария по ранее упомянутому стандарту они будут у каждой сцены, и для файла будет выбираться сцена только среди тех, у которых те же время суток и локация.

Был собран датасет из 4 классов (утро/день/вечер/ночь) размером 1668 изображений (таблица 1).

Таблица 1. Описание датасета для обучения модели классификации времени суток

Класс	**Тренировочные данные**	**Тестовые данные**
day	798	38
evening	781	36
morning	790	63
night	800	51

Модель была обучена с помощью CreateML [19], ее точность на тестовом датасете показала 71% (таблица 2). На реальных данных этого оказалось достаточно, так как распознаваться в итоге будут не одно изображение, а набор кадров

из видео. И уверенность в определённом классе может повыситься, а неудачные кадры с ложным определением не окажут сильного влияния на итоговый результат.

Таблица 2. Результат обучения модели классификации времени суток

Accuracy	**Precision**	**Recall**	**F1 Score**
0.71	0.72	0.71	0.71

Был разработан модуль, который обрабатывает кадры в видеофайле и с помощью обученной модели определяет время суток, после чего для каждого класса времени суток вычисляет среднюю уверенность по всем кадрам.

ПРОГРАММНЫЕ МЕХАНИЗМЫ ОПРЕДЕЛЕНИЯ ЛОКАЦИИ НА ВИДЕОЗАПИСИ

Для определения локации был найден проект indoor-scene-classification, который содержал датасет из 64 классов интерьеров [20]. Было принято решение на основе этого датасета обучить новую модель, предварительно дополнив датасет классами локаций экстерьеров. Итого в датасете было собрано 124 класса локаций (64 интерьера, 60 экстерьеров). Общий размер датасета – 18680 изображений.

Все локации в наборе классов являются подклассами двух родительских классов: interior и nature. Поэтому в случаях, когда модель не сможет точно определить локацию на видео, она, по крайней мере, определит, интерьер это или экстерьер, что уже наполовину сузит потенциальный круг поиска сцены (рис. 7).

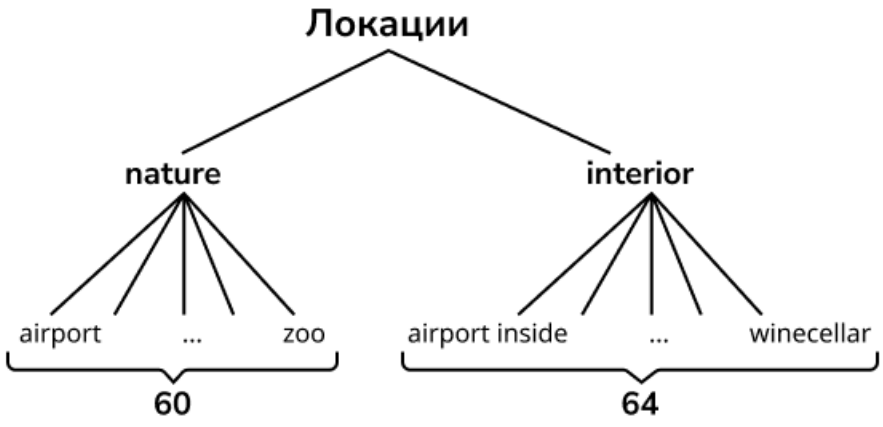


Рис. 7. Наследование классов локаций в датасете

С использованием CreateML была обучена модель классификации локаций на изображениях, тесты показали 76% точности на тестовом наборе данных (таблица 3). Был разработан модуль, который обрабатывает кадры видео и определяет его локацию, вычисляя среднюю уверенность в классе по всем кадрам.

Таблица 3. Результаты обучения модели классификации локаций

Accuracy	**Precision**	**Recall**	**F1 Score**
0.76	0.83	0.77	0.80

ПРОГРАММНЫЕ МЕХАНИЗМЫ КЛАССИФИКАЦИИ ТЕКСТА СЦЕНАРИЯ

В совокупности все модели классификации видео могут распознать 913 различных классов. Для сравнения файлов с текстом сценария было необходимо выделить то же множество классов из блоков текста. Но при этом нужно было учитывать, что в тексте могут использоваться синонимы слов или связанные по контексту слова или словосочетания.

С использованием механизмов обработки естественного языка wordnet из NLTK [21] для наборов классов каждой из моделей были отдельно составлены словари синонимов или связанных по контексту слов (рис. 8). Если хотя бы один из синонимов встретится в тексте блока экспозиции, то блоку присваивается класс по этому синониму.

```
day: день, дневное время, светлое время суток, деньки, среди дня, полдень, посреди дня
evening: вечер, вечернее время, закат, сумерки, предзакатное время, к вечеру, вечером
morning: утро, утреннее время, рассвет, заря, на заре, на рассвете, рано утром
night: ночь, ночное время, полночь, темное время суток, глубокая ночь, посреди ночи, ночью
```

Рис. 8. Пример словаря синонимов для классов времени суток

При обработке блока сцены текст очищается от пунктуации, все слова переводятся в начальную форму. Затем проводится поиск по каждому слову текста в каждом из словарей, и выявленные классы присваиваются блоку сценария. Таким образом, алгоритм не требователен к конкретному оформлению сцены по стандарту: достаточно просто упомянуть локацию в самом описании сцены или использовать слово, близкое по контексту.

Для классов, которые определялись моделью обнаружения объектов на видео, также был добавлен механизм подсчета количества объектов одного класса. Для локаций дополнительно задавался родительский класс. Для времени суток был реализован механизм, который отыскивает в тексте блока сцены формат времени (hh:mm), чтобы определить время суток в случаях, когда сценарист указал точное время событий в сцене.

Также нужно было учесть, что в сценарии присутствуют имена собственные, которые нужно определять как людей. Так как при загрузке текста сценария определяются диалоги, в которых указаны имена персонажей, которые произносят фразу, то в приложении хранится список всех персонажей, которые найдены в сценарии. Данные имена используются как синонимы к классу person (рис. 9).

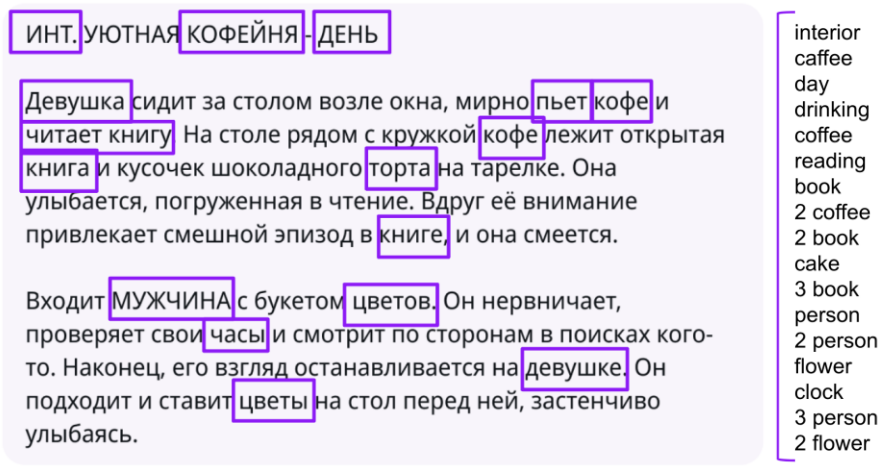


Рис. 9. Классификация блока экспозиции

ОПРЕДЕЛЕНИЕ СООТВЕТСТВИЯ ЭКСПОЗИЦИЙ СЦЕНАРИЮ

В результате разработки механизмов классификации и у файлов, и у блоков сценария потенциально могут определиться классы из полученного набора в 913 классов и дополнительных классов для обнаруженных объектов во множественном числе. Поэтому был необходим алгоритм, который сравнит классы файла с классами всех сцен и подберет лучшую сцену. При этом стоило учитывать, что некоторые классы могут повторяться в некоторых сценах чаще, чем другие.

Для решения этой задачи был выбран метод сравнения по косинусному сходству векторов, составленных с помощью списка классов для каждого файла или сцены. Вектора было решено строить из индексов TF-IDF, которые показывают наличие класса в элементе и важность этого класса относительно всех

остальных [22]. Используя такой подход, удастся лучше определять сцену для файла, опираясь на уникальные классы, которые не встречаются нигде больше или встречаются реже.

Был реализован модуль, который принимает файлы с набором определенных для них классов. Для каждого файла определяется область поиска фразы, опираясь на классы времени суток и локации, если они есть. Далее набор остальных классов каждого файла преобразуется в вектор TF-IDF индексов, и вычисляется косинусное сходство этого вектора с вектором классов каждой сцены (рис. 10). Сцена, с которой сходство наибольшее, определяется как подходящая.

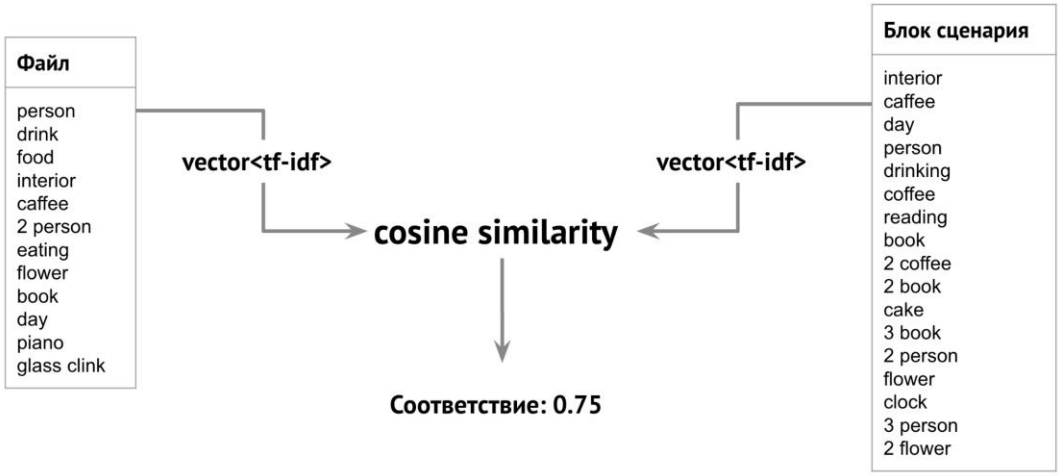


Рис. 10. Алгоритм классификации сцены экспозиции

ОЦЕНКА ТОЧНОСТИ И ПРОИЗВОДИТЕЛЬНОСТИ СИСТЕМЫ

С использованием библиотеки WhisperKit скорость расшифровки одного файла разнится от ¼ до ½ от длины файла. Для коротких файлов расшифровка происходит немного медленнее упомянутых скоростей. Это объясняется необходимостью инициализировать модель перед расшифровкой. Инициализация занимает несколько лишних секунд, что незаметно при расшифровке длинных файлов, но видно при обработке множества коротких.

Реализованный модуль с очередью файлов был протестирован на наборе из 421 аудиофайла общей длительностью более 9 часов. Расшифровка всех файлов была завершена примерно за 50 минут.

Модель классификации аудиофайла была протестирована на этом же наборе данных. Длительность распознавания одного файла составляет 0.004 от

длительности этого файла. Все файлы были классифицированы примерно за 4 минуты.

Модели классификации видео были протестированы на наборе данных из 218 видеофайлов. Модели определения локации и времени суток обрабатывают файл практически моментально. Время обработки файла моделью нахождения объектов на видео в среднем составляет 0.15 от длительности файла, а моделью определения действий на видео – 0.29 от длительности файла.

После расшифровки речи и классификации видео и аудио каждый файл был вручную связан с фактической сценой из сценария, подходящей для него. В сценарии всего имелось 820 фраз и 407 сцен-экспозиций.

При этом из общего набора данных в 639 файлов имелось:

- 378 файлов, фактически относящиеся к фразам;
- 187 файлов, фактически относящиеся к сценам экспозициям;
- 73 сторонних медиа файла, не связанные со сценарием.

Успешным результатом работы считается то, что разработанная система правильно определит фразу или сцену-экспозицию для большинства соответствующих файлов. Чтобы минимизировать число ошибок классификации сцен, на поиск и исправление которых пользователю потребовалось бы дополнительное время, было принято решение задать определённый порог уверенности. В случаях, когда алгоритмы недостаточно уверены в правильности классификации, сцена для файла не распознается. Программа сразу отобразит все подобные файлы, чтобы пользователь смог сортировать их вручную. Порог уверенности также должен решить проблему в тех случаях, когда были импортированы сторонние файлы, не связанные со сценарием. Для таких файлов тоже не нужно определять сцену.

Был создан отладочный механизм, который при получении результатов алгоритма классификации маркирует файлы одной из пяти меток:

1. Сцена распознана правильно;
2. Сцена для файла не определена;
3. Сцена распознана неправильно;
4. Сцена для стороннего файла не определена;
5. Сцена для стороннего файла определена.

После сбора данных о соответствии всех определенных файлов с использованием python-библиотеки sklearn была построена ROC-кривая, и по ней вычислен оптимальный порог, который необходимо задать для улучшения результатов алгоритма [23]. Так был вычислен порог в 34% (рис. 11).

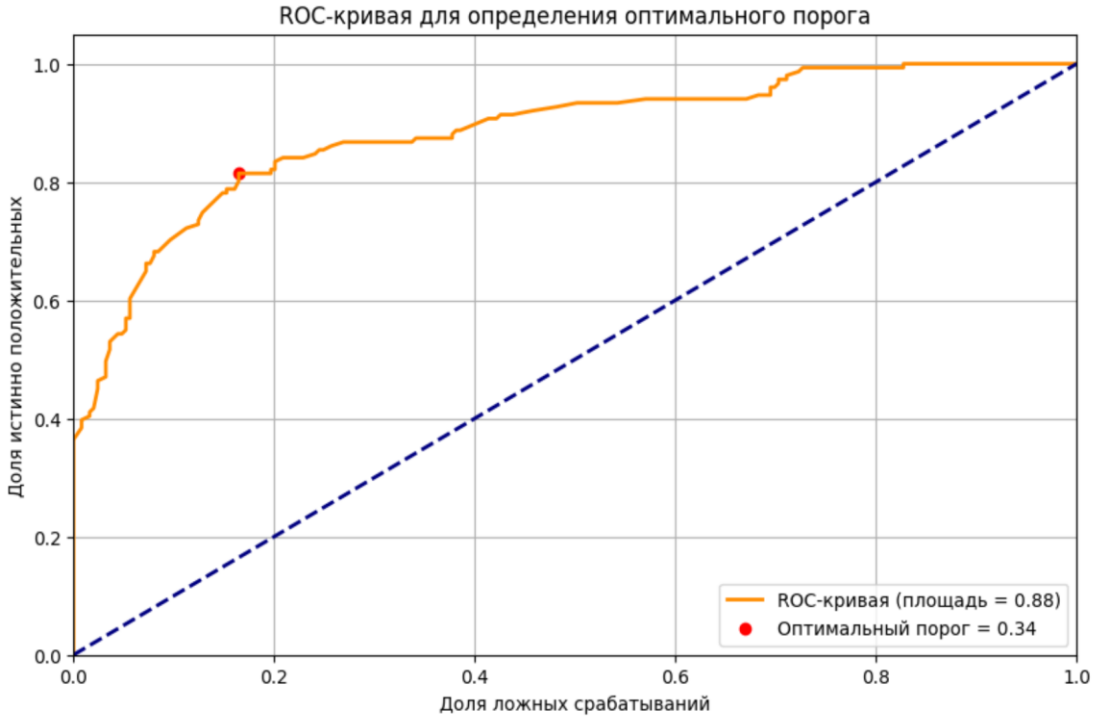


Рис. 11. Определение оптимального порога

Также были учтены случаи для сопоставления с длинными фразами. В этих случаях порог должен работать только тогда, когда у файла соответствие сценарию меньше определенного количества слов (в процессе тестирования было вычислено это количество — 8). Таким образом, если у файла с фразой сходится 8 и более слов, но соответствие ниже порога только потому, что сама фраза состоит из нескольких десятков слов, она всё равно присваивается файлу. Это необходимо в ситуациях, когда длинный монолог снимают в несколько дублей, и в одном дубле будет не вся фраза.

После исправлений в алгоритме была проведена повторная проверка, и теперь результат стал удовлетворительным: большинству файлов была правильно подобрана фраза. А у большинства файлов, для которых не удалось найти фразу правильно, фраза не определилась вовсе во избежание ошибок (рис. 12).

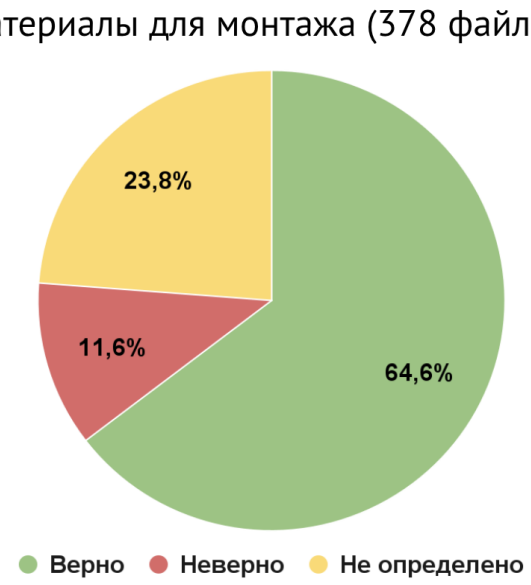


Рис. 12. Оценка точности механизма определения фразы

Алгоритм классификации сцен-экспозиций также работает правильно в большинстве случаев. К тому же у него меньший процент ошибок в сравнении с классификацией фраз (рис. 13).

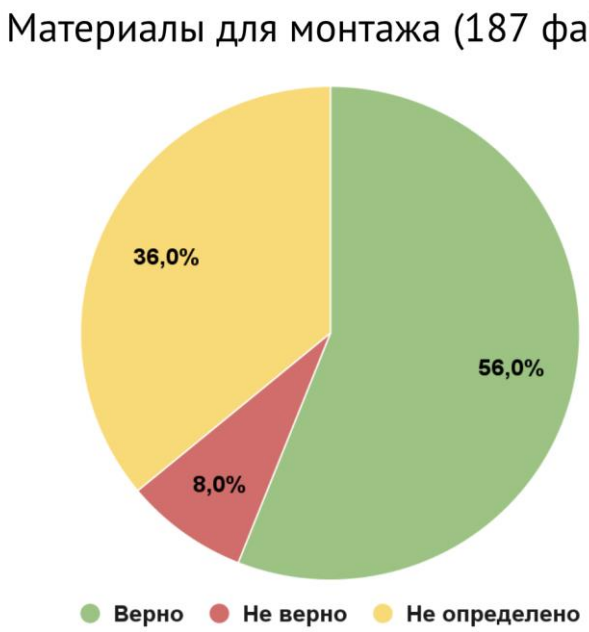


Рис. 13. Оценка точности механизма определения сцены экспозиции

Тестирование на наборе сторонних файлов также дало хороший результат: подавляющее большинство таких файлов не обрабатывается алгоритмами (рис. 14).

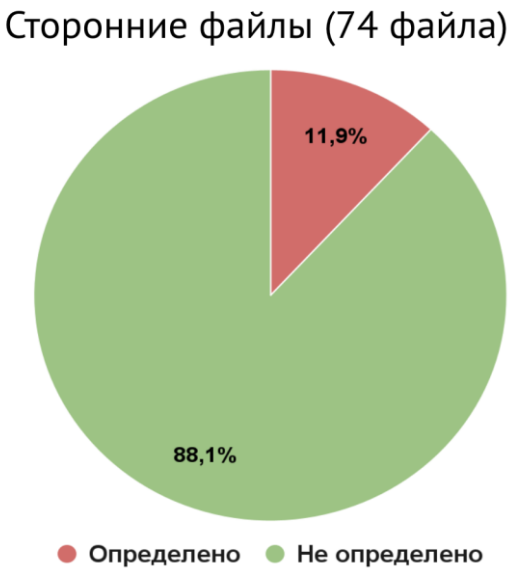


Рис. 14. Оценка точности механизма обнаружения сторонних файлов

Чаще всего оставшиеся ошибки возникают с фразами, состоящими из 1–3 слов, так как вероятность случайного произнесения хотя бы одного из этих слов высокая, и у коротких фраз порог преодолевается очень быстро. К тому же тестирование происходило на сценарии, где фраз почти вдвое больше, чем файлов. И некоторые фразы могут быть похожи друг на друга. На менее объемном сценарии точность алгоритма может быть выше.

ЗАКЛЮЧЕНИЕ

Представлено программное решение, которое позволяет автоматизировать сортировку медиа файлов и оценку соответствия их сценарию перед монтажом видеопроекта.

Программное решение было протестировано на наборе из более чем 400 аудиофайлов и 200 видеофайлов, в результате чего правильно определило 64% файлов для сцен-диалогов и 56% файлов для сцен-экспозиций. Таким образом, за

счет автоматизации удалось сократить время, затрачиваемое на сортировку, на 50%.

СПИСОК ЛИТЕРАТУРЫ

1. Монтаж | Теория кино // Студия “Кинокафе”, 2019. URL: https://www.kinocafe.ru/theory/?tid=1221.
2. Сколько времени длится съемка сцены? // Celebrity.fm, 2020. URL: https://celebrity.fm/ru/how-long-does-filming-a-scene-take.
3. Кинопродюсерство в кинематографии // Allbest, 2020. URL: https://revolution.allbest.ru/culture/01202211_2.html#text.
4. Создание документального фильма: как работают монтажеры – победители «Эмми» // Sayhi, 2019. URL: https://say-hi.me/24-kadra/sozdanie-dokumentalnogo-filma-kak-rabotayut-montazhyory-pobediteli-emmi.html.
5. Media Composer // AVID, 2023. URL: https://ch.avid.com/media-composer.
6. Как правильно оформить сценарий – правила с примерами // BandBand, 2023. URL: https://bandband.ru/blog/kak-oformit-scenarij.
7. Whisper // OpenAI, 2022. URL: https://openai.com/research/whisper.
8. OpenAI Whisper // GitHub, 2022. URL: https://github.com/openai/ whisper.
9. SwiftWhisper // GitHub, 2023. URL: https://github.com/exPHAT/ SwiftWhisper.
10. WhisperKit // GitHub, 2024. URL: https://github.com/argmaxinc/ WhisperKit.
11. An Effective Review of Phonetics Algorithms // ResearchGate, 2023. URL: https://www.researchgate.net/publication/375967141_An_Effective_Review_of_Phonetics_Algorithms.
12. Russian Soundex // GitHub, 2022. URL: https://github.com/roddar92/ russian_soundex.
13. Про многопоточность 2. GCD // Habr, 2021. URL: https://habr.com/ru/articles/578752/.
14. Combine // Apple Developer Documentation, 2024.

URL: https://developer.apple.com/documentation/combine.

15. Как правильно оформить сценарий – правила с примерами | Основные сценарные разделы // BandBand, 2023.
URL: https://bandband.ru/blog/kak-oformit-scenarij#dirs.

16. Sound Analysis // Apple Developer, 2022.
URL: https://developer.apple.com/documentation/soundanalysis.

17. YOLOv8 // Ultralytics, 2023.
URL: https://docs.ultralytics.com/ru/ models/yolov8.

18. VisualActionKit // GitHub, 2020.
URL: https://github.com/lukereichold/ VisualActionKit.

19. Creating an Image Classifier Model // Apple Developer, 2021.
URL: https://developer.apple.com/documentation/createml/creating-an-image-classifier-model.

20. keremberke/indoor-scene-classification // Hugging Face, 2022.
URL: https://huggingface.co/datasets/keremberke/indoor-scene-classification.

21. Sample usage for wordnet // NLTK, 2023.
URL: https://www.nltk.org/howto/wordnet.html.

22. Извлечение признаков из текстовых данных с использованием TF-IDF // Habr, 2023. URL: https://github.com/roddar92/russian_soundex.

23. Selecting the Optimal Probability Threshold for a Classification Model, ROC Curve Analysis and KS Score // DataScienceByExample, 2023.
URL: https://www.datasciencebyexample.com/2023/04/19/choose-thresold-for-classification-model-with-ROC-analysis.

AUTOMATION OF FOOTAGES SORTING BY SCREENPLAY TEXT FOR VIDEO EDITING

A. D. Nemanov[1] [0009-0004-1840-2347], I. S. Shakhova[2] [0000-0003-1591-5767]

[1, 2]*Institute of Information Technology and Intelligent Systems, Kazan Federal University, Kazan;*

[1]andrewoch@yandex.ru, [2]is@it.kfu.ru

Abstract

The video editing process involves numerous labor-intensive operations for sorting and preparing footages, requiring significant time investment. This article describes the development of a software solution that uses machine learning technology to automate these processes.

The primary focus is on creating a system capable of classifying and sorting media files according to the screenplay text, thereby increasing the efficiency of material preparation for editing. The system includes modules for speech recognition, audio and video classification, and algorithms for determining screenplay compliance.

Testing showed that the proposed system correctly classifies media files in most cases, significantly reducing rough-cut editing time.

Keywords*: video editing, automation, machine learning, speech recognition, audio classification, video classification, coreml, parallel computing, screenplay, soundex, tf-idf, cosine similarity, natural language processing.*

REFERENCES

1. Montazh | Theory of Cinema // Studio "Kinocafe", 2019. URL: https://www.kinocafe.ru/theory/?tid=1221.
2. Skol'ko vremeni dlitsya syemka stseny? // Celebrity.fm, 2020. URL: https://celebrity.fm/ru/how-long-does-filming-a-scene-take.
3. Kinoproduyserstvo v kinematografii // Allbest, 2020. URL: https://revolution.allbest.ru/culture/01202211_2.html#text.
4. Sozdanie dokumental'nogo fil'ma: kak rabotayut montazhery-pobediteli "Emmi" // Sayhi, 2019. URL: https://say-hi.me/24-kadra/sozdanie-dokumentalnogo-filma-kak-rabotayut-montazhyory-pobediteli-emmi.html.
5. Media Composer // AVID, 2023. URL: https://ch.avid.com/media-composer.
6. Kak pravil'no oformit' stsenaariy – pravila s primerami // BandBand, 2023. URL: https://bandband.ru/blog/kak-oformit-scenarij.
7. Whisper // OpenAI, 2022. URL: https://openai.com/research/whisper.
8. OpenAI Whisper // GitHub, 2022. URL: https://github.com/openai/ whisper.

9. SwiftWhisper // GitHub, 2023.
URL: https://github.com/exPHAT/ SwiftWhisper.
10. WhisperKit // GitHub, 2024.
URL: https://github.com/argmaxinc/ WhisperKit.
11. An Effective Review of Phonetics Algorithms // ResearchGate, 2023.
URL: https://www.researchgate.net/publication/375967141_An_Effective_Review_of_Phonetics_Algorithms.
12. Russian Soundex // GitHub, 2022.
URL: https://github.com/roddar92/ russian_soundex.
13. Pro mnogopotochnost' 2. GCD // Habr, 2021.
URL: https://habr.com/ru/articles/578752/.
14. Combine // Apple Developer Documentation, 2024.
URL: https://developer.apple.com/documentation/combine.
15. Kak pravil'no oformit' stsenariy – pravila s primerami | Osnovnye stsenarnye razdely // BandBand, 2023.
URL: https://bandband.ru/blog/kak-oformit-scenarij#dirs.
16. Sound Analysis // Apple Developer, 2022.
URL: https://developer.apple.com/documentation/soundanalysis.
17. YOLOv8 // Ultralytics, 2023.
URL: https://docs.ultralytics.com/ru/ models/yolov8.
18. VisualActionKit // GitHub, 2020.
URL: https://github.com/lukereichold/ VisualActionKit.
19. Creating an Image Classifier Model // Apple Developer, 2021.
URL: https://developer.apple.com/documentation/createml/creating-an-image-classifier-model.
20. keremberke/indoor-scene-classification // Hugging Face, 2022.
URL: https://huggingface.co/datasets/keremberke/indoor-scene-classification.
21. Sample usage for wordnet // NLTK, 2023.
URL: https://www.nltk.org/howto/wordnet.html.
22. Izvlechenie priznkov iz tekstovykh dannykh s ispol'zovaniem TF-IDF // Habr, 2023. URL: https://github.com/roddar92/russian_soundex.
23. Selecting the Optimal Probability Threshold for a Classification Model, ROC Curve Analysis and KS Score // DataScienceByExample, 2023.

URL: https://www.datasciencebyexample.com/2023/04/19/choose-thresold-for-classification-model-with-ROC-analysis.

СВЕДЕНИЯ ОБ АВТОРАХ

НЕМАНОВ Андрей Дмитриевич – бакалавр программной инженерии Института информационных технологий и интеллектуальных систем Казанского федерального университета.

Andrey Dmitrievich NEMANOV – Bachelor of Software Engineering, Institute of Information Technology and Intelligent Systems, Kazan Federal University.

email: andrewoch@yandex.ru

ORCID: 0009-0004-1840-2347

ШАХОВА Ирина Сергеевна – старший преподаватель кафедры программной инженерии Института информационных технологий и интеллектуальных систем Казанского федерального университета, г. Казань.

Irina Sergeevna SHAKHOVA – Senior Lecturer at the Department of Software Engineering, Institute of Information Technology and Intelligent Systems, Kazan Federal University, Kazan.

email: is@it.kfu.ru

ORCID: 0000-0003-1591-5767

Материал поступил в редакцию 19 июля 2024 года