（様式１）

令和元年度愛媛大学学生による調査・研究プロジェクト（プロジェクトＥ）研究成果報告書

令和　2　年　4　月　31　日

愛　媛　大　学　長　殿

プロジェクト代表者 氏　　名	理工学研究科　博士前期課程課程電子情報工学専攻 中山颯
指導教員氏名	所属　大学院理工学研究科 樋上　喜信

プロジェクト名：
コメント処理機能を有する分散型動画配信プラットフォームの開発

調査・研究の概要：

１．問題意識

現在，世界では様々な無料，有料動画配信プラットフォームがあり，YouTube[1]，ニコニコ動画[2]などが有名である．これらのプラットフォームは，中央集権的なシステムで，動画を各会社が運営するサーバにアップロードすることで，動画視聴が可能となる．中央集権的なシステムでは，動画が全て，単一の管理者によって管理され，管理者は自由にコンテンツを操作できる．そのため，管理者の都合や国家からの圧力等により，コンテンツが制限されたりすることがある[3]．

また，動画配信サイトや掲示板サイトの多くは，コメント，質問機能を有し，ユーザによってコンテンツに関する活発な議論が行われている．しかし，掲示板や動画サイト等のコメント機能は，匿名でコメントができるという手軽さから，心無い誹謗中傷コメントも多く，いじめや自殺などの社会問題につながっているという問題点がある[4]．

２．目的

本プロジェクトでは，誰でも簡単に動画サイトを運営できるプラットフォームを構築し，また，ソースコードをWeb上に公開することで，誰でも簡単に動画サイトを構築できる仕組みを提供する．それぞれ独立した権限を持つサーバからコンテンツを配信することで，中央集権的な管理者が存在しない，自由な動画の配信が可能となる．さらに，各ノードごとを結合させ，一つの動画プラットフォームとすることで，それぞれのノードで権利が分散された新しい動画プラットフォームの構築を目指す．

さらに，匿名での誹謗中傷コメントの問題を解決するために，本システムでは，自然言語処理により，誹謗中傷コメントをフィルタリングし，動画投稿者の設定したレベルに応じて，リアルタイムでコメントの公開，非公開を設定する．ユーザによる禁止単語の追加や，コメントに含まれる単語や文章構成から，コメント自体の評価を行い，フィルタリングを行う．本システムを利用して動画投稿を行うことで，動画の投稿者は投稿した動画の匿名コメントによる誹謗中傷のリスクを減らすことができ，ユーザも動画に対するコメントで不快な思いをすることなく，動画視聴やコメントを通した議論をすることができる．

３．方法

本プロジェクトのシステムは，コンテナ仮想化技術の一つであるDocker[5]を用いてアプリケーションを開発することにより，動作するOSを選ばずに，Windows，Mac，Linuxなど複数の環境でデプロイすることができる．

動画のストリーミングには，RTMP(Real Time Messaging Protocol)を用いて通信を行い，動画自体の形式は，HLS(HTTP Live Streaming)形式を用いる．また，動画サイトシステム自体はPHPフレームワークであるLaravelを用いて構築する．既存の動画サイトに可能な限り近づけることで，簡単に操作できるものを構築する．

コメントのフィルタリング機能には，禁止ワードによるフィルタリングに加え，Google Cloudが提供するNatural Language API[6]を用い，利用者が設定したレベルに応じたコメントのフィルタリングを行う．

参考文献

[1] Youtube, https://www.youtube.com/, (参照 2020-05-01)
[2] niconico(ニコニコ), https://www.nicovideo.jp/, (参照 2020-05-01)
[3] 動画の削除に関するトラブルシューティング, https://support.google.com/youtube/answer/6395024?hl=ja (参照 2020-04-28)
[4] 2chの誹謗中傷の裁判一覧まとめ, http://xn--2ch-5q0fn79k.net/, (参照 2020-04-28)
[5] Docker, https://www.docker.com/, (参照 2020-04-28)
[6] Natural Language API, https://cloud.google.com/natural-language, (参照 2020-04-30)

研究成果

本プロジェクトでは，可能な限り簡単に動画配信環境の構築を行うことのできるソフトウェアを提案し，開発を行った．動画配信システムは，コンテナ仮想化ソフトであるDocker[1]を用いて構築を行った．Dockerのインストールが可能な環境であれば，どの様なプラットフォームでも動作させることが可能である．

本システムは，動画の投稿機能，ライブ配信機能とそれらを閲覧する再生プレイヤー機能や投稿されたコンテンツに対するコメント機能とコメントフィルタリング機能を有する．

動画の再生には，ストリーミング方式を用い，動画のダウンロードと再生を同時に行うことが可能である．また，ライブ配信に関しては，OBS (Open Broadcaster Software) [2]等を用いることで，10〜20秒程度の遅延での動画配信を行うことが可能である．

コメント機能では，Pusher[3]を用いることで，動画を閲覧しているユーザ同士がリアルタイムにコメントを共有することを可能にした．

コメントフィルタリング機能では，動画の配信者が定めたレベルに応じて，三段階のコメントフィルタリングを行う．フィルタリングを行うために，コメントの文脈や単語から，コメントがポジティブやネガティブであるという度合いや感情の起伏の度合いを数値的に算出し，定めたレベルに応じて，投稿されたコメントの表示，非表示の切り替えを行う．

本プロジェクトで構築したソフトウェアを用いることで，独自のコメントフィルタリング機能を持った動画サイトを公開することができ，自分の好きな動画コンテンツをアップロードしたり，ライブ配信することが可能となった．

参考文献

[1] Docker, https://www.docker.com/, (参照 2020-04-28)
[2] OBS, https://obsproject.com/ja, (参照 2020-04-28)
[3] Pusher, https://pusher.com/, (参照 2020-04-28)

今後の課題

本プロジェクトで開発したソフトウェアにより，動画の配信環境を比較的簡単に構築できる様になった．しかし，課題としては，以下の項目が挙げられる．

１．ライブ配信の遅延が大きい
２．コメントフィルタリング機能の改善
３．分散型のプラットフォームとしての動作

１に関しては，動画のストリーミングプロトコルとして，RTMPを採用しているが，遅延が大きいため，より配信遅延の少ないWebRTCを使用した実装を検討している．
２に関しては，現在，コメントのフィルタリングにNatural Language APIを使用しているが，今後は言語コーパス等からデータを収集し，Deep Learningを用いた独自の分類器を構築し，さらに精度の高いコメントのフィルタリングを行うことを検討している．また，コメントの分類だけでなく，動画やコメントの内容によって，配信や動画が楽しいものか，悲しいものかなどを分類し，ステータスとして表示できる機能などを追加していきたい．
３に関しては，本プロジェクトで目標としていた，各ノードごとを結合させ，一つの動画プラットフォームとして動作させる機能を実装することができなかった．今後，各ノードごとにアップロードされている動画を共有する仕組みを構築することが課題である．

指導教員からのコメント

本研究は，動画投稿プラットフォームを分散型管理方式で構築しようとしたものであり，現在主流である中央集権型管理方式の問題点を解決する独創的な取り組みである．また，自然言語処理技術によるフィルタリング機能を導入し，投稿された動画に対する不快なコメントを自動的に除去するなど，工夫した機能が組み込まれている．本研究は，実用性かつ有用性の高いシステムを実現しており，今後，残された課題を解決することで，ビジネスにも発展することが期待できるような多大な成果が得られている．

目次

第 1 章

はじめに

現在，世界では様々な無料，有料動画配信プラットフォームがあり，YouTube[1], niconico[2] などが有名である．これらのプラットフォームは，中央集権的なシステムで，動画を各会社が運営するサーバにアップロードすることで，動画視聴が可能となる．中央集権的なシステムでは，動画が全て，単一の管理者によって管理され，管理者自由にコンテンツを操作される．そのため，管理者の都合や国家からの圧力等により，コンテンツが制限されたりすることがある [3].

また，動画配信サイトや掲示板サイトの多くは，コメント，質問機能を有し，ユーザによってコンテンツに関する活発な議論が行われている．しかし，掲示板や動画サイト等のコメント機能は，匿名でコメントができるという手軽さから，心無い誹謗中傷コメントも多く，いじめや自殺などの社会問題につながっているという問題点がある [4].

本プロジェクトでは，誰でも簡単に動画サイトを運営できるプラットフォームを構築し，また，ソースコードを Web 上に公開することで，誰でも簡単に動画サイトを構築できる仕組みを提供する．それぞれ独立した権限を持つサーバからコンテンツを配信することで，中央集権的な管理者が存在しない，自由な動画の配信が可能となる．さらに，各ノードごとを結合させ，一つの動画プラットフォームとすることで，それぞれのノードで権利が分散された新しい動画プラットフォームの構築を目指す．

さらに，匿名での誹謗中傷コメントの問題を解決するために，本動画配信プラットフォームでは，自然言語処理により，誹謗中傷コメントをフィルタリングし，動画投稿者の設定したレベルに応じて，リアルタイムでコメントの公開，非公開を設定する．ユーザによる禁止単語の追加や，コメントに含まれる単語や文章構成から，コメント自体の評価を行い，フィルタリングを行う．本動画配信プラットフォームを利用して動画投稿を行うことで，動画の投稿者は投稿した動画の匿名コメントによる誹謗中傷のリスクを減らすことができ，ユーザも動画に対するコメントで不快な思いをすることなく，動画視聴やコメントを通した議論をすることができる．

本プロジェクトのシステムは，コンテナ仮想化技術の一つである Docker[5] を用いてアプリケーションを開発することにより，動作する OS を選ばずに，Windows, Mac, Linux など複数の環境でデプロイすることができる．動画のストリーミングには，RTMP(Real Time Messaging Protocol) を用いて通信を行い，動画自体の形式は，HLS(HTTP Live Streaming) 形式を用いる．また，本動画配信プラットフォームはバックエンドに PHP フレームワークである Laravel[8]，フロントエンドに Vue.js[9] を用いて構築する．既存の動画サイトに可能な限り近づけることで，簡単に操作できるものを構築する．コメントのフィルタリング機能には，禁止ワードによるフィルタリングに加え，Google Cloud が提供する Natural Language API[10] を用い，利用者が設定したレベルに応じたコメントのフィルタリングを行う．

第 2 章

イントロダクション

本章では，はじめに本プロジェクトに至った要因である現在の動画配信における問題と匿名性による問題の 2 つを紹介し，そして，本プロジェクトで解決したい目的と目標について述べる．

2.1 現在の動画配信における問題

2020 年 5 月現在，インターネット上には数多くの動画配信プラットフォームがあり，代表的なものに YouTube[1] や niconico[2] がある．これらの動画配信プラットフォームの特徴として，ユーザは動画配信を各会社が運営するサーバで行う中央集権型システムとなっている．中央集権型システムの場合，配信者は容易に動画配信を行うことができ，新規参入のハードルが下がるため，多くの配信者による動画配信がコンテンツを盛り上げることが予想される．YouTube では，YouTube Partner Program といったサービスにより，人気配信者に対して動画の再生数に応じた広告収入を分配するサービスを行っている．さらに，最近ではスーパーチャット機能やメンバーシップ機能などを実装し，配信者や視聴者，コンテンツの集約に成功している．例えば YouTube によると，世界中の利用状況を以下のようになっている [12].

> 毎月 20 億人以上のログイン済みユーザーが YouTube を利用しており、1 日あたりの動画視聴時間は 10 億時間を超え、視聴回数は数十億回にのぼります。

しかし，これら中央集権型システムの問題点として，ユーザコンテンツなどが単一の管理者によって管理されることである．サービスの維持などのため，広告収入は必要であり，動画コンテンツは広告主の意向に沿うものでなければならず，管理者はユーザコンテンツの規制を行っている．例えば，YouTube ポリシーでは暴力的で生々しいコンテンツは規制対象となっている [13]．また，細かくコミュニティガイドライン [14] を制定し，こ

のガイドラインに違反した場合，動画の削除やチャンネル停止などの措置が取られる．このように，中央集権型システムでは単一の管理者によって管理されることで，管理者の意向に沿わないコンテンツを表現することは出来ない．

ここで，中央集権型システムからの脱却を目指すプロジェクトを紹介する．コンテンツの制限などの問題により SNS サービス Twitter[15] において，中央集権型システムからの脱却を目指すプロジェクト Mastodon[16] である．Mastodon の特徴として，プロジェクトのソースコードが公開されており，デプロイするサーバはだれでも自由に運用でき，ユーザはサーバを選んで所属する．また，異なるサーバ間での利用者同士のコミュニケーションも容易に可能となっている．

このような動きを受けて，動画配信においても同様に行えないかと考えたとき，GitHub などで公開されている動画配信 OSS は，セルフホスティングできる形で動画配信プラットフォームが開発されているものの，分散型の動画配信プラットフォームは見られなかった．また，すでに開発されている動画配信プラットフォームのうち，開発ツールのバージョンが古くなって動作しないものや UI/UX が現代のユーザにはそぐわないものが散見された．このように，分散型で，大手の動画配信サービスに対抗する事を目指し，管理者が容易にデプロイでき，現代のユーザが UI/UX で満足する Open Source Software(OSS) で公開されている動画配信プラットフォームは現在存在しない．現状が続けば，今後，配信者と視聴者の表現の自由が失われてゆく可能性がある．

2.2　匿名性による問題

本節では匿名性による問題について考える．現在，多くのユーザが利用する動画配信サービスにおいて，匿名性を有するコメント機能が実装されている．この機能により，配信者と視聴者の距離が縮まり，コンテンツが盛り上がる要因の一つとなっている．

しかし，匿名性を有するサービス（動画配信や掲示板等）には誹謗中傷などの問題がある．実際に誹謗中傷による事例が起きている．例えば，2 ちゃんねるなどの電子掲示板で長期間に亘って誹謗中傷を受けたスマイリーキクチ中傷被害事件 [11] が挙げられる．

匿名性のある表現の手軽さと表現の自由，それに対して誹謗中傷などの表現はトレードオフ関係にある．現在の多くの動画配信サービスは誹謗中傷などの表現を大きく取り締まるため，コメントでの表現が大きく規制される問題がある．中央集権型の場合，ガイドラインがすべてのコメントに適用されてしまうことが，分散型では表現の手軽さと自由，誹謗中傷などの表現の取り締まりのバランスを各管理者が柔軟に調整出来ることが求められていると考えられる．

2.3　本プロジェクトの目的，目標

本節では，本プロジェクトが解決したい目的と目標について述べる．

2.3.1　本プロジェクトの目的

本プロジェクトでは分散型動画配信プラットフォームを OSS として開発し，広告主などによるコンテンツの規制や匿名コメント機能から表現の自由を獲得することを目的とする．

2.3.2　本プロジェクトの目標

目的に従い，目標を次のように定める．

1. OSS な動画配信プラットフォームの開発
2. コメント機能における表現と規制の調整機能の実装

1 つ目の目標である OSS な動画配信プラットフォームの開発は，動画配信サービスとして基本的な動画のアップロードとライブストリーミング，コメント機能が可能なものを容易にデプロイ出来るように OSS として開発する．

2 つ目の目標のコメント機能における表現と規制の調整機能の実装では，開発する動画配信プラットフォームのコメント機能にて，投稿されたコメントに対し，管理者が設定する規制が行われるようにすることで，管理者が表現の自由と規制のバランスを取ることができるようにする．

第 3 章

関連研究

この章では，本プロジェクトと関連の深い，動画サービスや動画プラットフォームを提供する OSS の特徴についてまとめる．また，誹謗中傷コメントの検出に関する研究について紹介する．

3.1　関連動画配信プラットフォームの特徴

現在，世界中で数多くの動画配信サービスが提供されており，誰でも無料で動画をアップロードし，視聴が可能な環境が整っている．その中でも，独自の動画配信環境を持つというニーズは一定数存在し，各動画配信サービスに似せて作られたクローンやオリジナルの OSS が公開されている．

本節では，各動画サービスや OSS の特徴について紹介する．以下の表 3.1 に，動画サービスや動画配信環境を提供する OSS の特徴を示す．動画プラットフォームを提供する OSS の中には，既存の動画サイトを似せて作ったものもあるが，CMS の一部として動画共有機能を持つものなどもある．

これらの OSS は，OS や動作環境を限定するものが多く，デプロイまでの環境構築に多くの手間がかかる事が予想される

3.2　誹謗中傷コメントの検出に関する研究

誹謗中傷コメントの検出に関する研究には，以下のようなものが挙げられる．

1. 乾　孝司．言語表現の使用実態を踏まえたソーシャルメディア上の誹謗中傷行為の検出に関する研究 [20]
2. 西原陽子．電子掲示板からの文脈を考慮した誹謗中傷コメントの抽出 [37]

表 3.1　各動画サービス，OSS の特徴と比較

プロダクト名（*は OSS）	特徴	動画再生	ライブ配信	コメント機能	コメントフィルタリング	動作環境
youtube[1]	Google LLC[17] が提供する世界最大の動画共有サービス	○	○	○	○	-
niconico[2]	株式会社ドワンゴ [18] が提供する日本最大級の動画配信サービス	○	○	○	○	-
*streama[21]	Netflix[19] に似た UI の動画管理システム	○	×	×	×	java1.8
*AVideo[22]	Youtube や Netflix[19] にた UI にカスタマイズ可能な動画管理システム	○	○	○	×	PHP 5.6+ MySQL 5.0+ Apache web server 2.x
*PHPmotion [23]	CMS 機能つきメディア共有システム，ビデオ共有，写真/画像共有，オーディオ共有，ブログなどの機能がある	○	×	○	×	Linux,PHP 5.2.x, MySQL, LAME MP3 Encoder, Libogg + Libvorbis など
*Joruri Video[24]	日本で開発された自治体向け動画管理システム簡単な操作で動画のアップロードと閲覧が可能．現在はソースコード非公開	○	×	×	×	CentOS 6, Apache 2.x, MySQL 5.x, Ruby 1.9.2, Ruby on Rails 3.0.0, FFmpeg など

第 4 章

方法

本章では，動画配信プラットフォームのシステム構成と使用技術，コメントフィルタリング機能の実装方法に関して述べる．

4.1　システム構成

本プロジェクトで開発するシステムの構成を図 4.1 に示す．

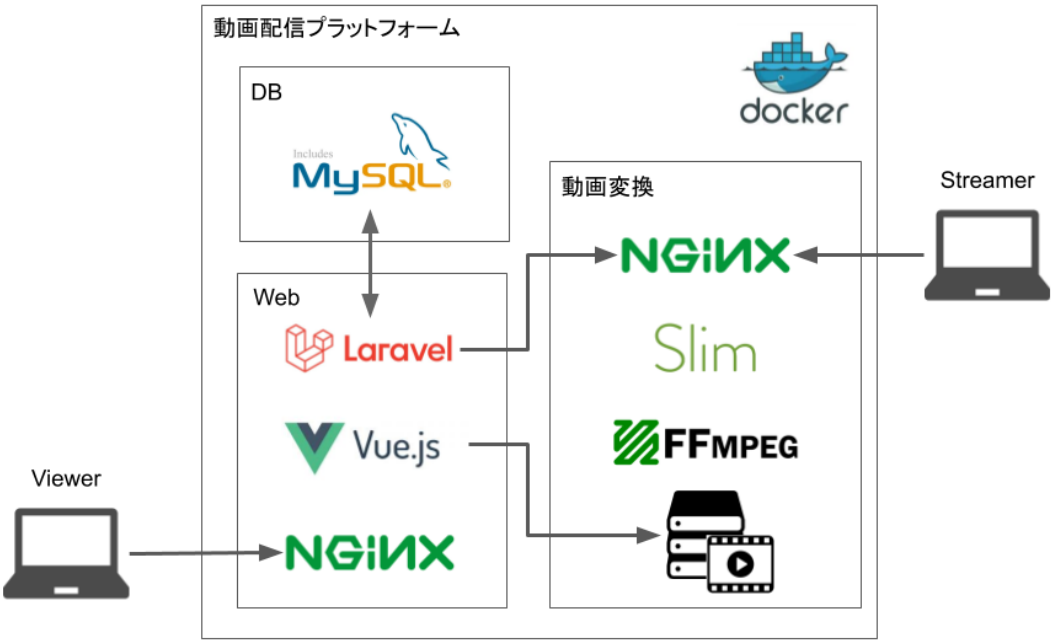

図 4.1　システム構成

本プロジェクトのシステムは，コンテナ仮想化技術の一つである Docker[5] を用いてアプリケーションを開発することにより，動作する OS を選ばずに，Windows, Mac, Linux

など複数の環境でデプロイすることが可能である．本動画配信プラットフォームをデプロイする場合は，Docker のインストールさえ行えば，コマンド一つで動画をデプロイすることが可能である [6].

本動画配信プラットフォームの利用者は，動画閲覧者と動画投稿者に分ける事ができる．動画閲覧者は，Web ブラウザから動画を視聴することが可能である．また，動画投稿者は，Web ブラウザから動画をアップロードすることで，動画投稿が可能である．また，OBS 等のライブストリーミングツールを用いて，ライブ配信を行うことも可能である．

本動画配信プラットフォームは，以下の 3 つのコンテナを連動させて動作させることで，動画サイトを提供する．

1. Web コンテナ
2. 動画変換コンテナ
3. DB コンテナ

Web コンテナでは，Web サーバに ningx[7]，バックエンドに PHP フレームワークである Laravel[8]，フロントエンドに Vue.js[9] を用いて動画サイトを構成する．動画変換コンテナは，nginx-rtmp-module[25] を用いて動画の配信や動画の変換を行う．DB コンテナは，MySQL[26] を用いて，動画サイトのユーザ情報や動画に関する情報の管理を行う．

4.2　使用技術について

本章では，動画配信プラットフォームで用いた，以下の技術について紹介する．

1. Docker
2. RTMP
3. HLS

4.2.1　Docker

Docker（ドッカー）はコンテナ仮想化を用いてアプリケーションを開発・配置・実行するためのオープンソースソフトウェアあるいはオープンプラットフォームである．Docker はコンテナ仮想化を用いた OS レベルの仮想化によりアプリケーションを開発・実行環境から隔離し，アプリケーションの素早い提供を可能にする．[28] つまり，Docker を用いてシステムを開発することで，Docker が対応している Windows,MacOS,Linux などの OS であれば，どの様な環境でもシステムを動作させることが可能であるということである．

本プロジェクトはオープンソースとして，ソースコードを公開をしており，システムの

開発者や利用者の環境が統一されていない．Docker を用いることで，複数の OS で開発やデプロイが可能であるという利点から，システムの開発に Docker を採用する．

4.2.2 RTMP

Real Time Messaging Protocol (RTMP) とは，Adobe Systems Incorporated[30] が開発している，Adobe Flash Player[31] とサーバーの間で，音声・動画・データをやりとりするストリーミングのプロトコルである [32].

特徴として，以下の 4 点が挙げられる．

1. リアルタイムにメッセージをやり取りするために開発されたプロトコル
2. Adobe Flash Player での利用を想定されている，映像や音声を通信させることができる
3. ユーザからのプッシュ配信が可能（配信者が自由なタイミングで配信を開始できる）
4. 映像は JPEG/H.263/VP6/AVC(H.264) のコーデック，音声は Linear PCM/ADPCM/MP3/AAC/speex のコーデックに対応

Youtube, ニコニコ生放送 [33], Twitch[34] などの有名なライブ配信サービスで動画配信を行う際のプロトコルとして採用されており，本プロジェクトでも，実績のある RTMP を使用することとした．

4.2.3 HLS

HLS (HTTP Live Streaming) とは，アップルが自社 iOS 向けに開発した，HTTP ベースのストリーミングプロトコルであり，インデックスファイルとセグメントファイルとに分かれて構成されている．インデックスファイルは m3u8 プレイリストと呼ばれ，セグメントファイルの場所や再生時間，再生順序などを定義したメタデータである．一方のセグメントファイルは ts ファイルと呼ばれており，MPEG2 Transport Stream 形式で細かく分割された複数の動画データファイル形式で配信される [35]．ABR(アダプティブビットレート:回線状況に応じて動的にビットレートを変える) 配信においては，画質ごとにビットレートを変えた動画を準備し，それを切り替えることで画質切り替えを実現する．[36] また，Safari では video エレメントで HLS 再生がデフォルトでサポートされている．

HLS を用いるメリットとしては，以下の点が挙げられる．

1. HTTP ベースなので Web サーバから配信が可能であり，ファイアウォール等のセキュリティ上の制限を受けずに配信が可能
2. Windows，Mac，iOS,Android などの複数の環境で再生可能

3. マルチビットレートでの配信が可能
4. CDN(Content Delivery Network) を使った構築により，高速で安定した配信環境を実現することが可能

また，デメリットは以下の通りである．

1. アダプティブビットレートを実現させるためには，複数のビットレートのファイルを準備するため，動画を保存するために必要な容量が増える
2. ライブ配信の場合，遅延時間が長い

本プロジェクトでは，HLS が Web サーバとの親和性が高い点や PC, スマートフォンなど多くの端末から再生可能であるため，採用することとした．

4.3　コメントフィルタリングについて

コメントフィルタリングには，禁止単語によるパターンマッチングと Google Cloud が提供する Natural Language API[10] を応用したフィルタリングの 2 種類を用いる．

禁止単語によるパターンマッチングでは，あらかじめ用意した禁止単語リストと，ユーザが任意に設定できる禁止単語リストの二つを合わせたものを用いる．禁止単語リストは，[37] のデータを元に，誹謗中傷コメントに含まれることが明らかな単語を抽出し，作成を行った．禁止単語リストを，付録 A に示す．

自然言語処理によるコメントの評価には，Natural Language API の機能の一つである，感情分析を応用することで，コメント表示の可否を判定する．本プロジェクトでは，利用者がコメントの制限レベルを設定することが可能であり，soft，medium，strict の三段階のコメントフィルタリングのレベルを決定できる．コメントフィルタリングレベルについての詳細は，4.3.1 に記載する．

コメントフィルタリング機能全体の処理フローを図 4.2 に示す．

4.3.1　Google CLoud Natural Language API を用いたコメントのフィルタリング

Natural Language API[10] とは，Google が提供する自然言語処理の機械学習モデルである．REST 形式の API としてサービスが提供されており，文章に対して，感情分析，エンティティ分析，構文解析など複数の分析が可能である．感情分析に関しては，中国語，英語,，フランス語，ドイツ語，イタリア語，日本語，韓国語，ポルトガル語（ブラジルとコンチネンタル），スペイン語の 9 か国語に対応している．

本プロジェクトのコメントフィルタリング機能では，感情分析を用いて，投稿されたコ

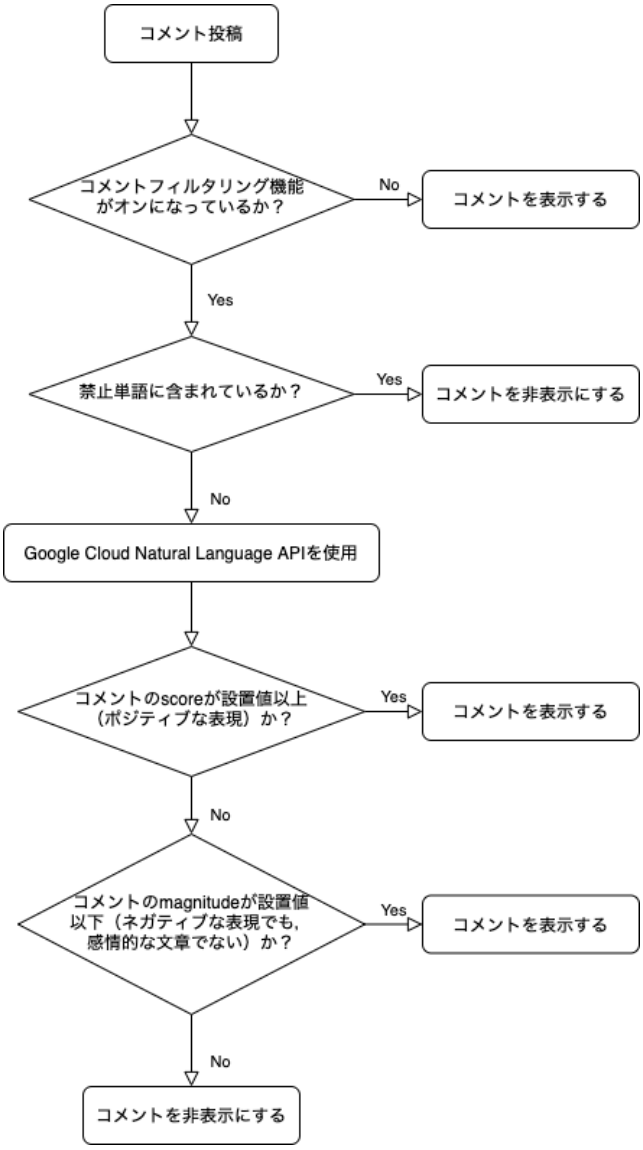

図 4.2　コメントフィルタリング機能の処理フロー

表 4.1　Google Cloud Natural Language API を用いて取得する値

項目	内容
score	-1.0（ネガティブ）〜1.0（ポジティブ）のスコアで感情が表される．
magnitude	指定したテキストの感情の強度（ポジティブとネガティブの両方）が 0.0〜+inf の値で示される。score と違って magnitude は正規化されていないため、テキスト内で感情が表現されるたびにテキストの magnitude の値が増加する．

メントの表示の可否判定を行う．以下の表 4.1 に，感情分析を用いて取得する値を示す．

この score と magnitude の 2 つの数値を元に，コメントフィルタリングを行うレベルに応じて閾値を決定し，コメント表示の可否を判定する．そこで，閾値を決定するために，表 4.2 のようにいくつかのコメント例に対して，Natural Language API を使用して，score と magnitude のサンプルを取得した．ここで得られた結果より，表 4.3 のようにそれぞれのフィルタリングレベルに対する閾値を決定する．これらの値を用いてコメント表示の可否判定を行うことで，設定値に応じたコメントフィルタリングが可能となる．

表 4.2　Google Cloud Natural Language API を用いて取得する値

文章（コメント）	score	magnitude
この動画は最高だな！	0.8	0.8
癖が強い	0.7	0.7
まぁまぁ面白かった	0.3	0.3
ちょっと微妙かな	0.1	0.1
360 億円あれば、うまい棒 36 億本食べれるのか... 一日、10 本食べても、3.6 億日？！うっめえ話だな！	0.1	1.2
普通ですね	-0.3	0.3
もし警備員が強盗に立ち向かったとしても装備の差で蜂の巣にされるのがオチ、薄給で良かったねぇ。	-0.5	0.5
こんな動画を出すくらいなら死んだ方がいい	-0.7	0.7
今日の動画最悪だな	-0.9	0.9
動画長すぎてつまらん	-0.9	0.9
クソ動画やな	-0.9	0.9
この動画作ったやつセンスなさすぎ死んだ方がいい	-0.9	0.9

表 4.3　コメントのフィルタリングレベルに対する score,magnitude の値

フィルタリングレベル	score	magnitude
soft	-0.3	0.3
medium	-0.6	0.5
strict	-0.8	0.8

第 5 章

結果

本プロジェクトで開発したシステムは，以下の機能を有する動画配信プラットフォーム（Panopticon）である．

1. 動画投稿機能
2. Live 動画配信機能
3. 動画再生機能
4. コメント機能

本章では，これらの機能と配信環境の構築方法について述べる．

5.1 配信環境の構築方法

本動画配信プラットフォームをデプロイするためには，以下のソフトウェアのインストールが必要である．

1. Docker 19.03.4 以上
2. docker-compose 1.24.0 以上
3. Make

ソフトウェアをインストール後，`https://github.com/ehimennlab/Panopticon` にて本動画配信プラットフォームをダウンロードし，以下のコマンドを入力すると，構築が完了する．

```
$ cd Panopticon
$ make serve
```

設定の詳細については，`https://github.com/ehimennlab/Panopticon` に記載する．

5.2　動画投稿機能

まず，動画投稿機能について述べる．動画を投稿するには，Web ブラウザとユーザアカウントが必要である．本動画配信プラットフォームをデプロイしたホストに対して，Web ブラウザを使用してアクセスする．（例：`http://example.com/login`）

ユーザアカウントが未作成の場合は，図 5.1 の新規作成フォームから新しくユーザを追加し，ログインを行う．

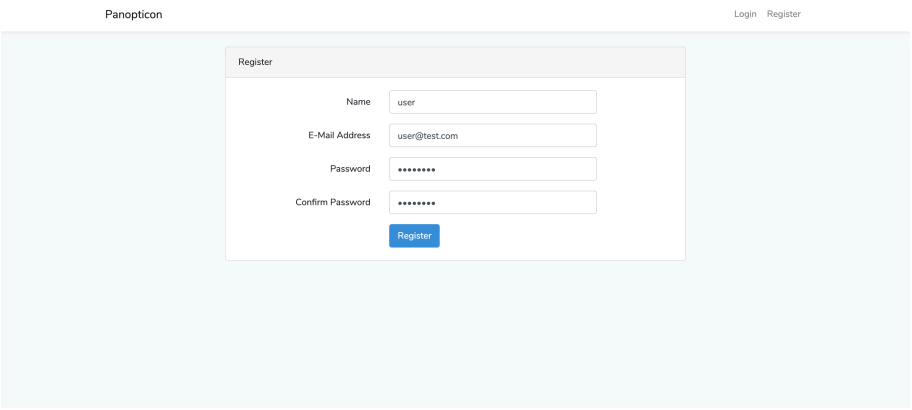

図 5.1　新規ユーザ登録画面

ユーザアカウントを所持している場合は，図 5.2 のログイン画面からログインする．

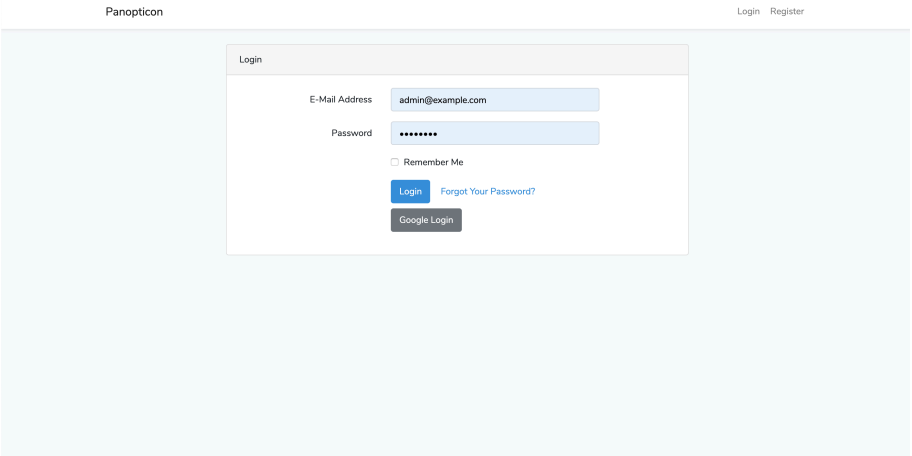

図 5.2　ログイン画面

その後，図 5.3 のように，ユーザページの動画投稿フォームから動画を投稿することで，動画を公開する事が可能である．

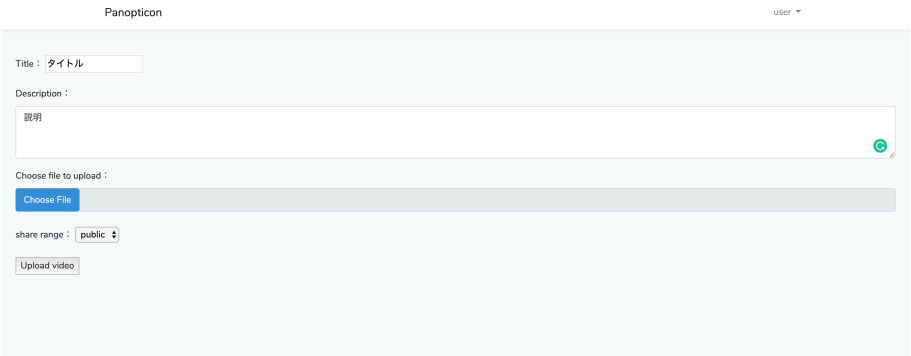

図 5.3　動画投稿画面

投稿した動画の一覧は，図 5.4，図 5.5 のように，ユーザページまたは，ホーム画面で確認できる．

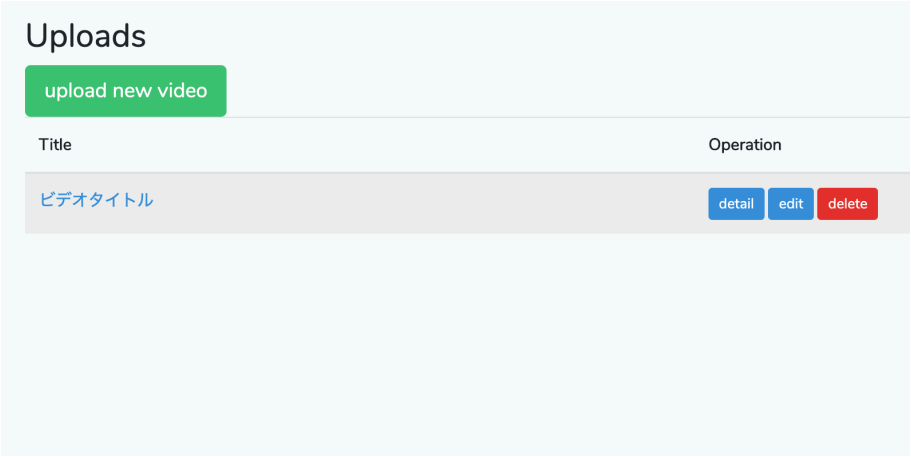

図 5.4　ユーザページの動画投稿一覧画面

5.3　Live 動画配信機能

Live 動画を配信するには，Web ブラウザ，ユーザアカウント，OBS 等の配信ソフトウェアの 3 つが必要である．本動画配信プラットフォームにログイン後，図 5.6 のように，ユーザページから Live 動画配信登録フォームに必要事項を入力し，送信する．

遷移後の画面で，図 5.7 のように，配信に必要な情報が得られる．

得られた Server 名とストリーミングキーの情報を図 5.8 のように OBS に入力し，配信を開始する事で，Live 動画配信を行う事ができる．

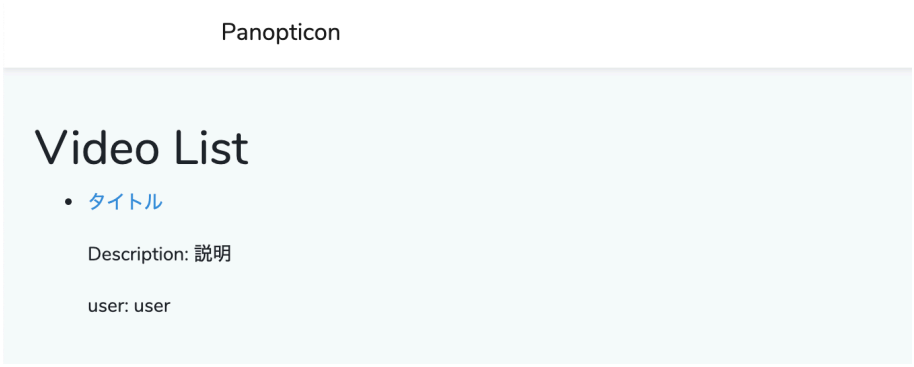

図 5.5　ホーム画面での動画投稿一覧

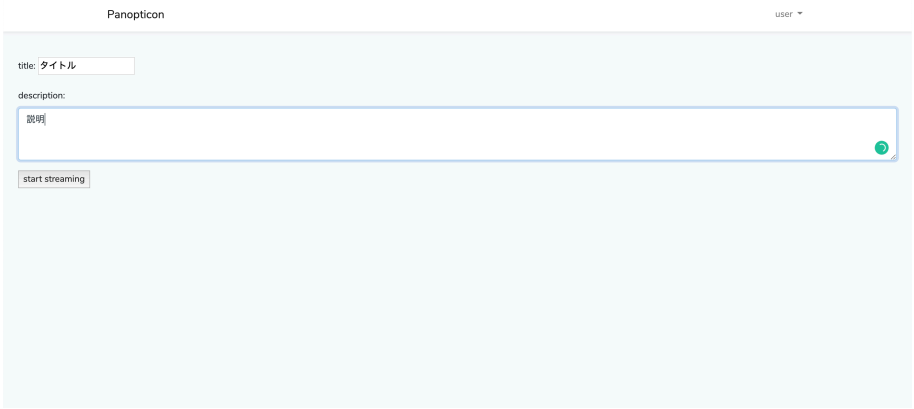

図 5.6　Live 情報登録画面

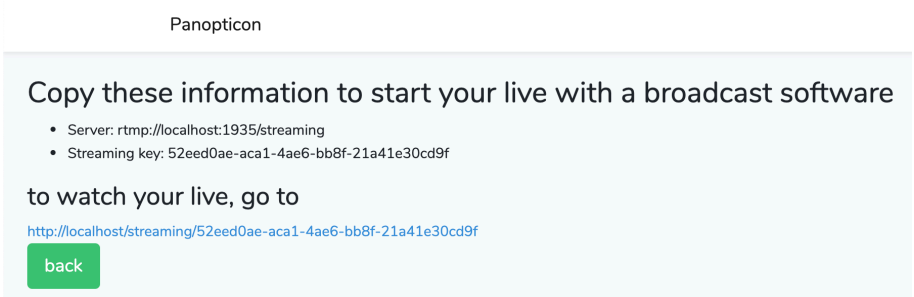

図 5.7　配信情報閲覧画面

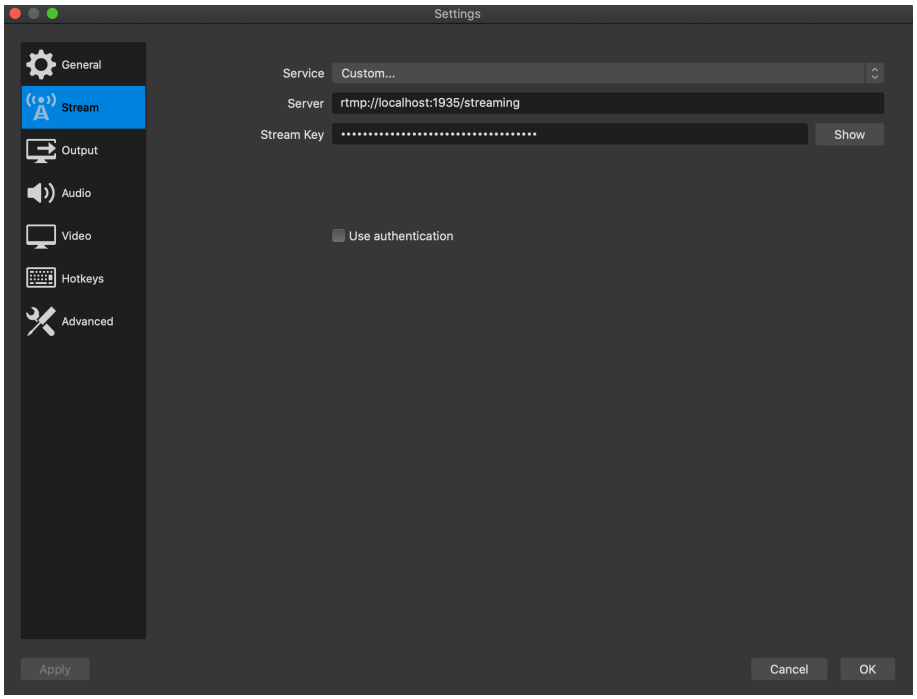

図 5.8　OBS の動画配信設定画面

5.4　動画再生機能

本動画配信プラットフォームは，Web ブラウザ上で動画を再生する．図 5.9 のように，ホーム画面などからアップロード動画または，Live 動画配信中の再生ページへのリンクをクリックし，動画再生ページに遷移する．

遷移後，図 5.10 のように動画の閲覧が可能である．動画プレイヤーは，以下の 5 つの機能を有する．

1. 動画の再生，一時停止
2. 音量の調整
3. 動画のスキップ
4. ピクチャーインピクチャー
5. 全画面化

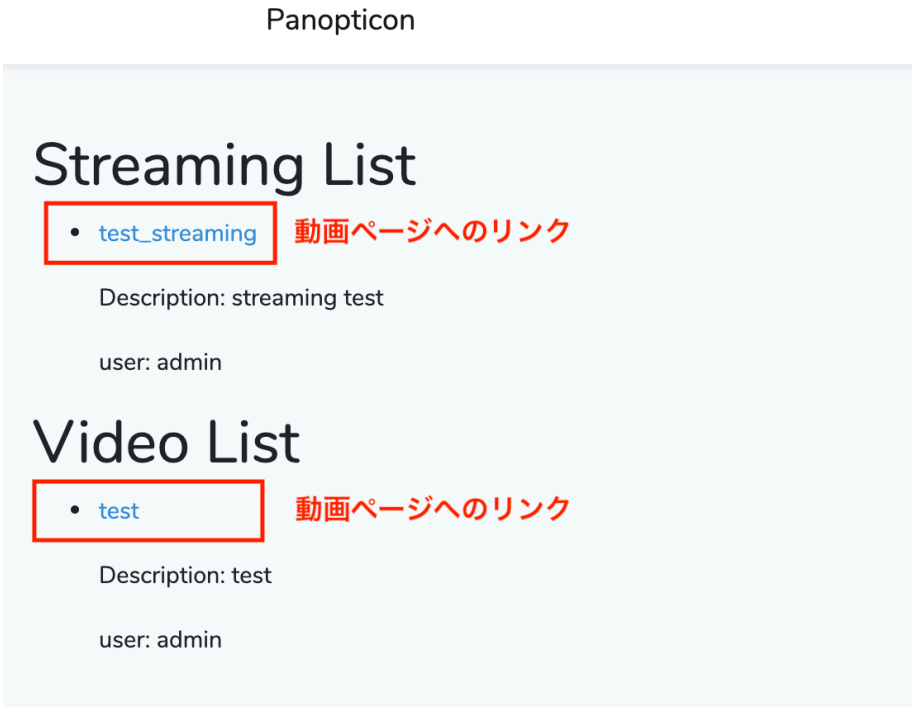

図 5.9　ホーム画面

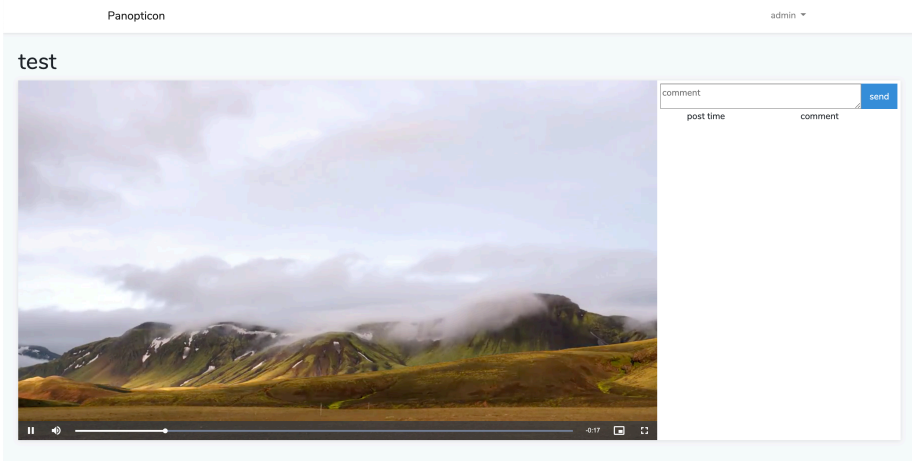

図 5.10　動画再生画面

5.5　コメント機能

本動画配信プラットフォームの動画再生画面にて，動画に対してコメントを投稿する事が可能である．コメントを投稿するには，動画再生画面にて，図 5.11 のように，プレイヤー右側のコメントボックスにコメントを入力し，送信をクリックする事で，コメントを投稿する事ができる．コメントは，投稿された日時が新しい順に上から表示される．

図 5.11　コメント画面

コメントのフィルタリング機能が ON になっている場合は，4.3 節の，コメントフィルタリング機能を使用して，誹謗中傷コメントの非表示処理を行う．投稿されたコメントの内容が，禁止単語リストに含まれる，または，コメント全体の評価スコアが，フィルタリングレベルで定めた基準値を下回る場合，投稿されたコメントは表示されない．

第 6 章

評価

本章では，本プロジェクトで構築した動画配信プラットフォーム（Panopticon）について，他動画配信サービスや OSS と比較する事で評価を行ったのち，コメントフィルタリング機能の評価を行う．

6.1　システムの評価について

本プロジェクトで構築した動画プラットフォームと他動画サービスや OSS との比較を行う事で，本動画配信プラットフォームの評価を行う．表 6.1 に本動画配信プラットフォームの特徴を示す．

本動画配信プラットフォームは，動画の再生，ライブ配信，コメントコメントフィルタリングが可能なシステムとなっており，OSS として現在公開中である．

動作環境に関して，表 3.1 と比較すると，本動画配信プラットフォームは，Docker，Docker Comopse さえ実行できる環境を用意するれば，コマンド一つでどのような OS 上でも動作させる事ができる．そのため，デプロイに関して，他の OSS と比較しても同等以上に簡便に配信環境を構築する事が可能である．よって，目標の 1 つ目である，OSS な

表 6.1　本動画配信プラットフォームの特徴

プロダクト名	動画再生	ライブ配信	コメント機能	コメントフィルタリング	動作環境
Panopticon	○	○	○	○	Docker 19.03.4 以上, docker-compose 1.24.0 以上

表 6.2　コメントとそれに対応するフィルタリングレベルの対応

コメント	score	magnitude	対応レベル
おもむろに韻をふむな（笑）吹いたわ（笑）	0.4	0.4	soft
Don’t rhyme suddenly haha I laughed	0	0	soft
Lol	0.3	0.3	soft
私はドナルドトランプが大統領を誇りに思う！みんなそうだろ？	0.1	0.8	soft
I am honored to call Trump my President. Who else is with me?!	0.1	0.3	soft
ゴミカス動画	0	0	soft
WTF	-0.2	0.2	soft
f**k you	-0.4	0.4	medium
この人嫌いです	-0.7	0.7	strict
I hate this guy	-0.8	0.8	strict

動画配信プラットフォームの開発に関しては，達成したと言える．

また，他の OSS と比較すると，コメントフィルタリング機能を有する動画配信 OSS はないため，その点において，他の動画プラットフォームと比較して優位性があると言える．

6.2　コメントフィルタリングについて

本動画配信プラットフォームには，コメントフィルタリング機能があり，投稿されたコメントに対し，管理者が設定する 3 段階のレベルに応じた規制が行われるようにすることで，表現の自由と規制のバランスを取ることが可能である．表 6.2 に，コメントの例と規制されるレベルの対応を示す．検証は，英語と日本語の二カ国語で行った．表中の対応レベルは，記載されるレベルによってフィルタリングが行われ，そのコメントを表示しないと判断することを示す．

Natural Language Cloud API は，英語にも対応しているため，英語でのコメントも解析可能である．Lol や WTF などのスラングにも対応している事が見受けられる．また，英語と日本語で，似たような意味の投稿も表現の違いから評価値が少し異なる事が見受けられる．しかし，検証した文章の中では，同じフィルタリングレベルでフィルタリングされるため，日本語と英語では，同じように動作すると推測される．

全体的に，人間の感覚に近い感情値が返却されるため，設定レベルに応じて正しくフィ

ルタリングされているように感じるが，f**k などの特定の単語に対して，低い感情値が返却されるため，特定の単語に対しては，禁止単語リストに登録するなどの対応が必要と考えられる．

実際に，この 3 段階のコメントフィルタリングが，適切な粒度や判断基準に基づいて，コメントのフィルタリングが行われているかどうかを判断するためには，ユーザによる検証が必要なため，今後の課題としたい．

第 7 章

結論

本プロジェクトでは，可能な限り簡単に動画配信環境の構築を行うことのできるソフトウェアを提案し，開発を行った．動画配信システムは，コンテナ仮想化ソフトである Docker[5] を用いて構築を行った．Docker のインストールが可能な環境であれば，どの様なプラットフォームでも動作させることが可能である．

本動画配信プラットフォームは，動画の投稿機能，ライブ配信機能とそれらを閲覧する再生プレイヤー機能や投稿されたコンテンツに対するコメント機能とコメントフィルタリング機能を有する．動画の再生には，ストリーミング方式を用い，動画のダウンロードと再生を同時に行うことが可能である．また，ライブ配信に関しては，OBS（Open Broadcaster Software）[27] 等を用いることで，10 20 秒程度の遅延での動画配信を行うことが可能である．

コメント機能では，動画を閲覧しているユーザ同士がリアルタイムにコメントを共有することを可能にした．

コメントフィルタリング機能では，動画の配信者が定めたレベルに応じて，三段階のコメントフィルタリングを行う．フィルタリングを行うために，コメントの文脈や単語から，コメントがポジティブやネガティブであるという度合いや感情の起伏の度合いを数値的に算出し，定めたレベルに応じて，投稿されたコメントの表示，非表示の切り替えを行う．

本プロジェクトで構築したソフトウェアを用いることで，コメントフィルタリング機能を持った動画サイトを公開することができ，管理者が表現の自由と規制のバランスを取りながら，動画プラットフォームを運営することが可能となった．

第 8 章

今後の課題

本プロジェクトで開発したソフトウェアにより，動画の配信環境を比較的簡単に構築できる様になった．しかし，課題としては，以下の項目が挙げられる．

1. ライブ配信の遅延が 10 秒〜20 秒程度と大きい
2. コメントフィルタリング機能の改善
3. 分散型のプラットフォームとしての動作

1 に関しては，動画のストリーミングプロトコルとして，RTMP を採用しているが，遅延が大きいため，より配信遅延の少ない WebRTC を使用した実装を検討している．

2 に関しては，現在，コメントのフィルタリングに Google Cloud が提供する Natural Language API[10] を使用してるが，今後は言語コーパス等からデータを収集し，Deep Learning を用いた独自の分類器を構築し，さらに精度の高いコメントのフィルタリングを行うことを検討している．また，コメントの分類だけでなく，動画やコメントの内容によって，配信や動画が楽しいものか，悲しいものかなどを分類し，ステータスとして表示できる機能などを追加していきたい．

3 に関しては，本プロジェクトで目標としていた，分散型の動画配信プラットフォームを構築し，各ノードごとを結合させ，一つの動画プラットフォームとして動作させる機能を実装することができなかった．今後，各ノードごとにアップロードされている動画やコメント，ユーザ情報等を共有する仕組みを構築することが課題である．

参考文献

[1] Youtube，https://www.youtube.com/，(参照 2020-05-01)

[2] niconico(ニコニコ)，https://www.nicovideo.jp/，(参照 2020-05-01)

[3] 動画の削除に関するトラブルシューティング，https://support.google.com/youtube/answer/6395024?hl=ja (参照 2020-04-28)

[4] 2ch の誹謗中傷の裁判一覧まとめ，http://xn--2ch-5q0fn79k.net/，(参照 2020-04-28)

[5] Docker，https://www.docker.com/，（参照 2020-04-28）

[6] ， Github「ehimennlab/Panopticon」，https://github.com/ehimennlab/Panopticon，（参照 2020-05-06）

[7] NGINX，https://www.nginx.com/，（参照 2020-05-06）

[8] Laravel，https://laravel.com/，（参照 2020-05-06）

[9] Vue.js，https://jp.vuejs.org/index.html，（参照 2020-05-06）

[10] Natural Language API, https://cloud.google.com/natural-language, （参照 2020-04-30）

[11] スマイリーキクチ中傷被害事件,https://ja.wikipedia.org/wiki/スマイリーキクチ中傷被害事件，(参照 2020-05-08)

[12] プレスルーム -YouTube- ， https://www.youtube.com/about/press/, (参照 2020-05-08)

[13] 暴力的で生々しいコンテンツに関するポリシー -YouTube ヘルプ-，https://support.google.com/youtube/answer/2802008?hl=ja, (参照 2020-05-08)

[14] コミュニティガイドラインの適用 -YouTube ヘルプ-, https://support.google.com/youtube/topic/9387060?hl=ja&ref_topic=2803138, (参照 2020-05-08)

[15] Twitter, https://twitter.com, (参照 2020-05-08)

[16] Mastodon, https://github.com/tootsuite/mastodon, (参照 2020-05-08)

[17] Google LLC，https://about.google/intl/ALL_jp/，（参照 2020-05-08）

[18] 株式会社ドワンゴ，https://dwango.co.jp/，（参照 2020-05-08）

[19] Netflix,https://www.netflix.com/,（参照 2020-05-08）
[20] 乾　孝司．言語表現の使用実態を踏まえたソーシャルメディア上の誹謗中傷行為の検出に関する研究，https://kaken.nii.ac.jp/ja/file/KAKENHI-PROJECT-15K20884/15K20884seika.pdf,（参照 2020-05-08）
[21] streama，https://github.com/streamaserver/streama,（参照 2020-05-08）
[22] AVideo，https://www.youphptube.com/,（参照 2020-05-08）
[23] PHPmotion，https://www.phpmotion.com/,（参照 2020-05-08）
[24] Joruri Video，https://joruri.org/product/jorurivideo/feature/,（参照 2020-05-08）
[25] Github「arut/nginx-rtmp-module」，https://github.com/arut/nginx-rtmp-module,（参照 2020-05-06）
[26] MySQL，https://www.mysql.com/jp/,（参照 2020-05-06）
[27] OBS，https://obsproject.com/ja,（参照 2020-04-28）
[28] Wikipedia 「Docker」,https://ja.wikipedia.org/wiki/Docker,（参照 2020-05-03）
[29] ライブ配信を支える技術と知識，https://engineer.dena.com/posts/2018.12/knowledge-for-livestreaming/,（参照 2020-05-03）
[30] Adobe Systems Incorporated，https://www.adobe.com/,（参照 2020-05-05）
[31] Adobe「Flash Player」，https://www.adobe.com/jp/products/flashplayer.html,（参照 2020-05-05）
[32] Wikipedia「RealTimeMessagingProtocol」，https://ja.wikipedia.org/wiki/Real_Time_Messaging_Protocol,（参照 2020-05-05）
[33] ニコニコ生放送，https://live.nicovideo.jp/,（参照 2020-05-05）
[34] Twitch，https://www.twitch.tv/,（参照 2020-05-05）
[35] ，HTTP Live Streaming, HLS のまとめ　概要・仕組み・課題など https://www.nice2meet.us/hls-http-live-streaming,（参照 2020-05-05）
[36] 動画サイト運営ノウハウブログ「HLS とは？：ストリーミング配信を実現する技術」，https://blog.socialcast.jp/05/post-729/,（参照 2020-05-05）
[37] ，西原陽子．電子掲示板からの文脈を考慮した誹謗中傷コメントの抽出，The 28th Annual Conference of the Japanese Society for Artificial Intelligence， 2014，p.3.（online），https://www.jstage.jst.go.jp/article/pjsai/JSAI2014/0/JSAI2014_1H4NFC01a3/_pdf,（参照 2020-05-06）

付録

付録 A　禁止単語リスト

ブス，ホモゲ，クソ，糞，自殺，アホ，アフォ，ドアホ，阿呆，バカ，馬鹿，ボケ，クズ，屑，カス，キチガイ，マジキチ，基地外，キモ，キモイ，ウザ，ウザイ，老害，パクリ，パク，フルボッコ，グロ，無能，DQN，ブサイク，不細工，ブサ，駄作，愚作，ババア，ババァ，ブチギレ，イライラ，NG，ショタ，ショボい，しょぼい，基地外，基地害，鬱，ブヒ，ブヒブヒ，アンチ，ワロタ，ワロス，ダサい，ダサイ，イラネ