Op:Code

Open Code for Hate-free Communication

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Overview of Online Hate Speech Detection Solutions

ActiveWatch digiQ INIMÕIGUSTE KESKUS HUMAN RIGHTS CENTRE ЦЕНТР ПО ПРАВАМ ЧЕЛОВЕКА "NEVER AGAIN" ASSOCIATION www.nigdywiecej.org MOVIMIENTO CONTRA LA INTOLERANCIA

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Introduction

The issue of online hate speech has been one of great interest in the past five years. While many approaches have been developed in recent years, the problem of automated detection of hate speech and offensive speech in online media is a complex one. First of all, the elements of complexity derive from multiple definitions produced by international bodies, different national legislations and different digital platforms. Secondly, hate-speech identification/classification sometimes proves to be a difficult task even for human agents (expert or non-expert) due to various cultural codes employed in different communication contexts (either national-linguistic or platform/community specific). Furthermore, as online communicators use various codes and combinations of codes (verbal – such as slang, nonverbal – such as images or emoticons, and paraverbal signifiers – such as capitalization, punctuation) text classification approaches that are mostly successful for other use cases yield poorer results for hate speech detection. Thirdly, tweets, social media and blog/news media comments tend to be informal and noisy sources which introduces another level of complexity – text processing for this particular case needs to keep into account various spellings (or misspellings), combinations of verbal, nonverbal and paraverbal signifiers (words, emoticons, punctuation/capitalization signifying tone/stance/emotionality). Linguistic diversity and features such as sarcasm or humor, cultural context, subversive tactics employed by users/commenters in online systems or difficulty arising from potential misclassification against users' freedom of expression are key factors to consider when designing automated online hate speech detection solutions.
This report provides an overview of key aspects of online hate speech detection and state-of-the-art tools and resources used in automated approaches to hate-speech detection.

Key concepts and definitions

A 2015 UNESCO study (Gagliardone et al. 2015) identifies definition, jurisdiction, comprehension and intervention as the key issues relevant to countering online hate speech. International Law Documents such as the Universal Declaration of Human Rights, International Covenant on Civil and Political Rights, European Convention of Human Rights, American Convention of Human Rights and the African Charter on Human and Peoples' Rights provide stipulations on freedom of expression, including freedom to seek, receive and impart information and ideas of all kinds, regardless of frontier or medium of expression. However, some limitations are imposed with respect to the issue of hate speech (either explicitly mentioned as such or not).

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

For example:

> *Any advocacy of national, racial or religious hatred that constitutes incitement to discrimination, hostility or violence shall be prohibited by law.*
>
> International Covenant on Civil and Political Rights, Article 20(2)

> *Any propaganda for war and any advocacy of national, racial, or religious hatred that constitute incitements to lawless violence or to any other similar action against any person or group of persons on any grounds including those of race, color, religion, language, or national origin shall be considered as offenses punishable by law*.
>
> American Convention of Human Rights, Article 13(5)

Definitions

Recent and growing international pressure on digital platforms to tackle the issue of hate speech has driven efforts to better define what platforms through their community standards define as hate speech.

In April 2018, Facebook publishes their definition with respect to 'protected characteristics':

> *We define hate speech as a direct attack on people based on what we call protected characteristics — race, ethnicity, national origin, religious affiliation, sexual orientation, caste, sex, gender, gender identity, and serious disease or disability. We also provide some protections for immigration status. We define attack as violent or dehumanizing speech, statements of inferiority, or calls for exclusion or segregation.*
>
> > *Tier 1 attacks, which target a person or group of people who share one of the above-listed characteristics or immigration status […]*
> > *Tier 2 attacks, which target a person or group of people who share any of the above-listed characteristics […]*
> > *Tier 3 attacks, which are calls to exclude or segregate a person or group of people based on the above-listed characteristics […]. We do allow criticism of immigration policies and arguments for restricting those policies.*

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

> Content that describes or negatively targets people with slurs, where slurs are defined as words commonly used as insulting labels for the above-listed characteristics.
>
> Facebook Community Standards

Facebook's definition was last updated in December 2019 and it explicitly includes violent speech in written or visual form, dehumanizing speech or imagery in the form of comparisons, generalizations, or unqualified behavioral statements to insects, animals, filth, bacteria, different types of criminals, statements of inferiority based on physical deficiency, mental deficiency, moral deficiency, expressions of disgust, cursing etc. This definition may serve as a good starting point for lexicon-based approaches as it is useful in defining categories of targets, but also categories of features of hate speech to be detected.

In June 2019, YouTube also announced some changes to their hate speech policies as follows:

> *Hate speech is not allowed on YouTube. We remove content promoting violence or hatred against individuals or groups based on any of the following attributes:*
>
> - *Age*
> - *Caste*
> - *Disability*
> - *Ethnicity*
> - *Gender Identity and Expression*
> - *Nationality*
> - *Race*
> - *Immigration Status*
> - *Religion*
> - *Sex/Gender*
> - *Sexual Orientation*
> - *Victims of a major violent event and their kin*
> - *Veteran Status*
>
> YouTube Help

The definitions employed by digital platforms such as Facebook or YouTube have the advantage of stemming from large numbers of real-world examples, covering different scenarios and being constantly updated. However, most of them rely on and are applicable to a human understanding of hate speech.

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Levels of analysis

In communication research, the key elements of hate speech analysis are the following:

- Content (what is being said?)
- Emitters (who is saying it?)
- Targets (who is it being said about?)
- Context (where is it being said?)

Although most approaches focus on content classification, recent scholarship shows a trend in bot digital social science and computer science approaches to use targets and context (Meza, Vincze, and Mogos 2018) or emitters and context (Pereira-Kohatsu et al. 2019) to improve on detection and classification approaches that are solely content-based.

Hate speech, offensive speech, dangerous speech

The issue of detecting hate speech is delicate due to the multiple definitions and their applicability in different contexts. Antagonistic or uncivil speech includes many subcategories and various degrees, not all of them included in the definitions of hate speech. Some expressions may be considered to be offensive, without being hate speech. Distinctions between categories such as hate speech, offensive speech and dangerous speech (used in the scientific or legal publications on the subject) may be subtle and contextual as it relies on judging the intent of the speaker, the reaction of the hearer/reader or the potential of the speech to lead to real-world acts of discrimination, exclusion or violence. Such attributes are very difficult to assess even by human coders in the case of online hate speech as the intent of unknown speakers (with little or no context provided) may be hard to make out, the reaction of the readers (whether or not they are likely to be offended by the language/expressions used) may depend on individual characteristics, and finally potential real world actions as effects may depend on the speakers' positioning, status or power in a context/community where the statements are made as well as the characteristics of the context/community.

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Overview of methods and approaches

Taxonomies of hate speech have emerged in the past few years with the goal of providing coding guidelines for human coders to provide examples in annotated datasets, but also breaking down the identification and classification task based on lexical features.

As mentioned in the introduction, the task of classifying abusive language is a difficult one due to several reasons as outlined in (Nobata et al. 2016):

- **It's not keyword spotting**: users obfuscate words/phrases that may trigger automated filters, use of just keywords can lead to many false positives.
- It's difficult to track racial and minority insults as **blacklists or filters are or should be continuously updated to reflect socio-cultural or political changes**, stereotypes or slang.
- **Abusive language may even be very fluent** and grammatically correct, not just noisy or misspelled.
- **Abusive statements may cross sentence boundaries**; although the meaning of individual sentences in a comment may not be offensive or hateful, its overall meaning may be.
- **Detecting sarcasm requires knowledge of the context/community**, its codes or even the individual emitters in an online context

Content classification

Hate speech detection approaches using natural language processing techniques relies on using different features as in most classification-related tasks. Schmidt and Wiegand provide an overview of most approaches divided into several categories (Schmidt and Wiegand 2017):

Surface features

Unigram and larger n-grams are most often used as predictive features. Some approaches consider character n-gram features as they seem to be more predictive than token n-grams (character level approaches deal better with capturing the similarity between spelling alternates). Other surface features considered are – URL mentions, punctuation, comment or token length, capitalization, non-alphanumeric characters, words not found in dictionaries.

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Word generalization

Approaches vary from Latent Dirichlet Allocation (LDA) methods that produce a topic distribution for each word to the word embeddings approach. Vector representations may indicate words that have similar meaning and be used as classification features. Paragraph or comment embedding methods have been shown to be more effective in the case of hate speech detection.

Sentiment Analysis

Hate speech and sentiment analysis are related problems and some solutions to hate speech detections include sentiment analysis / polarity classification as part of a multi-step process. High negative polarity may be used in conjunction with other feature-based classification or with target identification.

Lexicon-based approaches

Approaches based on lexical resources are very popular in hate speech analysis solutions. Especially for the English language, there are several dictionaries (slurs, insults, swear words) available on the web. Some approaches also involve using /applying weights to 'bad words'. However, research finds that lexicon-based approaches are insufficient as stand-alone features as some studies have shown that almost half of texts that contain 'bad words' are not in fact hate speech.

Linguistic features

Some approaches have used n-gram features combined with linguistic features. Linguistic features such as POS (part of speech) tags may be used as generic tools or specifically tailored to the problem. However, they are not shown to significantly improve hate speech detection. Syntactic dependency analysis or co-occurrence are however used to detect relations between tokens representing targets of hate speech and negative/offensive/hateful attributes associated.

Knowledge-based approaches

Very few approaches employ automatic reasoning based on existing ontologies (augmented with concepts and relationships to suit the particular task of detecting hate speech). The main reason is that existing ontologies should be augmented/adapted to suit the needs of different types of hate speech scenarios (different targets, different/complex negative stereotypes in different communities/cultures).

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Targets Identification

Target-based approaches use lexicons of words designating groups or categories of persons usually coupled with detection of violent, offensive or dehumanizing language. Target lexicons may be based on previous research on vulnerable groups in a specific national-linguistic context, NER tagging for groups/persons, or based on word embedding approaches.

For example, in (Salminen et al. 2018) a hate target taxonomy is defined in conjunction with hate language to yield a total of 29 main and sub-categories and an additional neutral category (in order not to generate pro-annotation bias). The categories described by Salminen et al. included below are defined in the context of comments to online media.

Language Category	**Description**
Accusation	Accusing someone of something, without relevant evidence to support it. Accusations of lies, treason, all types of felonies, etc.
Promoting Violence	Calling people to deal with something using violence, asking for murders; threatening human life.
Humiliation	Using words like: idiot, retard, stupid, dumb, trying to degrade someone.
Swearing	Filthy language, bad words, swearing, non-polit

Main Target Category	**Description**
Financial Power (Subcategories: Corporation, Wealthy)	*Hatred toward wealthy people and companies and their privileges. Pointing out their intentions to manipulate and commit crimes*
Political Issues (Subcategories: Terrorism, Politics, Ideology)	*Hate toward government, political parties and movements, war, terrorism, the flaws of the system.*
Racism & Xenophobia (Subcategories: Anti-white, Anti-black, Xenophobia)	*Racists comments toward black, white, asian. Generalizations about some characteristics, and hateful comments regarding refugees.*
Religion (Subcategories: Anti-Islam, Anti-Semitist)	*Everything about religion, including Judaism, Christianity, Islam, and religion in general. Both as a subject of hatred, or object.*
Specific Nation(s)	*Hate towards different countries, their systems, people (if the nationalities are mentioned), and certain events, like immigration, territory, and sovereignty.*
Specific Person	*Hate toward specific people who can be regular people, politicians, millionaires, celebrities, or some other related to specific news.*
Media (Subcategories: Towards media company, other)	*Comments and emotional outbursts about bias and false statements made on purpose by the corrupted media*
Armed Forces (Subcategories: Police, Military)	*Hate toward military, law enforcement, and the way they operate, which includes unethical behavior.*
Behavior (Subcategories: Humanity, other)	*Hate toward the world, humanity, immoral actions of some part of the society, ignorant people, people that committed certain actions, and that have certain habits.*

Tables adapted from (Salminen et al. 2018)

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Context and Meta-Information

Most hate speech detection tasks come from social media platforms or from blogs/media comments sections. Information about the context (page, article, topic, thread) or about the emitter (user names/nicknames and their posting history) has been used to improve hate detection by considering controversial topics/threads or users' posting history (number of previous posts/comments containing hate speech or number of replies). However, in some cases such information is not available or may not be used due to legal/privacy reasons.

Image and Multimodal analysis

Although most research into hate speech only considers text analysis, comments and posts on social media also contain emoticons and images. There are several recent works that deal with analyzing images (usually satirical image macro formats widely shared on social media) or incorporating the analysis of emoticons inserted into text messages.

Emoticons

Although in most cases symbols or non-word entities are excluded from natural language processing methods in the data cleanup phase, the issue of automatic hate speech detection may make use of the meanings conveyed through their use. Recent models have attempted to include emoticons in the analysis by converting them from iconic signifiers to symbolic signifiers. According to (Orasan 2018) emoticons or emojis can be converted into their corresponding word strings by using the emoji library (https://pypi.org/project/emoji/). Furthermore, the author suggests that they may also be grouped or tagged according to meaning by using the EmojiNet (http://emojinet.knoesis.org/home.php) as a resource.

Analyzing Images

The automated analysis of images for the purposes of hate speech identification/classification tasks may be approached in several ways by combining two or three inputs (the image, the image caption and text detected in the image by use of OCR) in CNN (Convolutional Neural Networks) + RNN (Recurrent Neural Networks) models, according to (Gomez et al. 2019). The authors compare several architectural models that include both text and image features and find that the **image features do not significantly improve the prediction** as compared to only using the text in the captions or tweets/posts/comments. Hate speech manifested in multimodal messages is based on complex relations between elements and cultural codes and references which makes identification/classification a very complex task.

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Overview of tools and resources

The last section presents a quick overview of common tools and resources used by researchers and developers in the past five years to develop hate speech detection and classification solutions.

Developer Tools & Libraries

Libraries	Description	Category
OSMod - The ConversationAI Moderator App	A machine-assisted human-moderation toolkit that uses Perspective AI from Google to predict the toxicity level of the comments.	Conversation AI Moderator
Scrapy	Scrapy is a free and open- source web-crawling framework written in Python. Originally designed for web scraping, it can also be used to extract data using APIs or as a general-purpose web crawler.	Scrapping
Apache Nutch	Apache Nutch is a highly extensible and scalable open source web crawler software project.	Web crawler
Ludwig	Ludwig is a toolbox that allows to train and test deep learning models without the need to write code.	NLP, CV, ML
kraken	An OCR system with script detection and multiscript recognition support, with built in word bounding boxes and character cuts to be used for extracting text out of images.	HTR/OCR system
Google Teachable	Teachable Machine is a web-based tool that makes creating machine learning models fast and easy. Image classification models can be exported and integrated into your own applications and platforms.	ML, image classification
Snorkel	'Snorkel is a system for programmatically building and managing training datasets without manual labeling. In Snorkel, users can develop large training datasets in hours or days rather than hand-labeling them over weeks or months.' https://www.snorkel.org/ https://github.com/snorkel-team/snorkel	Labelling

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

spaCy	**'spaCy** is a free open-source library for Natural Language Processing in Python. It features NER, POS tagging, dependency parsing, word vectors and more.' https://spacy.io/	NLP, ML
BERT	**'BERT**, or **B**idirectional **E**ncoder **R**epresentations from **T**ransformers, is a new method of pre-training language representations which obtains state-of-the-art results on a wide array of Natural Language Processing (NLP) tasks.' https://github.com/google-research/bert	NLP, ML
Transformers	**'Transformers** (formerly known as pytorch-transformers and pytorch-pretrained-bert) provides state-of-the-art general-purpose architectures (BERT, GPT-2, RoBERTa, XLM, DistilBert, XLNet, CTRL...) for Natural Language Understanding (NLU) and Natural Language Generation (NLG) with over 32+ pretrained models in 100+ languages and deep interoperability between TensorFlow 2.0 and PyTorch' https://github.com/huggingface/transformers	NLP, ML
Fast-Bert	'Fast-Bert is the deep learning library that allows developers and data scientists to train and deploy BERT and XLNet based models for natural language processing tasks beginning with Text Classification.' https://github.com/kaushaltrivedi/fast-bert	NLP,ML
Gensim	'Gensim is a Python library for *topic modelling*, *document indexing* and *similarity retrieval* with large corpora. Target audience is the *natural language processing* (NLP) and *information retrieval* (IR) community.' https://pypi.org/project/gensim/	NLP, ML
LASER	'LASER Language-Agnostic SEntence Representations - LASER is a library to calculate and use multilingual sentence embeddings.' https://github.com/facebookresearch/LASER	NLP, ML
Keras	'keras-text is a one-stop text classification library implementing various state of the art models with a clean and extendable interface to implement custom architectures.' https://raghakot.github.io/keras-text/	NLP, ML
FastText	'FastText is an open-source, free, lightweight library that allows users to learn text representations and text classifiers. Also includes Python module and API' (https://fasttext.cc/) See (Bojanowski et al. 2017) for enriching word vectors with subword information.	NLP, ML

Tools – Sentiment and Lexicons

Tools	Description	Category
SentiStrength	'SentiStrength estimates the strength of positive and negative sentiment in short texts, even for informal language. It has human-level accuracy for short social web texts in English, except political texts. SentiStrength reports two sentiment strengths:-1 (not	Sentiment Analysis

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

	negative) to -5 (extremely negative) AND 1 (not positive) to 5 (extremely positive) (free only for academic use)' http://sentistrength.wlv.ac.uk/	
Stanford NLP Sentiment Treebank	'Deep learning model which builds up a representation of whole sentences based on the sentence structure. It computes the sentiment based on how words compose the meaning of longer phrases.' https://nlp.stanford.edu/sentiment/	Sentiment Analysis
SentiWordNet	'SentiWordNet is a lexical resource for opinion mining. SentiWordNet assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity.' https://github.com/aesuli/sentiwordnet	Sentiment Analysis
Google Perspective / Conversation AI	'Perspective is an API that uses machine learning models to score the perceived impact a comment might have on a conversation. For support, see:' https://support.perspectiveapi.com/ https://conversationai.github.io	API
Hatebase	'Researchers are encouraged to take advantage of Hatebase's vocabulary dataset, which is a valuable lexicon for searching other data repositories such as public forums, as well as Hatebase's sightings dataset, which is useful for trending analysis' https://hatebase.org/academia	Lexicon
Hurtlex	'HurtLex is a lexicon of offensive, aggressive, and hateful words in over 50 languages. The words are divided into 17 categories, plus a macro-category indicating whether there is stereotype involved' https://github.com/valeriobasile/hurtlex	Lexicon
Online Abuse towards UK politicians	,404 abuse terms used in "Twits, Twats and Twaddle: Trends in Online Abuse towards UK Politicians", ICWSM 2018, and in "Online abuse of uk mps in 2015 and 2017: Perpetrators, targets, and topics"' http://staffwww.dcs.shef.ac.uk/people/G.Gorrell/publications-materials/abuse-terms.txt	Lexicon
Lexicon of abusive words	'This repository contains all new resources we created for our NAACL 2018 paper "Inducing a Lexicon of Abusive Words -- A Feature-Based Approach" by Michael Wiegand, Josef Ruppenhofer, Anna Schmidt and Clayton Greenberg. It also includes further details regarding our experimental set-up for which no space was available in the actual paper'. https://github.com/uds-lsv/lexicon-of-abusive-words	Lexicon

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Reddit Hate Lexicon	A lexicon based on Reddit hate speech – see You can't stay here: the efficacy of Reddit's 2015 ban examined through hate speech, download https://www.dropbox.com/sh/5ud4fwxvb6q7k20/AAAH_SN8i5cfmJRKJteEW2b2a?dl=0	Lexicon
Racial Slurs Database	Database of different designators and slurs used to refer to racial/national/ethnic groups http://www.rsdb.org/	Lexicon
Swear Word List	Dictionary of different swear words and curse words used in word filters https://www.noswearing.com/dictionary	Lexicon

Datasets

Name	Annotations	Size	Source	Link
Davidson et al.	Hate speech and offense	25000	Twitter	https://github.com/t-davidson/hate-speech-and-offensive-language
Wikipedia Detox	Personal attacks /insults	100000	Wikipedia	https://figshare.com/articles/Wikipedia_Detox_Data/4054689
Waseem	Hate speech, racism, sexism	16000	Twitter	https://github.com/zeerakw/hatespeech
Kaggle / Impermium	Insulting / Not insulting	2236	Forum	https://www.kaggle.com/c/detecting-insults-in-social-commentary/overview
OffensEval 2019	Offensive speech, targets	14000	Twitter	https://competitions.codalab.org/competitions/20011
Stormfront	Hate speech	10568	Forum	https://github.com/aitor-garcia-p/hate-speech-dataset
hatEval	Hate speech against immigrants	13000	Twitter	https://competitions.codalab.org/competitions/19935
Founta et al.	Hate speech, abuse	100000	Twitter	https://github.com/ENCASEH2020/hatespeech-twitter
MMHS150K Dataset	Hate speech in images	150000	Twitter	https://gombru.github.io/2019/10/09/MMHS/
Hate Meme Detection	Hate speech detection in Internet memes	5020	Google Images	https://github.com/imatge-upc/hate-speech-detection (utility script that downloads hate class memes)

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

MEMES dataset	30,000+ OCR'd political memes and their captions	30000	Political Memes and their captions	https://www.kaggle.com/ahmethamzaemra/memes-dataset/data https://www.kaggle.com/ahmethamzaemra/memes-dataset
Hatebase	Service to analyze hate speech	-	3,639 terms, 97 languages	https://hatebase.org/ Free for non-profit organizations
Meme Generator Data Set	Memes	86310	Memes harvested from Meme Generator.	https://www.kaggle.com/electron0zero/memegenerator-dataset/home

Features and models

In (Pereira-Kohatsu et al. 2019) the authors provide an overview of features and models used by researchers in the past years.

Source	Features	Model
(Djuric et al. 2015)	BOW, TF, TF-IDF, paragraph2vec embeddings	LR
(Zia et al. 2016)	unigrams, TF-IDF, retweets, favourites, page authenticity	**SVM**, NB, kNN
(Silva et al. 2016)	**sentence structure**	Rule based
(Waseem and Hovy 2016)	**Author gender**, length of tweets, length of user description, location, **char n-grams**, word n-grams	**LR**
(Waseem 2016)	**char n-grams**, **word n-grams**, **skip-grams**, **tweet length**, author gender, **clusters**, **POS**, Author Historical Salient Terms (AHST)	**LR**
(Badjatiya et al. 2017)	char n-grams, TF-IDF, BoWV, **random embeddings**, GloVe embeddings	LR, RF, SVM, **GBDT**, DNN, CNN, **LTSM**
(Davidson et al. 2017)	n-grams, TF-IDF, POS, readability, sentiment, hashtags, mentions, retweets, URLs, length	**LR**, NB, DT, RF, SVM
(Gambäck and Sikdar 2017)	**word2vec embeddings**, random embeddings, char n-grams	**CNN**
(Park and Fung 2017)	**char embeddings**, **word embedding**	CharCNN, WordCNN, and **HybridCNN**
(Del Vigna12 et al. 2017)	POS, sentiment analysis, **word2vec embeddings**, CBOW, n-grams, text	SVM, **LSTM**

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

	features, **word polarity**	
(Salminen et al. 2018)	**n-grams**, **semantic and syntactic**, **TF-IDF**, **word2vec embeddings**, **doc2vec embeddings**	LR, DT, RF, Adabost, **SVM**
(Zhang, Robinson, and Tepper 2018)	n-grams, POS, TF-IDF, mentions, hastags, length, readability, sentiment, mispellings, emojis, punctuation, capitalisation, **word embeddings**	SVM, **CNN + GRU**

Platform Architecture - Proposals

The aim is to build a platform that can be used to monitor and score conversations in terms of hate level and toxicity with the final scope of being able to help moderators to give real time feedback to commenters and / or to allow online readers to consume only relevant information. The platform should be able to monitor comments on given online sites or platforms. For each and every new comment received or detected, it should predict its toxicity level and if this level is above a defined threshold (easiest rule), it should perform actions like notifying the moderator(s) and / or the publisher, hiding the comment (if possible and required), update a dashboard with information, etc.
In this context, we see two main groups of tasks that need to be performed to achieve this project goals:

1. Train the classification engines that will be used when making the prediction on the toxicity level. These engines should target texts, as well as images.
2. Develop or customize an orchestration system that is able to implement the proposed workflow.

Train the ML classification engines

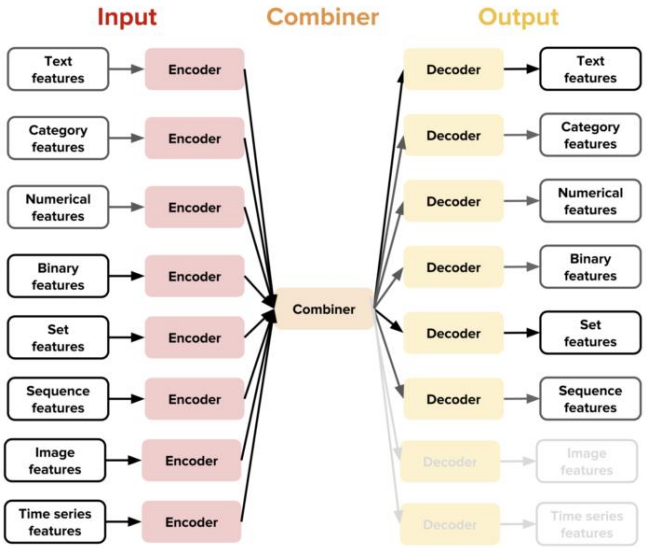
Training the classification engines to make predictions regarding the toxicity level of the comments or images (memes, for example) consist in preparing a training & validation dataset and selecting a framework to be used to train the engine using the training & validation dataset. Preparing the training & validation dataset can be done in at least two ways:

1. Scraping your data from the Web and then annotating this data to obtain the training & validation dataset for your classifiers. For this case, we encourage you to use Snorkel to label and manage your training datasets.
2. Finding and adapting an existing dataset. See the list of available datasets in the previous section of this document.

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

There are a plethora of ways to train your classifiers, using different open source available frameworks and libraries. From all of these, we recommend the usage of Ludwig from Uber AI Lab.

Ludwig was released in February 2019 by Uber and represents the most simple way to experiment with building machine learning models. It is a toolbox that is built on top of TensorFlow that allows you to create, train, experiment and use various ML models **without writing any line of code**. Finding the right model architecture and hyperparameters for your model is a difficult aspect of the deep learning pipeline. Normally, you could spend hours experimenting with different parameters and deep learning network architectures to find the model that would fit your specific problem. With Ludwig you do all of these in declarative mode. Ludwig documentation is really good and you will be able to start experimenting with your models really quickly. Ludwig allows you to train a deep learning model by only providing a file containing the data like a csv and a YAML configuration file in which we need to specify some information about the features contained in our data file like if they are dependent or independent variables. If more than one dependent/output variable is specified, Ludwig will learn to predict all of the output simultaneously (Gilbert Tanner). The main idea behind Ludwig is depicted (Gilbert Tanner) like follows:



These specific types of encoders and decoders can be set in the configuration file and provides you with a highly modularized and extensible architecture that has specific preprocessing steps for each type of data (Gilbert Tanner).

For the purpose of this project, you will need to build at least one Text Classification engine. You can find a simple example on how to build a text classifier with Ludwig here: https://uber.github.io/ludwig/examples/#text-classification.

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Also, Ludwig can help you to train an Image Classification engine, as well. You need to have an annotated images dataset. With this dataset, you can experiment with Ludwig by following this example: https://uber.github.io/ludwig/examples/#image-classification. For image classification we suggest using an existing memes dataset (see the datasets section above) that you could eventually clean up using Snorkel. To obtain a more complete images dataset, you could use the Meme Generator Data Set, combining images with hate speech phrases, making sure your dataset is balanced and you have a good variability of your training data to avoid overfitting.
Ludwig provides an easy way to use your engines and integrate them into a bigger platform by serving your predictions using Ludwig's serve command. This command lets you load a pre-trained model and serve it on an http server. CURL examples from Ludwig's web site:

- File: $ curl http://0.0.0.0:8000/predict -X POST -F 'image_path=@path_to_image/example.png'
- Text: $ curl http://0.0.0.0:8000/predict -X POST -F 'english_text=words to be translated'
- Both Text and File: $ curl http://0.0.0.0:8000/predict -X POST -F 'text=mixed together with' -F 'image=@path_to_image/example.png'

The host and the port where Ludwig is listening for incoming requests can be specified as optional arguments, -p PORT and -h HOST.
Another way to quickly and visually train an image classification model is to use Google Teachable Machine. With Google Teachable Machine you can build an image classification in terms of minutes if you start from an annotated dataset. The training happens inside the browser (you need a machine with really good resources for larger image datasets) and the result can be exported as a TensorFlow model. To understand more about how this tool was built, check its GitHub repository here: https://github.com/googlecreativelab/teachablemachine-community.

Augmenting the Image Classification with Text

If you want to augment your image classification with the text extracted out of the images, our suggestion would be to use kraken. kraken is an OCR system with script detection and multiscript recognition support, with built in word bounding boxes and character cuts to be used for extracting text out of images.
Public available kraken models are available. For extracting english written text from memes or other images containing text, you could use the default english model from here: https://github.com/mittagessen/kraken-models/tree/master/pyrnn/default.
Otherwise, training a specific kraken model is done following the steps described here: http://kraken.re/ketos.html. In short, the process of training such a model is as follows:

1. Prepare the training data by segmenting the images and generating HTML files (transcription environments) where you would do manual transcription.

$ ketos transcribe -o output.html image_1.png image_2.png …

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

You could also use the existing default model to prefill the transcription environments:
$ ketos transcribe -p ~/english.mlmodel -p output.html image_1.png image_2.png ...

2. Manual transcribe as many images as you can. Transcription has to be diplomatic, i.e. contain the exact character sequence in the line image, including original orthography.
3. the contents of the filled transcription environments have to be extracted through the ketos extract command:

$ ketos extract --output output_directory *.html
The result will be a directory filled with line image text pairs NNNNNN.png and NNNNNN.gt.txt and a manifest.txt containing a list of all extracted lines.

4. Training data is just a directory containing image-text file pairs as produced at step 3. The minimal example to train a new model is:

$ ketos train training_data/*.png

5. You could also fine tune an existing model with new training data by resuming the training of an already existing model:

$ ketos train -i model_best.mlmodel more_memes/*.png
With your text and / or image classification engines ready, you only need to develop or customize an orchestration system that is able to implement the proposed workflow. The two architecture proposals that follow is our suggestion on the way to go for the orchestration system.

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Architecture Proposal (AP1): Your Orchestration System (OSMod) is Served By Publisher

OSMod - The ConversationAI Moderator App is a machine-assisted human-moderation toolkit that uses Perspective AI from Google to predict the toxicity level of the comments. By building your moderation platform on top of OSMod you get all the orchestration functionality out of the box and you'll only have to hook your own trained engines into the system to replace Perspective AI from the picture.

OSMod is developed using the TypeScript language. TypeScript is a superset of JavaScript that compiles to plain JavaScript. Typescript is an open-source programming language developed and maintained by Microsoft. It's a strict superset of JavaScript, adding optional static typing to it. It was first launched on October 1st, 2012 and now is at version 3.8.

"TypeScript in 5 minutes" section of TypeScript documentation is the best way to start learning / understanding TypeScript.

Installation instructions for OSMod can be found here: https://github.com/conversationai/conversationai-moderator/blob/master/README.md. There are multiple ways to run the orchestration system, including a way that involves running it into a docker container.

After installation, a running OSMod orchestration system should be comprised from the following components:

1. A MySQL database that holds all of the applications state. The data model documentation can be found here: https://github.com/conversationai/conversationai-moderator/blob/master/docs/modeling.md.
2. The Frontend-Web Server service hosting the static ReactJS site. This sends messages to the Backend API service.
3. The Backend API service is responsible for querying the SQL database and sending data to the front-end service. This is also the endpoint that receives requests from the commenting platform it is supporting moderation of; and it sends requests back to the commenting platform with user actions (e.g. to hide, reject or approve comments).
4. Backend Work Queue service responsible for managing concurrent queue of asynchronous work.
5. A number of **assistant services** responsible for automating tasks. Out of the box, this is just the Perspective API. **This is the place where you should hook your own ML services.**

The GitHub repository describes in very concise manner all the components of the system - this documentation can be found here: https://github.com/conversationai/conversationai-moderator/tree/master/docs.

To understand how OSMod works, you need to start from understanding how comments go through the orchestration system. Here is the documentation for that:

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

https://github.com/conversationai/conversationai-moderator/blob/master/docs/comment_flow.md.
In summary, the comment workflow looks like the following:

1. A publisher submits a comment or an article to OSMod through a specific API endpoint.
2. The comment is sent to the assistants for prediction - here is the hook for your ML models / services.
3. The assistant comes back through a callback URL with a prediction / resolution.
4. A ModerationRule is created to try to resolve the comment based on a predefined rule.
5. If the comment is resolved by the rule, then a task is created to notify the publisher.
6. If the comment is not resolved by the rule, then it is made available to the OSMod frontend so that moderators can approve or reject comments either singularly or in bulk, notifying the publisher about the resolution.

Architecture Proposal (AP2): You Feed Your Own Orchestration System

The architecture proposal above (AP1) assumes that you have the publisher serving you with comments or other online content to be moderated. If you'd like to build a moderation platform that is not triggered by a publisher, then you will need a mechanism that will monitor a specific online content source (a newspaper online, a twitter feed, etc.) and trigger the workflow, as per your needs. On the other hand, at the end of the workflow you will eventually need a way to expose the result of your moderation into a dashboard and to send the feedback to the original online content source, if this is possible.
Our suggestion for these cases is to use architecture proposal (AP1) and to add a way to trigger the workflow by monitoring the online source(s) of interest.

- Scrapy is the way to go if you want to regularly check comments on specific Web pages and trigger the moderation workflow, when needed. If you need more Web Crawling complexity, you could even integrate Apache Nutch into the whole picture.
- Twitter API can be used to get public tweets to be monitored: https://developer.twitter.com/en/docs/labs/tweets-and-users/quick-start/get-tweets
- Facebook API can be used to get public posts to be monitored: https://developers.facebook.com/docs/graph-api/reference/post/

NOTE: Check https://github.com/conversationai/conversationai-moderator/tree/a1ddb5e33e51ac0b0d25444134b0079598a66de8/packages/backend-api/src/integrations for an example of intregrating YouTube with OSMod for YouTube channels moderation.

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Glossary

NLP Natural Language Processing
ML Machine Learning
CV Computer Vision
DNN deep neural network
POS Part-of-Speech tagging
SVM Support Vector Machines
CNN Convolutional Neural Network
RF Random Forests
DT Decision Trees
FNN Feedforward Networks
RNN Recurrent Neural Networks
GDBT Gradient Boosted Decision Trees
LSTM Long Short-Term Memory
GRU GRU Gated Recurrent Unit Networks
LR Logistic Regression
kNN K-nearest Neighbor
NB Naïve Bayse
OCR Optical Character Recognition
HTR Handwritten Text Recognition

References

Badjatiya, Pinkesh, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. "Deep Learning for Hate Speech Detection in Tweets." In *Proceedings of the 26th International Conference on World Wide Web Companion*, 759–760.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. "Enriching Word Vectors with Subword Information." *Transactions of the Association for Computational Linguistics* 5: 135–146.

Davidson, Thomas, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. "Automated Hate Speech Detection and the Problem of Offensive Language." In *Eleventh International Aaai Conference on Web and Social Media*.

Del Vigna12, Fabio, Andrea Cimino23, Felice Dell'Orletta, Marinella Petrocchi, and Maurizio Tesconi. 2017. "Hate Me, Hate Me Not: Hate Speech Detection on Facebook." In *Proceedings of the First Italian Conference on Cybersecurity (ITASEC17)*, 86–95.

Djuric, Nemanja, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015. "Hate Speech Detection with Comment Embeddings." In *Proceedings of the 24th International Conference on World Wide Web*, 29–30. ACM.

Gagliardone, Iginio, Danit Gal, Thiago Alves, and Gabriela Martinez. 2015. *Countering Online Hate Speech*. UNESCO Publishing. https://www.google.com/books?hl=en&lr=&id=WAVgCgAAQBAJ&oi=fnd&pg=PA3&dq=online+hate+speech+unesco&ots=TaamaoJQVB&sig=xUFAShQSkdkdHtMImSPL50myDRE.

Gambäck, Björn, and Utpal Kumar Sikdar. 2017. "Using Convolutional Neural Networks to Classify Hate-Speech." In *Proceedings of the First Workshop on Abusive Language Online*, 85–90.

Co-Funded by the Rights, Equality and Citizenship Programme of the European Union (2014 - 2020)

Gomez, Raul, Jaume Gibert, Lluis Gomez, and Dimosthenis Karatzas. 2019. "Exploring Hate Speech Detection in Multimodal Publications." *ArXiv Preprint ArXiv:1910.03814*.

Meza, Radu Mihai, Hanna-Orsolya Vincze, and Andreea Mogos. 2018. "Targets of Online Hate Speech in Context." *Intersections* 4 (4).

Nobata, Chikashi, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. "Abusive Language Detection in Online User Content." In *Proceedings of the 25th International Conference on World Wide Web*, 145–153.

Orasan, Constantin. 2018. "Aggressive Language Identification Using Word Embeddings and Sentiment Features." In . Association for Computational Linguistics.

Park, Ji Ho, and Pascale Fung. 2017. "One-Step and Two-Step Classification for Abusive Language Detection on Twitter." *ArXiv Preprint ArXiv:1706.01206*.

Pereira-Kohatsu, Juan Carlos, Lara Quijano-Sánchez, Federico Liberatore, and Miguel Camacho-Collados. 2019. "Detecting and Monitoring Hate Speech in Twitter." *Sensors* 19 (21): 4654.

Salminen, Joni, Hind Almerekhi, Milica Milenković, Soon-gyo Jung, Jisun An, Haewoon Kwak, and Bernard J. Jansen. 2018. "Anatomy of Online Hate: Developing a Taxonomy and Machine Learning Models for Identifying and Classifying Hate in Online News Media." In *Twelfth International AAAI Conference on Web and Social Media*.

Schmidt, Anna, and Michael Wiegand. 2017. "A Survey on Hate Speech Detection Using Natural Language Processing." In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, 1–10.

Silva, Leandro, Mainack Mondal, Denzil Correa, Fabrício Benevenuto, and Ingmar Weber. 2016. "Analyzing the Targets of Hate in Online Social Media." In *Tenth International AAAI Conference on Web and Social Media*.

Waseem, Zeerak. 2016. "Are You a Racist or Am i Seeing Things? Annotator Influence on Hate Speech Detection on Twitter." In *Proceedings of the First Workshop on NLP and Computational Social Science*, 138–142.

Zhang, Ziqi, David Robinson, and Jonathan Tepper. 2018. "Detecting Hate Speech on Twitter Using a Convolution-Gru Based Deep Neural Network." In *European Semantic Web Conference*, 745–760. Springer.

Zia, T., M. S. Akram, M. S. Nawaz, B. Shahzad, A. M. Abdullatif, R. U. Mustafa, and M. I. Lali. 2016. "Identification of Hatred Speeches on Twitter." In *Proceedings of 52nd The IRES International Conference*, 27–32.

Suryatej Reddy Vyalla, Vishaal Udandarao, Tanmoy Chakraborty "Memeify: A Large-Scale Meme Generation System" 1-5.

Gilbert Tranner. "Introduction to Uber's Ludwig" https://gilberttanner.com/blog/introduction-to-ubers-ludwig