



# FINAL REPORT

## Deploying an Accurate Classifier to Stop Online Violence Against Children using NLP

Dec 2022- Jan 2023

DATE: 01/02/2023

A project by the Omdena Marseille Chapter



## Table of Contents

<b><i>Executive Summary</i></b>	<b>3</b>
<b><i>1. Introduction</i></b>	<b>3</b>
<b>1.1. Problem statement</b>	<b>3</b>
<b>1.2 Previous related work on the topic</b>	<b>3</b>
<b>1.2 Project goal</b>	<b>4</b>
<b><i>2. Our solution</i></b>	<b>4</b>
<b>2.1 Domain research</b>	<b>4</b>
<b>2.2 Dataset creation</b>	<b>5</b>
2.2.1 Data collection	5
2.2.2 Data annotation	6
2.2.3 Data pre-processing	7
2.2.4 Exploratory Data Analysis (EDA)	7
<b>2.3 Model experimentation and improvement</b>	<b>8</b>
2.3.1 The selected model	9
2.3.2 Fine-tuning using native Tensorflow	9
After 3 epochs, he obtained an accuracy of 0.79 and a val_accuracy of 0.72, both of which are satisfying.	9
2.3.3 Fine-tuning using the TFTrainer class	10
<b>2.4 Model deployment</b>	<b>11</b>
<b><i>3. Conclusions</i></b>	<b>12</b>
<b><i>References</i></b>	<b>13</b>
Author	13
Participants (in alphabetical order)	13

## Executive Summary

'Deploying an Accurate Classifier to Stop Online Violence Against Children using NLP', a project by the Omdena Marseille Local Chapter led by chapter leader Alexandre Iang, lasted 8 weeks and saw the cooperation of an international team of 12 AI engineers. These cooperated steadily to define the problem, collect sentence that fall into the category of 'hate' from social media, pre-process and annotate them, create a ML algorithm for their classification based on pre-defined labels, and eventually deploy a POC of the model in Streamlit. The project, which run smoothly and was completed successfully, will be followed by a chapter which will aim at improving both the model and the way the model is deployed.

### 1. Introduction

This challenge was created to deploy an accurate classifier to identify grooming behavior in online chats with children. Its main aim was to stop online violence against children. The challenge, which united an international team of AI engineers over 8-weeks, was led by chapter lead Alexandre Iang.

The common language for the chapter was English, although all chapter work was done on French data. Most engineers had at least a working knowledge of French. Platforms such as GitHub, Notion, Trello, and a dedicated Slack channel were used to coordinate and keep track of the engineer's work.

#### 1.1. Problem statement

The project was designed to reduce Online Sexual Exploitation and Abuse of Children (OSEAC). With a 15,000% rise in online Child Sexual Abuse Materials (CSAM) online from 2005 to 2020, it is clear that online child violence is growing exponentially. In 2021, the National Center for Missing and Exploited Children's CyberTipline received 29.3 million reports of CSAM, making 2021 the worst year on record for online child sexual abuse.

Online grooming is the primary way that adults with a sexual interest in children or those who wish to harm them utilise to approach their preys ([1], [2]). "Grooming is a multidimensional phenomenon in which an adult aims to solicit a child into a seemingly voluntary interaction with the intention of sexually abusing that child." In a study Save the Children published last year, Grooming in the Eyes of a Child ([3]), we found that children who are the object of grooming often do not realize what is happening so they do not recognize they are in danger until they are being extorted into providing increasingly harmful imagery or even to meeting an online predator in person.

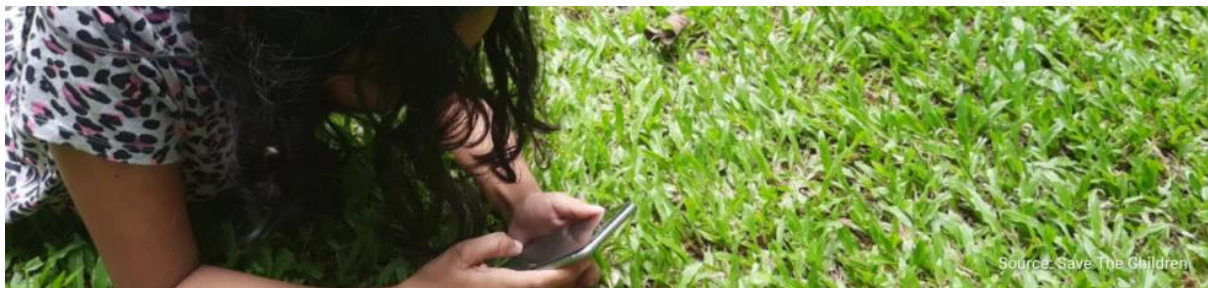
#### 1.2 Previous related work on the topic

In 2020, Save the Children US collaborated with Omdena to address online violence. Of the various products that were generated from the sprint, the most promising was a classifier algorithm using Natural Language Programming to identify online grooming combined with a chatbot that can warn the children that they may be chatting with a groomer. Since then, a team of three engineers associated with the original project has continued to refine the technology. The core team now wants to expand on the work to build an industry usable solution at scale.

From the original challenge, the team has a large dataset of more than 800,000 lines taken from the Perverted Justice project, a project from 2003 to 2019 that used online volunteers as decoys to entrap predators that sought to contact minors to obtain sexual images or videos from them or to meet them in person. During the challenge and afterward, the team tagged much of the training data with labels, such as male or female, predator or victim, and level of risk of the conversation, but the data still requires extensive processing, and in particular, the team need to improve and systematize the way judge and annotate the level of risk. In addition to the data already have, the team is actively attempting to obtain additional databases of online grooming chats from a variety of sources, such as law enforcement agencies.

## 1.2 Project goal

The team goal was to stop online violence against children by deploying an accurate classifier to identify harmful/grooming behaviours in online chats with children.



## 2. Our solution

For this challenge, we decided to collect additional data on hate speech from the Internet, classify them using five labels related to the field of ‘hate’, and then clean the data and use it to fine-tune a pre-existing language model.

### 2.1 Domain research

The first days of the project were dedicated to the onboarding, and to the definition of the materials to be used by the engineers to acquire domain knowledge.

At this early stage, different classes of hate were identified. Five among the most important classes of hate were defined as followed:

- **Sexism:** Refers to acts of violence that: 1) occur or linger in cyberspace; 2) are sexist, or sexual in nature; and 3) who reiterates dominant gender norms targeting girls and boys (tarnishing the former’s reputation and threatening the latter’s masculinity);
- **Racism:** Cyber racism is most commonly defined as racism which occurs in the cyber world. This includes racism which occurs on the internet such as racist websites, images, blogs, videos, and online comments as well as racist comments, images or language in text messages, emails or on social networking sites. It can be defined more broadly as any use of information and communication technologies to transmit racist

attitudes and behaviour including the transfer of racially offensive content that is intended to cause harm or distress to another person [4];

- **Homophobia:** Sexual minorities often make greater use of the internet to look for specific socialization environments in which they can meet other people with the same orientation or can avoid face-to-face social rejection and homophobic bullying. Paradoxically, this greater use of the internet to escape offline discrimination could lead to greater exposure to OSVR (Online Sexual Victimization and Risks). The internet is an environment that reproduces societal prejudices, so it is reasonable to think that homophobia and discrimination will also be present online, causing higher rates of OSVR among sexual minorities. In turn, the higher rate of OSVR could partially explain the higher rate of negative mental health outcomes found among sexual minorities [5];
- **Hate speech:** In common language, ‘hate speech’ refers to offensive discourse targeting a group, or an individual based on inherent characteristics (such as race, religion or gender) and that may threaten social peace. To provide a unified framework for the United Nations to address the issue globally, the UN Strategy and Plan of Action on Hate Speech defines hate speech as...“any kind of communication in speech, writing or behaviour, that attacks or uses pejorative or discriminatory language with reference to a person or a group on the basis of who they are, in other words, based on their religion, ethnicity, nationality, race, colour, descent, gender or other identity factor.” [6];
- **Bullying:** Bullying is a form of aggressive behaviour in which someone intentionally and repeatedly causes another person injury or discomfort. Bullying can take the form of physical contact, words, or more subtle actions. The bullied individual typically has trouble defending him or herself and does nothing to “cause” the bullying. Cyberbullying is verbally threatening, or harassing behaviour conducted through such electronic technology as cell phones, email, social media, or text messaging. Cyberbullying is verbally threatening, or harassing behaviour conducted through such electronic technology as cell phones, email, social media, or text messaging [7].

The five classes above were used to determine which data to collect and later, as labels for data annotation.

## 2.2 Dataset creation

The purpose of this task was to collect a dataset of hate speech in French language from various social media networks. The collected data was subsequently used to develop an artificial intelligence (AI) algorithm to detect hate speech on social media.

The task leader for this task was Mohamed Khandil.

### 2.2.1 Data collection

The data collection method used in this task is web scraping, a cost-effective and efficient way to collect data from social media networks. This method involves automatically extracting data from websites using python code and/or no-code tools. The scraping scripts used in the project are available in the project's repository, in the task's directory: <https://github.com/OmdenaAI/marseille-chapter-stop-online-violence-nlp/tree/main/src/tasks/task-1-data-collection-and-eda>

These were developed by Lamia Sekkai, Mohamed Kandil, Mk\_Veee, Rukshar Alam, Romy Sayah, Mohsen Selseleh.

The datasets collected and utilised in the project are listed in the Table below, along with the names of those who participated in the task:

Dataset	Dataset name	Contributors
1	Twitter	Lamia Sekkai, Hemanth Sai, Mohamed Kandil
2	Reddit	Mohamed Kandil

Data from Snapchat, Instagram, and Reddit were collected but discarded due to lack of participants in the annotation task.

The datasets in the Table above are stored in the task's directory: [https://github.com/OmdenaAI/marseille-chapter-stop-online-violence-nlp/tree/main/src/data/raw\\_data](https://github.com/OmdenaAI/marseille-chapter-stop-online-violence-nlp/tree/main/src/data/raw_data).

During this task, helpful tutorials from platforms like YouTube and Medium were shared with the team by Mohammed Raouf, Ganesh Lokare, Caterina Bonan, and Mohamed Kandil. The existence of public datasets on the topic, which were however not used for training purposes, was pointed out by Ganesh Lokare and Mariana Escobar Quiceno. The chapter leader Alexander Lang additionally conducted a workshop tutorial on how to build an ETL pipeline with the twitter API and airflow.

### 2.2.2 Data annotation

Once the dataset was established, the data had to be annotated. Data annotation is needed to help AI algorithms learn to do a required task accurately, e.g., in our case, all instances of hate speech.

To the request of the task leader Mohamed Khandil, all sentences were manually annotated using the categories of hate defined above:

- SEXISM;
- RACISM;
- HOMOPHOBIA;
- HATE SPEECH;
- BULLYING.

An additional class, called 'NONE', was used for those sentences that were not suitable for the current project.

The annotation team included Lamia Sekkai, Amine Tiffal, Hemanth Sai, Alexander Lang, Caterina Bonan, Rukshar Alam, Chukwudi Okereafor, and Haider Ali Khan. The tool used for annotation was [Doccano](#). [Label Studio](#) was also considered but eventually disregarded.

The annotated data files are stored in the chapter's repository's dedicated directory: [https://github.com/OmdenaAI/marseille-chapter-stop-online-violence-nlp/tree/main/src/data/annotated\\_data](https://github.com/OmdenaAI/marseille-chapter-stop-online-violence-nlp/tree/main/src/data/annotated_data).

### 2.2.3 Data pre-processing

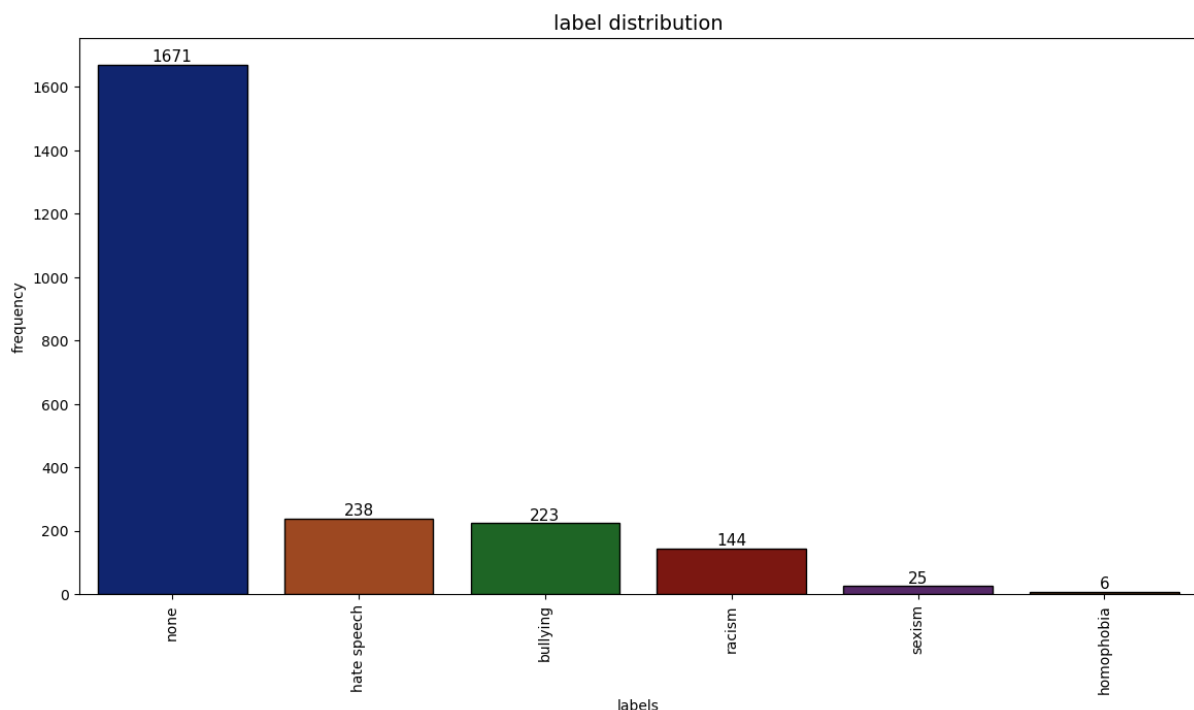
The annotated data was pre-processed to make it ready for the model development step. The pre-processing involved removing stop words, punctuation, URLs, emojis, and other irrelevant information. This step is fundamental in any ML pipeline based on textual data, as it helps the AI algorithm to focus on the most relevant information, thus improving its accuracy (in our case, in detecting hate speech). However, some of the steps listed above are not required if the models to be used to implement hate speech detection are Transformers or Language Model based.

The data pre-processing team was led by Mohamed Khandil and composed of Lamia Sekkai and Hemanth Sai.

### 2.2.4 Exploratory Data Analysis (EDA)

According to Wikipedia [8], exploratory data analysis (EDA) is an approach of analysing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods.

One of our collaborators, Hassan Outlaouait, worked on the Twitter data and noticed that the labels used by the various collaborators needed to be standardised, which he did by choosing to sentence case them. He also signalled the overall data imbalance between the six chosen labels, and suggested to balance the dataset using the SMOTE technique for oversampling. The imbalance is illustrated in the following graph created by Chukwudi Okereafor:



Okereafor also graciously provided the following word cloud for our datasets:







### 2.3.1 The selected model

The model chosen for our classifier was the one developed by Chukwudi Okereafor. Okereafor fine-tuned [DistilBert](#) for our multi-class text classification problem using the Twitter dataset.

Fine-tuning in the HuggingFace's transformers library involves using a pre-trained model and a tokenizer that is compatible with that model's architecture and input requirements. Each pre-trained model in transformers can be accessed using the right model class and be used with the associated tokenizer class. Since we needed DistilBert for a classification task, Okereafor chose the DistilBertTokenizer class to tokenize our texts and then used TFDistilBertForSequenceClassification model class in a later section to fine-tune the pre-trained model using the output from the tokenizer.

The DistilBertTokenizer generates input\_ids and attention\_mask as outputs. This is what is required by a DistilBert model as its inputs. Therefore, Okereafor defined the tokenizer object using the from\_pretrained() method which downloads and caches the tokenizer files associated with the DistilBert model. He used padding and truncation to make sure all the vectors were the same size.

Once the texts in an encoded form, there is one further step is needed before beginning the fine-tuning process: the conversion of the input encodings and labels into a TensorFlow Dataset object. Okereafor did this by passing them to the from\_tensor\_slices constructor method.

Okereafor fine-tuned Distilbert using two different methods, which are overviewed below.

### 2.3.2 Fine-tuning using native Tensorflow

The from\_pretrained() method is used to initialize a pre-trained model. This loads in the weights and initializes the model with the defined configurations. The DistilBert model and other models available in the transformers library are standard tf.keras.Model classes (and torch.nn.Module in the case of Pytorch), and so they can be used just as any native TensorFlow and Keras API. The results of the training can be seen in the image below:

```
Epoch 1/6
WARNING:tensorflow:From C:\Users\PANDORA\anaconda3\lib\site-packages\tensorflow\python\autograph\pyct\static_analysis\liveness.py:83:
Analyzer.lamba_check (from tensorflow.python.autograph.pyct.static_analysis.liveness) is deprecated and will be removed after 2023-09-23.
Instructions for updating:
Lambda fuctions will be no more assumed to be used in the statement where they are used, or at least in the same block.
https://github.com/tensorflow/tensorflow/issues/56089
115/115 [=====] - 399s 3s/step - loss: 0.9206 - accuracy: 0.7426 - val_loss: 0.8543 - val_accuracy: 0.7468
Epoch 2/6
115/115 [=====] - 335s 3s/step - loss: 0.7553 - accuracy: 0.7705 - val_loss: 0.7030 - val_accuracy: 0.7662
Epoch 3/6
115/115 [=====] - 341s 3s/step - loss: 0.6209 - accuracy: 0.7924 - val_loss: 0.6718 - val_accuracy: 0.7749
Epoch 4/6
115/115 [=====] - 355s 3s/step - loss: 0.4879 - accuracy: 0.8341 - val_loss: 0.7031 - val_accuracy: 0.7511
Epoch 5/6
115/115 [=====] - 352s 3s/step - loss: 0.3748 - accuracy: 0.8680 - val_loss: 0.7365 - val_accuracy: 0.7641
Epoch 6/6
115/115 [=====] - 333s 3s/step - loss: 0.3356 - accuracy: 0.8976 - val_loss: 0.7079 - val_accuracy: 0.7965
```

After 3 epochs, he obtained an accuracy of 0.79 and a val\_accuracy of 0.72, both of which are satisfying.

### 2.3.3 Fine-tuning using the TFTrainer class

The TFTrainer (Trainer for Pytorch) is a class provided by the transformers library that offers a simple, yet feature-rich, method of training and evaluating models. The following code by Okereafor demonstrates how to define the configuration settings and build a model using the TFTrainer class:

```
1 #from transformers import TFDistilBertForSequenceClassification, TFTrainer, TFTrainingArguments
2
3 #training_args = TFTrainingArguments(
4     #output_dir='./results',          # output directory
5     #num_train_epochs=3,              # total number of training epochs
6     #per_device_train_batch_size=16,  # batch size per device during training
7     #per_device_eval_batch_size=64,   # batch size for evaluation
8     #warmup_steps=500,                # number of warmup steps for learning rate scheduler
9     #weight_decay=0.01,               # strength of weight decay
10    #logging_dir='./logs',             # directory for storing logs
11)
12
13 #with training_args.strategy.scope():
14     #trainer_model = TFDistilBertForSequenceClassification.from_pretrained("distilbert-base-uncased", num_labels=5)
15
16 #trainer = TFTrainer(
17     #model=trainer_model,              # the instantiated 🤗 Transformers model to be trained
18     #args=training_args,               # training arguments, defined above
19     #train_dataset=train_dataset,      # training dataset
20     #eval_dataset=val_dataset,         # evaluation dataset
21)
```

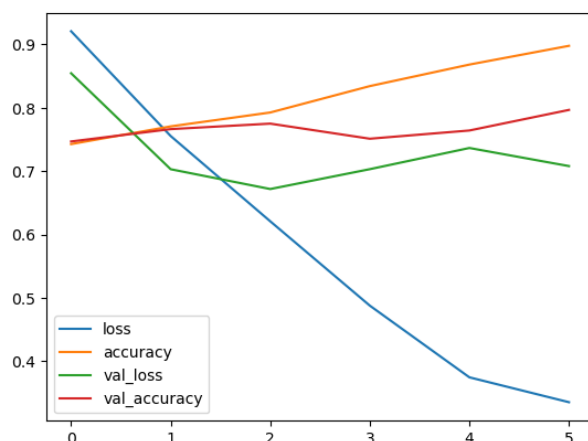
Python

The TFTrainingArguments is how customization arguments are set for the training loop. This makes them available for later use in the TFTrainer class. The model is instantiated using the TFDistilBertForSequenceClassification class, and then finally built by instantiating the TFTrainer class and passing in the different options defined along with the datasets.

The results of this fine-tuning can be seen below:

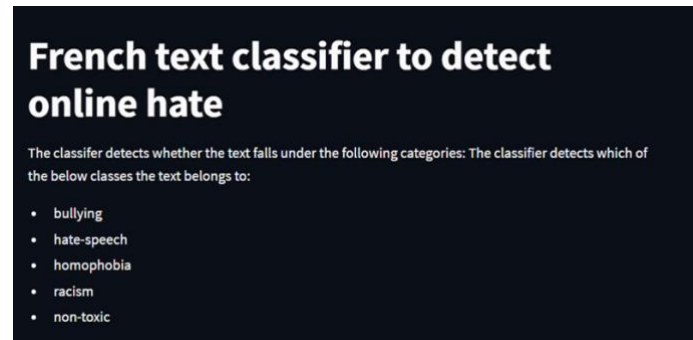
	loss	accuracy	val_loss	val_accuracy
0	0.920580	0.742607	0.854313	0.746753
1	0.755346	0.770537	0.702975	0.766234
2	0.620867	0.792443	0.671753	0.774892
3	0.487880	0.834064	0.703073	0.751082
4	0.374818	0.868018	0.736478	0.764069
5	0.335593	0.897590	0.707937	0.796537

A visualisation of model loss and accuracy is given below:



When testing the fine-tuned models to make predictions about new data, Okereafor demonstrated that both models yielded identical results, and 89% accuracy.

## 2.4 Model deployment

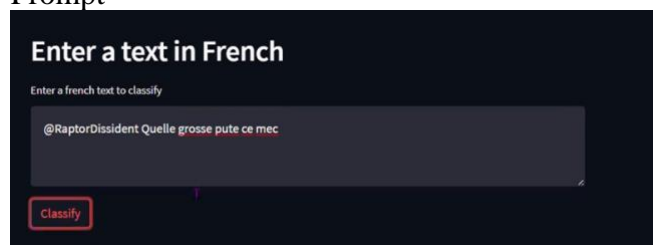


During the last two weeks of the project, the chapter leader Alexandre Iang worked on the deployment of a POC of the model in [Streamlit](#).

A POC, or proof of concept, is an experiment intended to show that a program, product, or system can be successfully deployed in the real world. Streamlit, on the other hand, is an open-source app framework for Machine Learning and Data Science. It helps create and publish web apps in a short time. It is compatible with major Python libraries such as scikit-learn, Keras, PyTorch, SymPy(latex), NumPy, pandas, Matplotlib etc., and it does not require any front-end experience.

The POC for this project is quite simple and has the form of an interface that prompts a sentence in French from the user, and then classifies it using our ML model. Below is an example of 'hate speech' and 'homophobic' prompts, and their correct classifications:

### Prompt



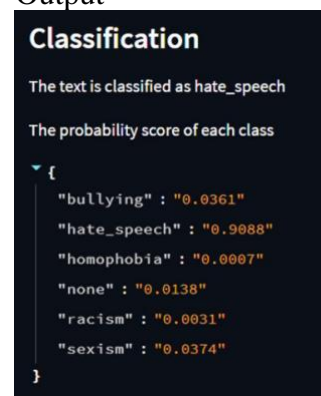
**Enter a text in French**

Enter a french text to classify

@RaptorDisident Quelle grosse pute ce mec

Classify

### Output



Enter a french text to classify

'oh menace de mort ? Petit signalement on verra si tu vas rester longtemps sur insta. Vas donner ton petit Q au vieux gay ça t'ira bien de faire une page comme ça plutôt qu'une page politique où tt est faux'

Classify

### Classification

The text is classified as homophobia

The probability score of each class

```
{
  "bullying": "0.0003"
  "hate_speech": "0.0003"
  "homophobia": "0.9964"
  "none": "0.0012"
  "racism": "0.0005"
  "sexism": "0.0013"
}
```

As illustrated in the image above, the application provides not only a classification for the sentence, but also the probabilities that the sentence has to belong to all considered classes. As was our goal, also non toxicity is correctly detected, as illustrated by the sentence below and its classification.

### Prompt

Enter a french text to classify

@maxvandewiele penser qu'il y a trop d'immigrés n'est pas du racisme, l'immigré n'est pas une « race » Et ils peuvent s'appuyer sur des arguments économiques; culturels, patrimoniaux pour avancer cette pensée Donc pas du racisme

Classify

### Output

The text is classified as non\_toxic

The probability score of each class

```
{
  "bullying": "0.0008"
  "hate_speech": "0.0005"
  "homophobia": "0.0003"
  "none": "0.9977"
  "racism": "0.0005"
  "sexism": "1e-04"
}
```

## 3. Conclusions

The chapter was a success and lead to the creation of a POC of our classification model, which we published in Streamlit. When it comes to the project's main goal, that of tackling on-line violence against children, our classifier performs in a satisfactory way. The results are certainly promising, and can be considered a first yet huge step towards our joint goal to stop on-line violence against minors altogether.

Further work can be forecasted to improve both the model and the way the model is deployed. The team are now planning a follow-up chapter in which a custom extension that will predict the content of a text in real time will be created, and subsequently deployed within an EC2 instance on Amazon Web Services (AWS). The final application should work as follows: once suspicion of grooming reaches a threshold based on its similarity to the training data, it will trigger an action, which may differ depending on the platform it is deployed on and the objectives of the intervention. As an example, the team may warn the child through the chatbot without alerting the groomer, call a moderator, or shut down the chat entirely.

Future extensions of the project to languages other than French are desirable and should be encouraged.

## References

- [1] Sørensen K. 2015. Grooming – a strategic process. Published in: Is it really that bad? An anthology of online sexual abuse of children and young people, written by Jakobsen G, Sørensen K, Almind H & Gundorff H. Save the Children Denmark.  
<https://resourcecentre.savethechildren.net/node/12241/pdf/is-it-really-that-bad-an-anthology-of-online-sexual-abuse-of-children-and-young-people.pdf>
- [2] Greijer S. & Doek J. (2016). Terminology Guidelines for the Protection of Children from Sexual Exploitation and Sexual Abuse. ECPAT International and ECPAT Luxembourg. <http://luxembourgguidelines.org/>
- [3] [https://pelastakaalapset.s3.eu-west-1.amazonaws.com/main/2021/08/03151159/grooming\\_in\\_the\\_eyes\\_of\\_a\\_child\\_2021.pdf](https://pelastakaalapset.s3.eu-west-1.amazonaws.com/main/2021/08/03151159/grooming_in_the_eyes_of_a_child_2021.pdf)
- [4] <https://racismnoway.com.au/about-racism/cyber-racism/>
- [5] <https://www.sciencedirect.com/science/article/pii/S0747563221000509>
- [6] <https://www.un.org/en/hate-speech/understanding-hate-speech/what-is-hate-speech>
- [7] <https://www.apa.org/topics/bullying>
- [8] [https://en.wikipedia.org/wiki/Exploratory\\_data\\_analysis](https://en.wikipedia.org/wiki/Exploratory_data_analysis)

## Author

This report was written and submitted by Caterina Bonan, Postdoctoral Researcher at the University of Cambridge, who takes full responsibility for any mistake or inaccuracy in it.

## Participants (in alphabetical order)

Alexandre Iang (chapter leader);  
Amine Teffal;  
Caterina Bonan;  
Chukwudi Okerefor;  
Divya Muthu Krishnan;  
Haider Ali Khan;  
Hassan Outlaouait;  
Hemanth Sai;  
Lamia Sekkai;  
Mohamed Kandil;  
Romy Sayah;  
Rukshar Alam;  
Vedanth Baliga.