# Hate Speech Detection with Deep Learning Models in PyTorch

Carlos Eduardo Posada
c.posada@mpp.hertie-school.org

Maximilian Kupi*
m.kupi@mpp.hertie-school.org

Michael Bodnar
m.bodnar@mpp.hertie-school.org

Nikolas Schmidt*
n.schmidt@mpp.hertie-school.org

## 1. Introduction

This project aims to develop and implement a tool that identifies hate speech on social media. We plan to apply a two-sided hybrid natural language deep learning approach to classify data retrieved from Twitter.

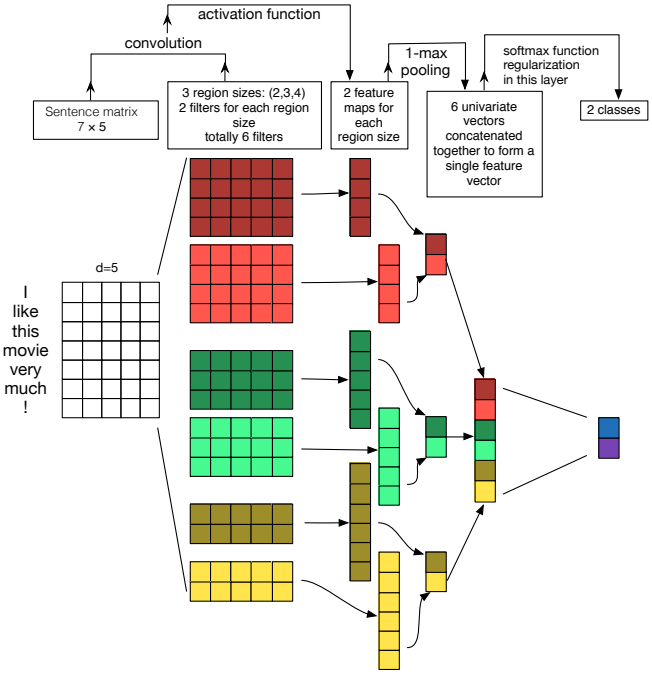


Figure 1. Example illustration of a Convolutional Neural Network architecture for sentence classification [7]

Firstly, we plan to apply two different data pre-processing methods to obtain two input matrices:

(a) transforming technique to create word embedded data from two pre-existing datasets of tweets

(b) dictionary approach to count hateful terms from a database

Secondly, we plan to employ a convolutional neural network (see Figure 1) in order to classify hate speech based on the two input matrices.

The literature on hate speech detection has evolved over the past years, regarding its definition, its context of occurrence, and the applied computational methods. In its beginnings, surface feature extractions of the text (e.g. token and character n-grams) were combined with other methods (e.g. bag of words, word generalisation or term frequency-inverse document frequency) for classification [6].

Before neural networks, a number of machine learning algorithms were employed in the literature, for example support vector machines. However, in recent years, neural networks have become the state of the art – in particular in combination with the use of different word-embedding techniques to represent and group text data in a vector space.

Research has also explored the utilization of further characteristics of hate speech: Sentiment analysis can work as a sub-approach to hate speech detection, which focuses on strong negative sentiments to differentiate normal from hate speech [6]. However, this has not improved results substantially, in particular when considering the legally-binding definitions of hate speech. For this reason, we plan to use lexical resources with defined hate terms.

The next section is going to delve deeper into the motivation to automatically detect hate speech. Section 3 will introduce the methods we plan to use for evaluating our model, while the following section will introduce the datasets and computational setup. The proposal will conclude with detailing the project's modules and each individual team member's contribution thereto.

## 2. Motivation

With the widespread use of social media, there are growing concerns about hateful content. Abusive content, such as tweets and Facebook comments, target individuals or particular groups based on ethnicity, nationality, gender iden-

*These authors will be sharing the project between the NLP and Python class.

tity, sexual orientation or disabilities. Studies have shown that social media can serve as a propagation mechanism and suggested hateful online messages to be predictors of real-world hate crimes [4].

In order to discourage violence, social networks and other online venues have sought to filter these harmful messages from their platforms. However, manual removal of hateful comments is unfeasible due to the high volume of comments in a growing number of social media venues. Thus, most recently, automatic methods have been employed. As explained in the previous section, detecting, classifying and removing hate speech is a complex task from a technical point of view.

Some of the difficulties of these methods include [3]:

- The lack of a common definition of hate speech, which leads to different databases of hateful comments built with different classification criteria. Hate speech detection tools which are trained on these databases can thus carry biases.
- The nuances of natural language, such as sarcasm, sentiment and double meanings, increase the difficulty of discerning between hate speech and allowed free expression.
- Keeping up with the lexical evolution of hate speech leads to a need to constantly update the datasets.

Given these difficulties, the use of deep learning methods can bring significant improvements in hate speech detection. For this reason, in a context of high polarization, rising social tensions, and potential deadly consequences of hate speech, we find this project to be particularly relevant.

## 3. Evaluation

We will split the data into

1. Training dataset (70%),
2. Validation dataset (15%), and lastly,
3. Testing dataset (15%)

in order to train and test the model on current standards.

Furthermore, we will use two different measures to compare the quality of different approaches and to judge the overall quality of our model:

1. Accuracy
2. F1-score

Our model will have three different categories that were already labelled in the datasets. We can calculate the accuracy in each category by comparing the output of the model to the labels. The total distribution of the model output will be compared to the total distribution of the gold standard labels.

We will use the $F1$ score for the hate speech category to compare our different approaches. The $F1$ score is a weighted average of *precision* and *recall*.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} = \left( \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \right)$$

*Precision* is the ratio between correctly identified hate speech entries ($TP$, true positive) and overall identified hate speech entries ($TP + FP$, true positive and false positive).

$$Precision = \frac{TP}{TP + FP}$$

*Recall* is the ratio between true positives ($TP$) and the sum true positives and false negatives ($FN$), i.e. hate speech mistakenly identified as not hate speech [5].

$$Recall = \frac{TP}{TP + FN}$$

## 4. Resources

The following datasets will be used in the project:

1. The "Automated Hate Speech Detection and the Problem of Offensive Language" dataset from Davidson et al. [1], comprised of 25,000 tweets
2. The "Hate and Abusive Speech on Twitter" dataset from Founta et al. [2], comprised of 100,000 tweets

Both datasets have the following respective categories, according to which the data is labelled:

1. hate_speech, offensive_language, neither
2. hateful, abusive, normal, spam

In order to merge both datasets, will filter the spam category in the second dataset.

Our computing setup (see Figure 2 on next page) includes AWS resources which will be deployed through AWS SageMaker. We will use Visual Studio Code and Google Colab to program and preliminary test our model. The training of the model will happen on the remote AWS resources. We will use a GitHub repository [1] to exchange the code between these instances as well as to document the project.

[1] https://github.com/MaximilianKupi/nlp-project.

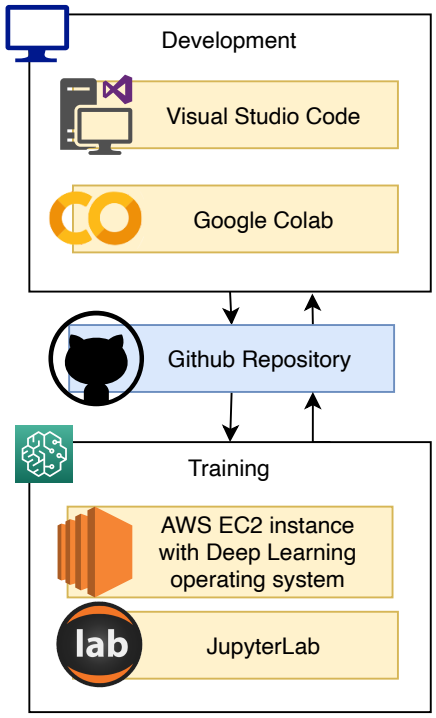


Figure 2. Deployment pipeline including IDE for experimentation and AWS resources for training

## 5. Contributions

We split the model design and implementation into two modules and their respective sub-tasks. For each module we assigned a primarily responsible sub-team (in parentheses):[2]

1. Data pre-processing and loading
   (E. Posada & M. Kupi)
   (a) Obtaining and cleaning the datasets
   (b) Twitter specific text pre-processing
   (c) Word / sentence vectorisation (as first data input)
   (d) Implementing a dictionary approach potentially based on Hatebase.org (as second data input)
   (e) Splitting data into test, validation and testing set
   (f) Specifying and implementing the data loader
   (g) Testing and iterating the module
   (h) Creating module-specific visualisations for the final paper

2. Model architecture and training
   (M. Bodnar & N. Schmidt)
   (a) Choosing width and depth of the model
   (b) Choosing optimizer as well as activation and loss functions
   (c) Choosing stopping rule, regularisation, dropout, learning rates etc.
   (d) Potentially performing a hyperparameter grid search
   (e) Running and tracking the training
   (f) Training and tuning the model based on the results of the validation set
   (g) Creating module-specific visualisations for the final paper

The paper writing will be divided according to the technical tasks performed by each of the team members. All creative ideation takes place in the whole team. Project management as well as facilitation of the working sessions is done by M. Kupi.

[2] We have formed the sub-teams such that each sub-team consists of one team member attending both the NLP and the Python class. This person will be the "lead coder" in the sub-team.

## References

[1] T. Davidson, D. Warmsley, M. Macy, and I. Weber. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515, 2017.

[2] A.-M. Founta, C. Djouvas, D. Chatzakou, I. Leontiadis, J. Blackburn, G. Stringhini, A. Vakali, M. Sirivianos, and N. Kourtellis. Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior. Technical report, AAAI Press, 2018.

[3] S. MacAvaney, H. R. Yao, E. Yang, K. Russell, N. Goharian, and O. Frieder. Hate speech detection: Challenges and solutions. *PLoS ONE*, 14(8):1–16, 2019.

[4] K. Müller and C. Schwarz. Fanning the Flames of Hate: Social Media and Hate Crime. *SSRN Electronic Journal*, 2017.

[5] H. Nottelmann and U. Straccia. sPLMap: A probabilistic approach to schema matching. In *Lecture Notes in Computer Science*, page 89, 2005.

[6] A. Schmidt and M. Wiegand. A Survey on Hate Speech Detection using Natural Language Processing. *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10, 2017.

[7] Y. Zhang and B. Wallace. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. *arXiv e-prints*, 2015.