Understanding Class(ifier) Differences
XAI Lab Course WS20/21

Felix Dietrich
Technical University of Munich (TUM)
felixtj.dietrich@tum.de

Abstract

Understanding class differences and classifier differences is a problem at the heart of machine learning. We want to know what drives the decision-making process. For this, we reproduce an inter-dataset class similarity and intra-dataset class homogeneity approach to find class differences. In an attempt at finding classifier differences, we use the explainability package LIME for experiments using its feature importance scores.

1 Introduction

In machine learning, we are often unsure about the design choices we make for our classifiers and how the classification decisions come to be. We frequently don't know what makes one set of hyperparameters better than another set, or one architecture superior over another one. Knowing how a classification decision has been made increases trust in the system and makes it more transparent. It is hard to get an intuition about today's hard-to-interpret black-box machine learning models. It is vital to understand or explain the differences between classifiers to get better insights into their inner workings. Not only do we need to understand the classifier differences but also class differences based on the underlying data.
For our paper we choose hate-speech classification as our text classification task. We try to analyze five different datasets for finding class differences in Section 2. To analyze the categories of the datasets, we follow the approach of Fortuna et al. (2020) in Subsection 2.2. We try to reproduce their inter-dataset class similarity, Subsection 2.2.1, followed by their intra-dataset class homogeneity, Subsection 2.2.2. To obtain a combined dataset for comparison and for broader abuse detection, we standardize the labels between the datasets, Subsection 2.3. For finding classifier differences, Section 3, we first establish and evaluate our viable classification methods, Subsection 3.1, and then turn to the *LIME* (Ribeiro et al., 2016) explainability package for finding explanation differences, Subsection 3.2. Finally we run experiments using the LIME feature importance explanations, Subsection 3.3, where we look at the average feature importance distributions, Subsection 3.3.1, the average cosign similarities of the feature importance scores between classifiers, Subsection 3.3.2, and a way of measuring the stability of a prediction when we omit the most important feature word, Subsection 3.3.3.
The LIME package produces local explanations as local feature importance scores. The feature importance scores indicate approximately how influential a feature, or in our case a word, is for the overall decision of the classifier. There are other explainability methods available such as *SHAP* (Lundberg and Lee, 2017) and *SAGE* (Covert et al., 2020), which produce local feature importance scores and global feature importance scores, respectively. SHAP and SAGE showed to be hard to integrate into our text classification task, because the feature space, the vocabulary, is too large.

2 Class Differences

For finding class differences we follow the approach by Fortuna et al. (2020). They analyze multiple datasets and compare them with experiments to find ways to improve upon guidelines for data collection or data annotation strategies in the future. We follow their experiments for *inter-dataset class similarity*, Subsection 2.2.1, and *intra-dataset class homogeneity*, Subsection 2.2.2, for five selected English Twitter hate-speech datasets. We use the results to create a combined dataset where we map the datasets' classes to either *non-abusive* or *abusive*, Subsection 2.3.

2.1 Datasets

There are lots of hate-speech datasets available covering different kinds of abuse. We want to select comparable datasets with the same content format. We therefore choose datasets covering abuse on the Twitter platform in the English language. Those constraints narrow down the available datasets by a lot. We select five promising datasets and refer to them by names: *Davidson* (Davidson et al., 2017), *Waseem* (Waseem and Hovy, 2016), *Basile* (Basile et al., 2019), *Zampieri* (Zampieri et al., 2019), and *Founta* (Founta et al., 2018). We refer to the union of all datasets as *Combined* dataset. An overview of the datasets can be found in Table 1.

The class distributions of the datasets are often imbalanced as seen in Figure 1. In the Davidson dataset, our second largest dataset containing 24783 tweets, a small amount of tweets is classified as hate speech (5.77%), the majority as offensive (77.4%), and the rest as neither. This makes this dataset the most imbalanced one of our datasets.

The Waseem dataset deals with racism (11.6%) and sexism (20.2%). We observe that the majority of the dataset does not contain hate or abusive speech with 68.03%. In the Basile dataset, our smallest dataset, 12971 tweets, a more balanced class distribution can be seen with 42.1% of the tweets classified as hate speech and 57.9% as not containing hate speech. For the Zampieri dataset we see a slightly more imbalanced dataset again, with the classes offensive (32.9%) and the majority not offensive with 67.1%. The Founta dataset is by far our largest dataset containing 99799 tweets. It also has the most classes with normal (53.9%), abusive (27.1%), hateful (5.0%), and spam (14.1%).

All datasets have their imbalances in their class distribution. Often this has to do with the way the data has been selected from Twitter. Usually tweets are collected using a keyword search for offensive or abusive words. The datasets vary heavily by their definitions of the classes, topic and authors of the tweets, and time period in terms of current events.

2.2 Analyzing categories

We follow the approach of Fortuna et al. (2020) and compare the categories across the different hate speech datasets. We compare the similarities between the categories of the various datasets and the homogeneity of each single category in the datasets. We do this by computing centroids of the tweets by averaging the FastText (Bojanowski et al., 2017) embeddings of each word in the tweet. For both comparisons we start the computation as follows:

1. Pre-process the tweets by lowercasing all words, normalizing user-names, urls, hashtags, and other elements using the *ekphrasis* (Baziotis et al., 2017) library. We use ekphrasis over NLTK since it is specialized to deal with texts from social networking services such as Twitter in our case.

2. Embed the words using FastText pre-trained 300-dimension English Common Crawl embeddings (Mikolov et al., 2018). They state in Fortuna et al. (2020), that they trained FastText embeddings using the English Wikipedia pre-trained embeddings, but as far as our research is concerned, there is no way to fine-tune the embedding using FastText as of now. We just use the provided embeddings without training them.

3. Compute the tweets' centroids by averaging the word embeddings of all the words in the respective tweet.

2.2.1 Inter-dataset class similarity

In this Subsection we compare the categories of all datasets by their semantic similarity. We do this by continuing the procedure in Subsection 2.2 with:

4. Compute the centroids of the categories by averaging the the tweets centroids of each category.

After acquiring the centroids of each category we reduce the dimensionality of those 300-dimensional class centroid vectors to a 2D representation by performing a Principal Component Analysis (PCA) (Pearson, 1901). The results of this PCA can be seen in Figure 2. We will standardize the labels, in Subsection 2.3, to abusive and non-abusive, red and blue respectively in the Figure 2. The abusive classes and non-abusive classes group relatively well into two distinct areas. The PCA results of Fortuna et al. (2020) shows the similarities between abusive classes without showing non-abusive categories in order to make a more fine-grained sub-categorization of abuse. We chose to also include the non-abusive classes and won't make such a fine-grained analysis for merging the categories of the datasets.

2.2.2 Intra-dataset class homogeneity

This Subsection compares the categories using their internal homogeneity. Homogeneity reflects the variation of tweets in a class. If the tweets inside a category are all very similar the category is more homogeneous if they are less similar they are more inhomogeneous. For computing the homogeneity we append the following steps to the process of Subsection 2.2:

4. Compute the cosign similarity distance between all tweets from the same category.

5. Average the cosign similarities of the same category to obtain the homogeneity metric of a category.

Figure 3 shows the result of this procedure. The non-abusive classes are grouped together below being less homogeneous than the abusive classes grouped above. This makes intuitively sense since the non-abusive classes are more broadly defined then the abusive classes and therefore more varied. Compared with Figure 2 of Fortuna et al. (2020) there seem to be some issues with either our calculations or theirs. Our resulting homogeneity scores are larger by one magnitude. The largest of their scores is roughly 0.04, whereas ours is at roughly 0.53. The relative sorted ordering of the classes is also off for the Waseem and Davidson dataset. We could not reproduce their numbers after checking the procedure multiple times. It might have to do with different pre-processing steps and the use of other embeddings, but we also checked for those cases.

2.3 Label standardization between datasets

Similar to the idea of Fortuna et al. (2020), we standardize our labels so that the categories of the datasets are comparable across the different datasets. We choose a binarization approach where we map the class labels of dataset to the categories non-abusive and abusive. We do not make a fine-grained destinction between different kinds of abuse. Later on in Subsection 3.1, we compare the F1-scores over different classifiers with the binarized datasets and also evaluate the performance on each dataset when we train the classifiers with all the datasets combined.

3 Classifier Differences

In this Section we try to find differences with classifiers. We first denote and evaluate the different classification options, Subsection 3.1, then introduce the LIME explainability package, Subsection 3.2, for running experiments using its feature importance explanations, Subsection 3.3. In our experiments, we look at the average feature importance distributions, Subsection 3.3.1, the average cosign similarities of the feature importance scores, Subsection 3.3.2, and a possible prediction stability metric, Subsection 3.3.3.

3.1 Classifiers

There are multiple viable ways to do text classification such as hate-speech classification. We look at seven viable, mostly classical machine learning, classification approaches from *scikit-learn* (Pedregosa et al., 2011) and fit them to our datasets for evaluation. We do the same for three different simple neural network architectures implemented using *TensorFlow* (Abadi et al., 2015). Table 2 shows the classifiers trained and evaluated macro F1-scores on each dataset. The classifiers up top are the almost out-of-the-box scikit-learn classifiers using *tf-idf* features: *LinearSVC*, *GaussianNB*, *ComplementNB*, *DecisionTreeClassifier*, *KNeighborsClassifier*, *RandomForestClassifier*, and *MLPClassifier*.
In the bottom part of the Table 2, we have our simple neural networks using small 16-dimensional word embeddings as features: *DenseClassifier* using a hidden 15-dimensional dense layer, *LSTMClassifier* using a 32+32-dimensional Bidirectional LSTM layer, and *CNNClassifier* using an 1D convolution layer with 64 filters and kernal size of 8 followed by a max-pooling layer and a 10-dimensional dense layer. Everything neural network is of course followed by the final dense classification layer.
Looking further at the macro F1-scores of Table 2, we can observe that the LinearSVC classifier performs well compared to our other classifiers. Comparing the two naive Bayes classifiers, GaussianNB and ComplementNB, we see that ComplementNB performs significantly better. This might be mainly because the ComplementNB is specially designed to handle our imbalanced datasets. Our neural classifiers seem to work equally well to other classifiers on the Combined dataset where we binarized our labels.
In an attempt to compare the performances of the classifiers across all datasets we also run a comparison over the binarized versions of the datasets. For binarizing the labels, we map the classes to abusive and non-abusive according to the inter-dataset class

similarity PCA results Figure 2. Each binarized dataset has now the same classes. In Table 3 we can observe the F1-scores evaluated on the respective test-set of the dataset for classifiers trained on the individual training-set of the respective dataset, indicated by superscript i, and classifiers trained on the training-set of the Combined dataset, indicated by superscript c. We can not compare the F1-scores with our previous Table 2, which shows macro F1-scores over multiple classes. We observe that we mostly lose points in our F1-score if we train on the Combined dataset compared to the individually trained ones. But there are some cases where we improve our scores. Using a classifier trained on the Combined dataset might be a way to create a general abuse detection model over different kinds of abuse. Although, we have to keep in mind that 59.2% of the data in the Combined dataset stems from the large Founta dataset.

3.2 LIME

We want to use a explainability method for finding further classifier differences. For this we use the *LIME* (Ribeiro et al., 2016) package, which is short for *local interpretable model-agnostic explanations*. LIME can be agnostically applied to a black-box model to get a local explanation for a decision instance. Figure 4 shows an example explanation for an instance of the Basile dataset using the LinearSVC classifier. LIME is approximating the prediction of the classifier locally with an interpretable model. In the example the classifier correctly predicts the hate speech class for the tweet. The feature importance scores located in the top-right indicate the most relevant words used for the predictions. The most influential words for the prediction are also highlighted in the tweet's text. The orange color contributes to hate speech and the blue color to the none class. We can observe, that the explanation correctly indicates the swear words as most important contributors for this hate speech instance.

3.3 LIME experiments

With LIME, introduced in the previous Subsection 3.2, we get local explanations for predictions of single instances. We want to attempt to combine multiple local explanations into a global explanation to get a global insight into the classifier differences. For this we focus on three classifiers in particular: ComplementNB, LinearSVC, and LSTMClassifier. The ComplementNB is a generative classifier, whereas the other two are discriminative ones. We run the LIME explainer for each classifier and each dataset for up to 3500 test-set instances, filtering out the few tweets containing less than 6 unique features i.e. words. We then try run more global experiments in the next Subsections.

3.3.1 Feature importance distributions

In our first experiment using the collected local explanations in form of feature importance scores, we calculate average feature importance distributions. The average distribution of those feature importance scores might vary with with classifiers and with the datasets. We want to compare the classifiers and analyze their average reliance on the top features, whether a classifier usually focuses on fewer or more features for its prediction than another. To do this we try to get an average feature distribution of the feature importance scores. We do this by taking the absolute values of the feature importance scores, sorting the values, normalizing each distribution to one, averaging over all distributions, and normalizing again the averaged distribution to one. The results of this process can be seen in Figure 5. The datasets' class distributions are shown below to better compare the distributions considering the class imbalances.
In Figure 5, the datasets Waseem, Basile, and Zampieri show no significant differences in the distributions among the classifiers. There are significant differences in the Davidson and Founta datasets. The discriminative classifiers, LinearSVC and LSTMClassifier, focus more on the most important feature, whereas the generative classifier, ComplementNB, has a more spread out feature distribution focusing on more features on average. The Davidson and Founta datasets seem to have more discriminative features than the Wassem, Dasile, and Zampieri datasets. Those datasets might contain tweets with more discriminative words such as swear words in the case of hate-speech detection.

3.3.2 Feature importance similaities

The second experiment takes a look at the average cosign similarity of the feature importance scores between classifier pairs over the datasets. We would expect the feature importance scores of an instance to be very similar across different classifiers. Figure 6 shows the resulting cosign similarities of feature importance scores averaged over many explanations. As expected, the feature importance scores are similar. We can observe that the

ComplementNB is the least similar to the LSTM-Classifier, and a bit more similar to the LinearSVC classifier. The two discriminative classifiers, LinearSVC and LSTMClassifier, are the most similar to each other. This seems to be consistent with the results of Subsection 3.3.1, where there are significant differences between generative and discriminative classifiers.

3.3.3 Classifier prediction stabilities

In this third experiment we attempt a potential classifier stability metric. We want further analyze how much the classifiers are dependent on the most influential feature. If we were to omit this most important feature, how often would our prediction change, in other words how stable would our prediction be. We can observe in Table 4 how much our F1-score changes if we omit the most influential feature word. ComplementNB seems to be usually affected the least by this omission. This might again be because of its nature as generative classifier. It is intuitive, that the performance of discriminative classifiers is more affected since we omit the most important discriminative feature of the instance. All three classifiers are performing significantly worse on the Zampieri dataset after the omission for an unknown reason.

4 Discussion

In this paper we where able to find some obvious class differences and classifier differences. We tried to reproduce the inter-dataset class similarity, Subsection 2.2.1, and intra-dataset class homogeneity, Subsection 2.2.2. Those two approaches showed some differences between the classes but in the end the understanding of the class differences is still lacking. Better ways to get more subtle understanding need yet to be analyzed.
As for classifier differences, Section 2, we looked at multiple classifiers and compared their (macro) F1-score performances. We observed that the classifiers perform differently but we still mostly lack the understanding why they differed. With our three experiments, average feature importance distributions, Subsection 3.3.1, average feature importance similarities, Subsection 3.3.2, and classifier prediction stabilities, Subsection 3.3.3, we observed the obvious significant differences with between our generative classifier and our two discriminative classifiers. The transferability of those experiments for other more subtle comparisons seems questionable. If we were to compare very similar classifiers of the same architecture with only different hyper-parameters, the outcomes of those experiments will probably be too similar to be of significant use. For the stability experiment, there also seems to be a problem with possibly changing the ground truth when we omit the most important feature from the instance. If we drop for example the swear word responsible for the abuse classification ground truth, we might now have a non-abusive instance with a wrong target. LIME might also not be suitable, in terms of being to unstable, in order to generate a global explanation from it.
Overall our results highly depended on the choice of data and how imbalanced the datasets are. Using a model-agnostic black-box approach for getting an understanding of the differences seems to be still rather difficult. Trying a more focused approach with only comparing two classifiers and using more model-specific methods might turn out to be more promising.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Nozza Debora, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, Manuela Sanguinetti, et al. 2019. Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *13th International Workshop on Semantic Evaluation*, pages 54–63. Association for Computational Linguistics.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Ian Covert, Scott Lundberg, and Su-In Lee. 2020. Understanding global feature contributions with additive importance measures. *Advances in Neural Information Processing Systems*, 33.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11.

Paula Fortuna, Juan Soler, and Leo Wanner. 2020. Toxic, hateful, offensive or abusive? what are we really classifying? an empirical analysis of hate speech datasets. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6786–6794.

Antigoni Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 12.

Scott Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Karl Pearson. 1901. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv preprint arXiv:1903.08983*.

A Appendix

Dataset id	**Classes**	**Number of Instances**	**Reference**
Davidson	hate-speech, offensive, neither	24783	(Davidson et al., 2017)
Waseem	none, racism, sexism	16907	(Waseem and Hovy, 2016)
Basile	none, hate-speech	12971	(Basile et al., 2019)
Zampieri	none, offensive	14100	(Zampieri et al., 2019)
Founta	normal, abusive, hateful, spam	99799	(Founta et al., 2018)

Table 1: Overview of the selected hate-speech datasets. The data instances in each dataset are English tweets from the Twitter social networking service. Each dataset covers diffent kinds of abuse.

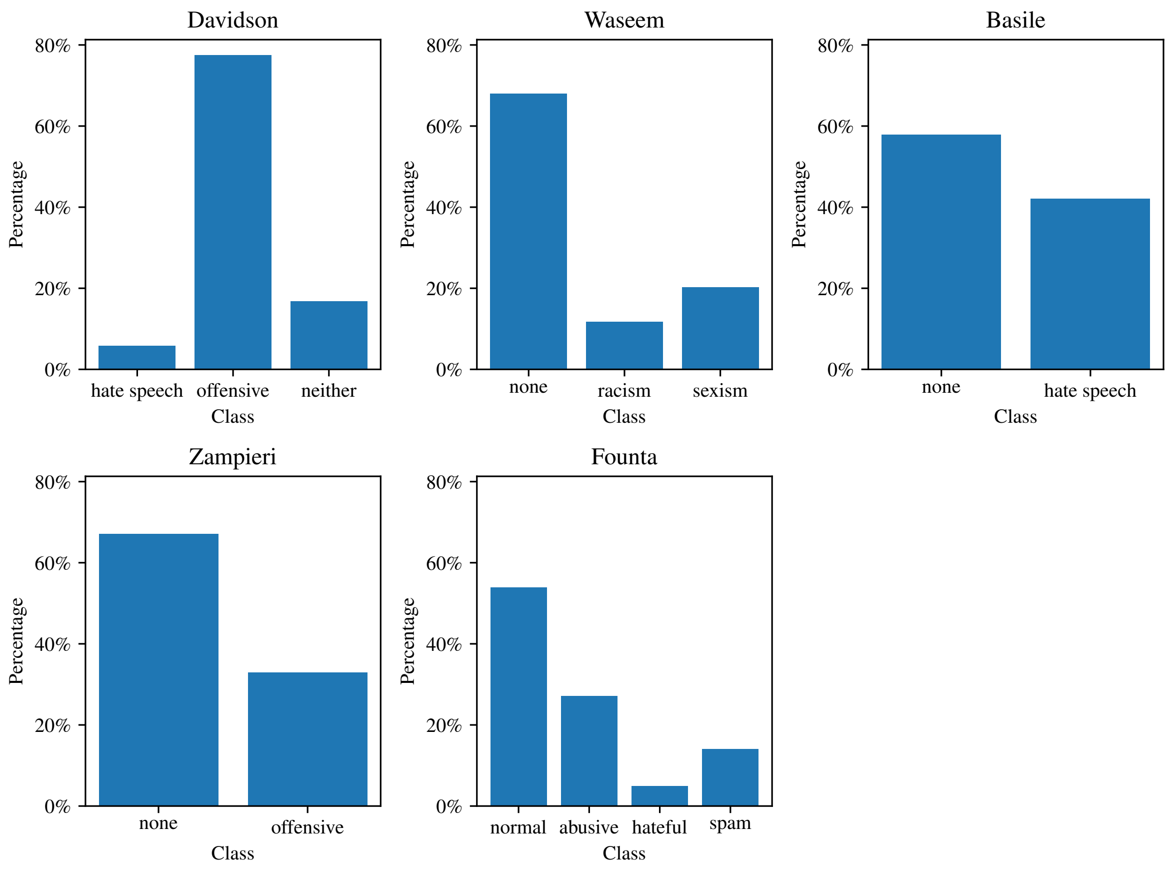

Figure 1: Class distributions of the datasets. There are class imbalances in all datasets but the Davidson dataset is imbalanced the most, containing only a relatively small amount of tweets of the none/neither class.

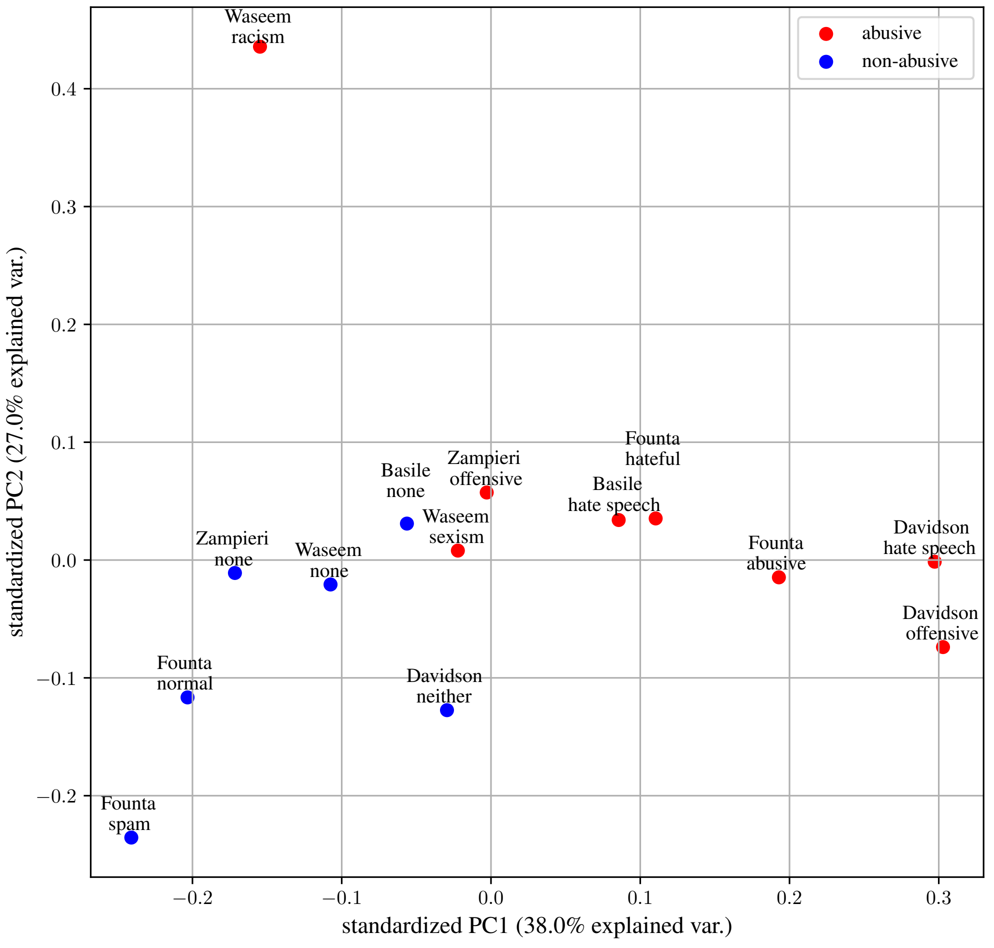

Figure 2: Inter-dataset class similarity PCA results. The labels have been categorized into the binary categories: *non-abusive* and *abusive*.

	Davidson	Waseem	Basile	Zampieri	Founta	Combined
LinearSVC	68.7	76.9	72.5	70.5	64.2	87.8
GaussianNB	53.6	41.4	64.0	55.6	45.7	72.7
ComplementNB	61.0	73.2	73.2	63.0	55.9	83.3
DecisionTreeClassifier	66.6	71.8	66.7	64.3	59.2	85.0
KNeighborsClassifier	55.2	65.4	66.6	60.8	48.2	73.8
RandomForestClassifier	56.6	73.7	71.7	66.4	57.1	86.9
MLPClassifier	68.3	72.9	69.8	66.9	61.6	85.1
DenseClassifier	67.1	74.3	70.7	68.0	63.2	86.6
LSTMClassifier	61.6	73.0	70.0	66.9	64.2	87.9
CNNClassifier	62.7	44.1	70.5	69.4	64.1	87.8

Table 2: Macro F1-scores for different classifier choices trained and evaluated on each dataset. The group of classifiers up top lists viable classifiers offered by scikit-learn, whereas the group below consists of simple neural networks implemented in TensorFlow.

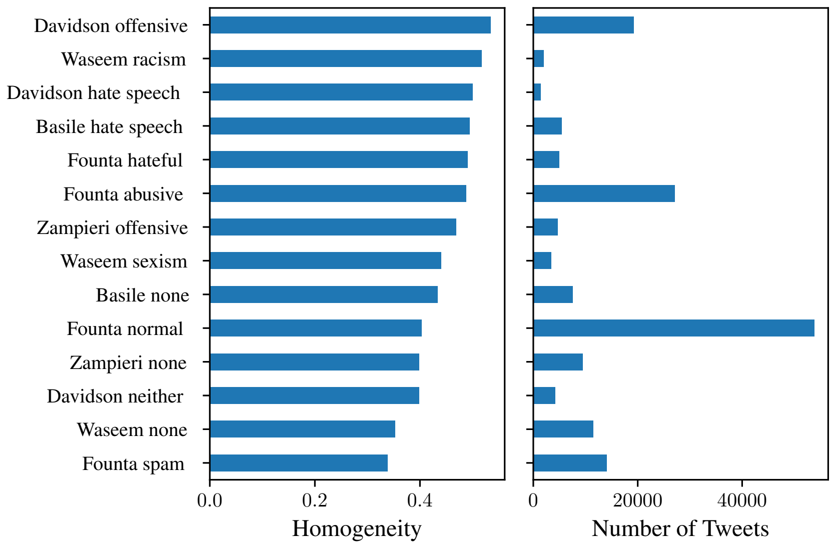

Figure 3: Intra-dataset class homogeneity and the number of tweets for each class.

	Davidson	Waseem	Basile	Zampieri	Founta	Combined
^{i}LinearSVC	97.2	71.0	67.2	56.9	89.6	85.0
^{c}LinearSVC	95.4	67.5	68.1	51.8	87.9	
^{i}GaussianNB	89.4	48.5	60.3	42.1	72.1	67.3
^{c}GaussianNB	80.3	42.9	48.7	33.4	71.2	
^{i}ComplementNB	94.5	68.7	68.7	43.4	83.6	80.5
^{c}ComplementNB	94.6	60.6	61.4	49.7	82.4	
^{i}DecisionTreeClassifier	96.6	67.0	61.2	50.8	86.5	82.0
^{c}DecisionTreeClassifier	94.4	61.6	63.6	48.7	85.0	
^{i}KNeighborsClassifier	92.5	57.5	62.3	41.3	70.3	67.2
^{c}KNeighborsClassifier	82.6	45.1	55.2	33.8	67.0	
^{i}RandomForestClassifier	95.3	69.6	65.0	47.9	88.7	83.8
^{c}RandomForestClassifier	95.0	58.1	58.3	39.1	88.8	
^{i}MLPClassifier	96.1	70.0	64.8	54.1	87.5	82.1
^{c}MLPClassifier	93.8	61.5	62.8	51.3	85.4	
^{i}DenseClassifier	96.5	68.2	64.6	54.4	87.9	84.1
^{c}DenseClassifier	96.0	67.0	66.7	54.7	86.5	
^{i}LSTMClassifier	96.6	70.1	65.7	54.6	88.7	85.6
^{c}LSTMClassifier	96.7	67.9	68.1	57.3	88.0	
^{i}CNNClassifier	95.6	71.1	66.5	57.1	89.1	85.6
^{c}CNNClassifier	96.9	68.5	68.0	56.9	88.4	

Table 3: Comparable F1-scores for the classifiers trained on the binarized datasets, with classes non-abusive and abusive. Superscript i indicates that the classifier is trained and evaluated on the respective dataset of the column. Superscript c indicates that the classifier is trained on the Combined dataset and evaluated for each dataset of the columns.

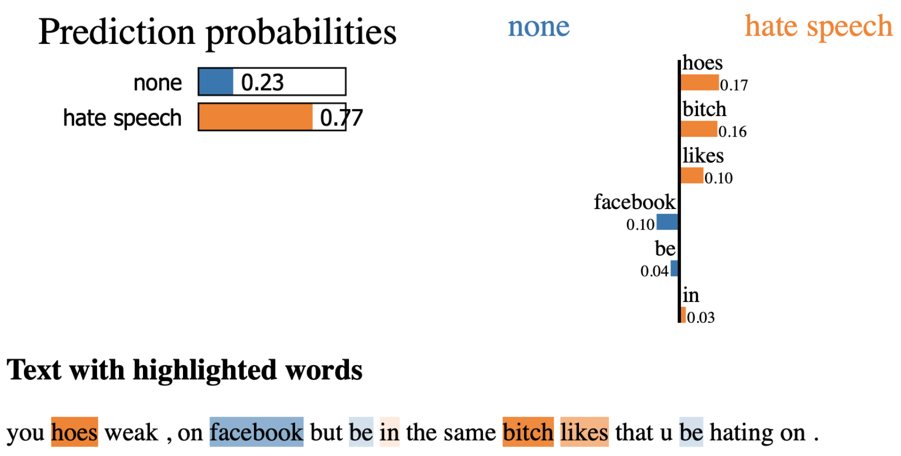

Figure 4: Example local explanation produced by LIME using the LinearSVC classifier with the Basile instance: *"You hoes weak, on Facebook but be in the same bitch likes that u be hating on."* The classifier correctly predicts the hate speech class. The upper right shows the feature importance scores of the 6 most important features i.e. words.

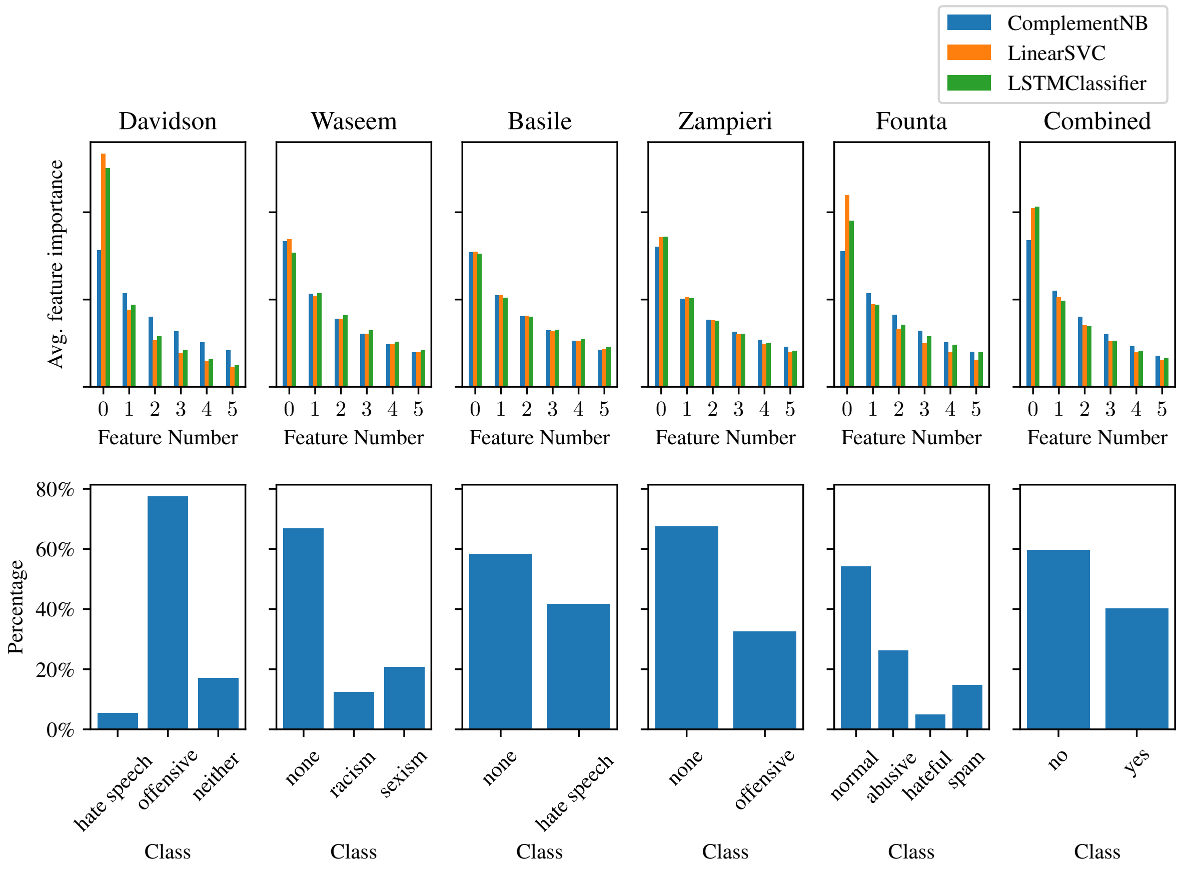

Figure 5: Average feature importance distributions on top with the datasets' class distributions below.

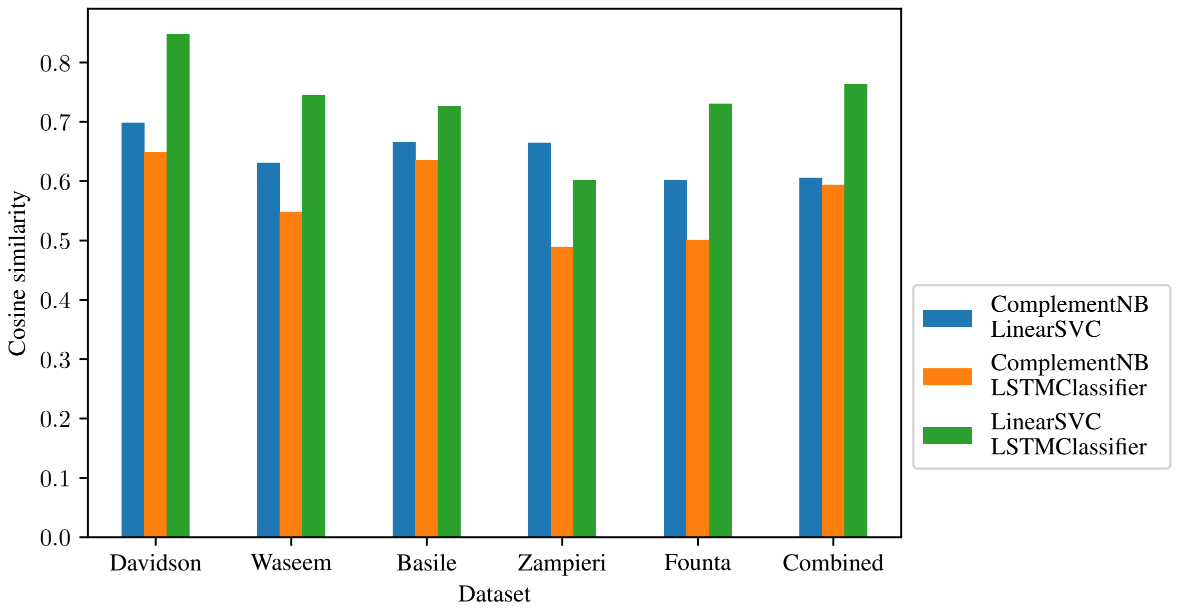

Figure 6: Average cosign similarity between feature importance scores of a classifier pair for each dataset.

		F1	F1 (omitted)	abs.	rel. %
Davidson	ComplementNB	58.35	46.17	-12.18	-20.87
	LinearSVC	66.23	44.35	-21.88	-33.03
	LSTMClassifier	60.88	39.72	-21.16	-34.76
Waseem	ComplementNB	73.80	60.33	-13.47	-18.26
	LinearSVC	76.95	56.03	-20.92	-27.18
	LSTMClassifier	72.93	54.53	-18.40	-25.23
Basile	ComplementNB	68.82	53.03	-15.79	-22.94
	LinearSVC	67.12	48.63	-18.49	-27.55
	LSTMClassifier	65.46	46.55	-18.91	-28.88
Zampieri	ComplementNB	40.77	13.64	-27.14	-66.55
	LinearSVC	55.92	22.33	-33.59	-60.07
	LSTMClassifier	53.52	25.62	-27.90	-52.13
Founta	ComplementNB	56.39	46.18	-10.21	-18.11
	LinearSVC	62.53	34.11	-28.43	-45.46
	LSTMClassifier	62.46	35.72	-26.74	-42.81
Combined	ComplementNB	82.09	69.11	-12.98	-15.81
	LinearSVC	85.91	48.52	-37.40	-43.53
	LSTMClassifier	86.54	48.69	-37.85	-43.74

Table 4: Stability of the macro F1-score for the selected classifiers over all datasets when omitting the most important feature i.e. word of an instance for reclassification.