Latent Feature-based Data Splits to Improve Generalisation Evaluation: A Hate Speech Detection Case Study

Maike Züfle[1] and **Verna Dankers**[1] and **Ivan Titov**[1,2]
[1]ILCC, University of Edinburgh
[2]ILLC, University of Amsterdam
m.s.zufle@sms.ed.ac.uk vdankers@ed.ac.uk ititov@inf.ed.ac.uk

Abstract

With the ever-growing presence of social media platforms comes the increased spread of harmful content and the need for robust hate speech detection systems. Such systems easily overfit to specific targets and keywords, and evaluating them without considering distribution shifts that might occur between train and test data overestimates their benefit. We challenge hate speech models via new train-test splits of existing datasets that rely on the clustering of models' hidden representations. We present two split variants (SUBSET-SUM-SPLIT and CLOSEST-SPLIT) that, when applied to two datasets using four pretrained models, reveal how models catastrophically fail on blind spots in the latent space. This result generalises when developing a split with one model and evaluating it on another. Our analysis suggests that there is no clear surface-level property of the data split that correlates with the decreased performance, which underscores that task difficulty is not always humanly interpretable. We recommend incorporating latent feature-based splits in model development and release two splits via the GenBench benchmark.[1]

1 Introduction

Developing generalisable hate speech detection systems is of utmost importance due to the environment in which they are deployed. Social media usage is rapidly increasing, and the detection of harmful content is challenged by non-standard language use, implicitly expressed hatred, a lack of consensus on what constitutes hateful content, and the lack of high-quality training data (Yin and Zubiaga, 2021a). When developing hate speech detection models in the lab, it is, therefore, vital to simulate evaluation scenarios requiring models to generalise outside the training context. 'In the wild', NLP models may encounter text from different periods

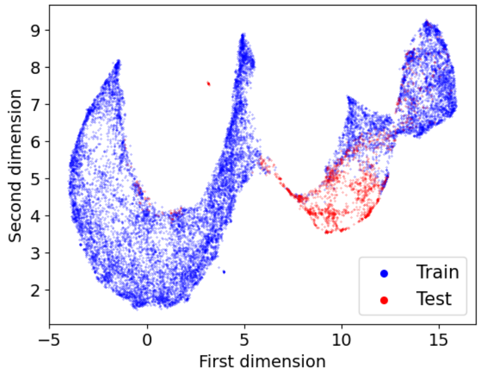


Figure 1: A UMAP projection of BERT's representations, showing the proposed train-test split, that is constructed by grouping clusters in the latent space.

(Lazaridou et al., 2021), authors (Huang and Paul, 2019) or dialects (Ziems et al., 2022), including unseen words (Elangovan et al., 2021) and words whose spelling changed or was obfuscated (Serra et al., 2017). Performing successfully on this data despite such distributional changes is called *out-of-distribution* (o.o.d.) generalisation.

How can the ability to generalise best be measured? Despite recent work illustrating that i.i.d. testing does not adequately reflect models' generalisability (e.g. Søgaard et al., 2021), evaluation using randomly sampled test sets is still the status quo (Rajpurkar et al., 2016; Wang et al., 2018, 2019; Muennighoff et al., 2023). Potentially, this is because obtaining and annotating new data is expensive, and it is hard to define what o.o.d. data is (Arora et al., 2021). For *humans*, properties like input length (Varis and Bojar, 2021) or spelling mistakes (Ebrahimi et al., 2018) might determine difficulty. But this need not be the same for *models*. Evaluating models using a notion of model-dependent difficulty is gaining some traction (e.g. Godbole and Jia, 2022) but still remains largely unexplored.

Contributing to that line of work, we propose a method that reuses existing datasets but splits them in a new way by relying on models' latent features.

[1]Our implementation is available at https://github.com/MaikeZuefle/Latent-Feature-Splits

Proceedings of the 1st GenBench Workshop on (Benchmarking) Generalisation in NLP, pages 112–129
December 6, 2023 ©2023 Association for Computational Linguistics

We cluster hidden representations using k-means and distribute clusters over the train and test set to create a data split. An illustrative example of such a split is shown in Fig. 1. We present two variants (SUBSET-SUM-SPLIT and CLOSEST-SPLIT). While this method is in principle applicable to any classification problem, we experiment with four language models and two hate speech datasets (that include Reddit, Twitter and Gab data). The results suggest that these splits approximate worst-case performance. Models fail catastrophically on the new test sets, while their performance on independent test data is on par with other systems trained on i.i.d. training sets. The difficulty is relatively stable across different models. We analyse the data splits through correlation analyses, and do not find one clear surface-level property of the data split to be predictive of split difficulty. This underscores that model-based difficulty can be quite elusive. We release two of our data splits for inclusion in the GenBench benchmark.

The remainder of this work is structured as follows: Section 2 elaborates on related work, followed by the introduction of the hate speech datasets (Section 3) and the proposed splitting method (Section 4). Section 5 presents model evaluation results, Section 6 analyses the splits in detail, and we conclude in Section 7. The GenBench eval card can be found in Appendix A.

2 Related Work

This section discusses related work on o.o.d. generalisation evaluation (Section 2.1), followed by a discussion on why generalisation is a persisting challenge in hate speech detection (Section 2.2).

2.1 Generalisation evaluation

It is now well-established within NLP that models with high or even human-like scores (e.g. Chowdhery et al., 2022) on i.i.d. splits do not generalise as robustly as the results would suggest. This has been demonstrated using synthetic data (i.a. Lake and Baroni, 2018; McCoy et al., 2019; Kim and Linzen, 2020) and for natural language tasks (i.a. Sinha et al., 2021; Søgaard et al., 2021; Razeghi et al., 2022). Alternative methods of evaluation have become more prominent, such as testing with different domains (e.g. Tan et al., 2019; Kamath et al., 2020; Yang et al., 2022) and adversarial testing, using both human-written (Kiela et al., 2021) and automatically generated adversarial examples (e.g. Zhang et al., 2020; Chen et al., 2019; Gururangan et al., 2018; Ebrahimi et al., 2018).

However, these types of evaluation require collecting or creating new data points, which is not always feasible for datasets that have been in use for years. Re-splitting existing datasets in a non-i.i.d. manner makes more efficient use of existing datasets, and, accordingly, new data splits have been developed, that typically use a feature of the input or the output to separate train from test examples. Splits that rely on the input use, for example, word overlap (Elangovan et al., 2021), linguistic structures (Søgaard, 2020), the timestamp (Lazaridou et al., 2021), or the context of words in the data (Keysers et al., 2019) to generate a split. Similarly, Broscheit et al. (2022) maximise the Wasserstein distances of train and test examples. Alternatively, one can evaluate generalisation using output-based non-i.i.d. splits: Naik et al. (2018) analyse the predictions of a model to find challenging phenomena, and Godbole and Jia (2022) re-split a dataset based on the predicted log-likelihood for each example.

The splitting method we propose relies neither on the discrete input tokens nor the output, but instead uses the internal representations of finetuned models.

2.2 Hate speech detection

With the rise of social media platforms, hate speech detection gained traction as a computational task (Jahan and Oussalah, 2023), leading to a wide range of benchmark datasets. Most of these datasets rely on data from social media platforms, such as Reddit (Qian et al., 2019; Vidgen et al., 2021), Twitter (ElSherief et al., 2021), Gab (Qian et al., 2019; Mathew et al., 2020), or Stormfront (de Gibert et al., 2018). This work is restricted to hate speech classification using a Reddit dataset (Qian et al., 2019) and a Twitter and Gab dataset (Mathew et al., 2020), which we will elaborate on in Section 3.

Recent advances in NLP such as the introduction of large language models have led to impressive results in hate speech detection (Fortuna and Nunes, 2018; Vidgen et al., 2019). Nonetheless, non-i.i.d. generalisation is a persisting challenge (Yin and Zubiaga, 2021b), because models tend to overfit to specific topics (Nejadgholi and Kiritchenko, 2020; Bourgeade et al., 2023), social media users (Arango et al., 2019), or keywords, such as slurs or pejorative terms (Dixon et al., 2018; Kennedy et al., 2020; Talat et al., 2018; Palmer et al., 2020; Kurrek et al., 2020). When such overt terms are missing, models often fail to detect hate speech (ElSherief et al., 2021). In response to these generalisation issues, recent works combine existing hate speech datasets (Fortuna et al., 2018; Salminen et al., 2020; Chiril et al., 2022; Bourgeade et al., 2023), which is a challenging task in itself considering the inconsistent definition of hate-speech across datasets (Nejadgholi and Kiritchenko, 2020).

Augmenting datasets or evaluating whether a model overfits to particular users or data sources requires annotated data. However, these characteristics are often unavailable due to privacy requirements or because the annotations were not included in the dataset release. Therefore, this work aims to find a data split that can evaluate generalisation without such annotations, relying instead only on a model's internal representations.

3 Data

We develop and evaluate our splitting method using the following two hate speech datasets.

3.1 Reddit

We use a widely used topic-generic Reddit dataset, proposed by Qian et al. (2019). The dataset includes 22,317 examples. Each example in the dataset is labelled as either *hate* (23.5%) or *noHate* (76.5%). The dataset was collected from ten different subreddits by retrieving potential hate speech posts using hate keywords taken from ElSherief et al. (2018). The hate keywords correspond roughly to the following categories: *archaic*, *class*, *disability*, *ethnicity*, *gender*, *nationality*, *religion*, and *sexual orientation*. The data is structured in conversations that consist of at most 20 comments by the same or different authors. These comments were manually annotated with *hate* or *noHate*, with each annotator assigned five conversations.

3.2 HateXplain

The second dataset is HateXplain (Mathew et al., 2020), which is also topic-generic and widely used. It contains 20,148 examples from Twitter and Gab. Posts from the combined collection were filtered based on a lexicon of hate keywords and phrases by Davidson et al. (2017); Mathew et al. (2019); Ousidhoum et al. (2019). The selected posts were then manually annotated. HateXplain examples are labelled as either *hateful* (31%), *offensive* (29%) or *normal* (40%), as proposed by Davidson et al.

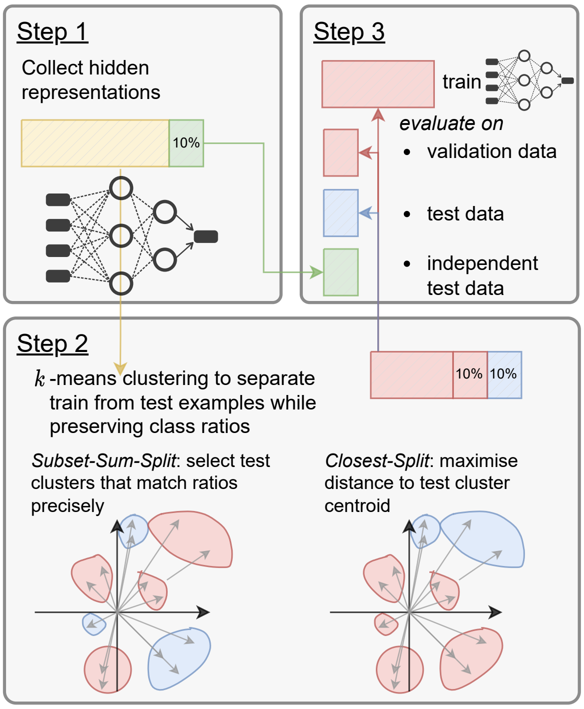


Figure 2: Overview of the proposed splitting method.

(2017). Offensive speech differs from hate speech in that it uses offensive terms without directing them against any person or group in particular. All offensive and hate examples are annotated with the community that they target. These communities include, among others, *Africans*, *Jewish People*, *Homosexuals* and *Women*, and we use them for further analysis of our data splits in Section 6.

4 Methodology

Our proposed splitting strategy, for which we introduce two variants, is detailed in Section 4.1. We evaluate our splits through comparisons to a random splitting baseline and on external test sets. We discuss the corresponding experimental setups in Section 4.2.

4.1 Constructing Data Splits

The construction of the data splits involves three steps, that are depicted in Fig. 2. In step 1, the method extracts the latent representations of inputs from a language model that was finetuned on the task using one of the hate speech datasets mentioned above. In step 2, the data is clustered based on these representations and clusters are assigned to either the train or the test set. In step 3, language models are then trained and evaluated on this new split. In addition to the obtained test set, the language models are also evaluated on independent test data, that was set aside for this purpose.[2]

The key idea behind the approach is that language models implicitly capture salient features of the input in their hidden representations, where inputs with similar properties are close together (Thompson and Mimno, 2020; Grootendorst, 2022). Assigning clusters to the train and test set thus accomplishes separation based on latent features, and by finetuning we ensure that the clusters separate examples based on *task-specific* features.

Obtaining Hidden Representations We finetune a language model for the given task, using the independent test data as validation set to optimise hyperparameters. We then obtain latent representations for each input example, leveraging the representation of the `[CLS]` token after the final layer as a representation of the input, as is commonly done (e.g. May et al., 2019; Qiao et al., 2019).

Since for high-dimensional data, distance metrics fail to accurately capture the concept of proximity (Beyer et al., 1999; Aggarwal et al., 2001) and tend to overly rely on individual dimensions (Timkey and van Schijndel, 2021) we conduct experiments with low-dimensional representations and full-dimensional ones. To this end, we either project the full representations into d_U-dimensional spaces using UMAP post-training (McInnes et al., 2020), or obtain d_B-dimensional representations by introducing a bottleneck in the model between the last hidden layer and the classification layer. The bottleneck is a linear layer that compresses the hidden representations, forcing the model to encode the most salient latent features into a low-dimensional space before classifying the examples.

Clustering and Splitting the Data Each representation from step 1 gives the position of an input example in the latent space. The examples are clustered in this space using the k-means algorithm (Lloyd, 1982).

Hyperparameters of the k-means clustering can be found in Table 3. After clustering, each cluster is assigned to either the train or the test set, keeping two constraints: A fixed test data size (we choose 10%) and train and test set need to have equal class distributions. Without equal class distributions, it would be unclear whether changes in performance are due to the increased difficulty of the test set, or the changes in label imbalance. A partition of the dataset that fulfils these constraints will be referred to as *target* in this work.

To reach the target test set, two algorithms, SUBSET-SUM-SPLIT and CLOSEST-SPLIT, are designed to decide how to split the clusters. Both algorithms lead to an under-representation of parts of the latent space in the model's training set, but whilst SUBSET-SUM-SPLIT might under-represent smaller, potentially distant pockets of the latent space, CLOSEST-SPLIT under-represents a single connected region. The algorithms are explained in detail below.

Method 1: SUBSET-SUM-SPLIT The constraints on the class and test ratios explained above, and the additional constraint of keeping whole clusters together can be described by the Subset Sum Problem (Kellerer et al., 2004). In this setting, the Subset Sum Problem can be modified to a *multidimensional* Subset Sum Problem: The multidimensional target consists of the number of desired test examples for each class in the dataset. The task is then to select a subset of the clusters, such that the number of examples for each class sums up to the desired target. To improve the chances of reaching the desired target, the Subset Sum Problem is solved for $k = 3$ to $k = 50$ clusters and the solution closest to the desired target using the smallest k is taken as the test set. If the closest solution does not match the exact target sum, examples from another randomly selected cluster are used to complete the test set. Note that the clusters in the test set do not necessarily lie close to each other in the latent space, as this is not a constraint for this algorithm.

Method 2: CLOSEST-SPLIT In contrast to the SUBSET-SUM-SPLIT, the CLOSEST-SPLIT aims to put as much distance as possible between the train and test clusters. This leads to an even bigger under-representation of parts of the latent space in the training set. Once the clusters have been computed, their centroids are calculated. The cluster that lies farthest away from all the other clusters is identified and added to the test set. If the size of the farthest cluster exceeds the target test set size, the next farthest cluster is taken instead. Cosine similarity between cluster centroids is used as the distance measure. Then *nearest neighbour* clustering with the cluster centroids is performed, as long as the size of the test set does not exceed the target size. When this nearest-neighbour clustering is finished,

[2] Note that the split thus only includes 90% of the data. Setting aside the 10% is for quality control of the models and could be omitted when future work applies our method.

individual examples that are closest to one of the test set centroids are added to the test set until the target size is reached. As for the SUBSET-SUM-SPLIT, the algorithm is performed for $k = 3$ to $k = 50$ clusters. k is selected such that the number of individual examples added is minimised.

4.2 Evaluating Splits' Difficulty

Models We use four transformer language models to obtain and evaluate the data splits: BERT-Base(-Cased) (Devlin et al., 2019), its smaller variant BERT-Medium (Turc et al., 2019; Bhargava et al., 2021), HateBERT (Caselli et al., 2021), a BERT-Base-Uncased model that was further pretrained on abusive Reddit data using the MLM objective, and RoBERTa-Base (Liu et al., 2019). From these models, we extract the full hidden representations, hidden representations via a bottleneck, for $d_B \in \{10, 50, 200\}$, and hidden representations post-processed using UMAP, for $d_U \in \{10, 50, 200\}$.

Model Evaluation Having obtained data splits based on four language models and hidden dimensions with different sizes, the first way of evaluating models is by finetuning the language models on their respective SUBSET-SUM-SPLIT and CLOSEST-SPLIT. The hyperparameters used for finetuning are listed in Table 4, Appendix B, and we estimate d_U and d_B by varying their values for the Reddit dataset. We compare the results obtained with the proposed data splits to a baseline split, which takes the same examples but splits them randomly while maintaining class proportions. Random splits are generated using three different seeds, and the proposed data splits are obtained with three different clustering seeds. For each data split involved, the models are trained with three seeds that determine the classifier's initialisation and the presentation order of the data. The results are averaged accordingly.

The evaluation metrics are accuracy and F1-scores. For the Reddit dataset, the F1-score is the score of the *hate* class, whereas for HateXplain, the F1-score is macro-averaged over the three classes.

To better understand the robustness of the results, we perform an additional set of experiments on the most challenging data splits observed, to answer the following questions:

1. *Is split difficulty driven by the input or by task-specific latent features?* For the Reddit data, we split the dataset based on task-agnostic hidden representations obtained from pretrained models to analyse whether task-specific representations (i.e. representations finetuned on the task) are needed to create challenging data splits.
2. *Do models trained on new splits perform on par with conventional models on independent data?* Using HateXplain, we test the finetuned models on the independent test data that was set aside earlier to ensure that the newly obtained train data is still informative enough for test data sampled according to the original distribution.
3. *Is the difficulty of the data splits model-independent?* We also examine whether a split obtained by the hidden representations of a specific model is also challenging for other models using HateXplain data.

model	Reddit Hate F1	HateXplain Macro F1
BERT-base	81.96 ± 0.5	66.0 ± 0.36
BERT-medium	81.58 ± 0.66	60.18 ± 0.42
HateBert	82.34 ± 0.59	66.25 ± 0.35
RoBERTa	82.15 ± 0.61	64.1 ± 0.9

Table 1: Results for the Reddit and HateXplain dataset on random splits using 90% of the data. Random splits are generated using three different seeds and models are trained with three initialisation seeds. Mean and standard errors are reported.

5 Results

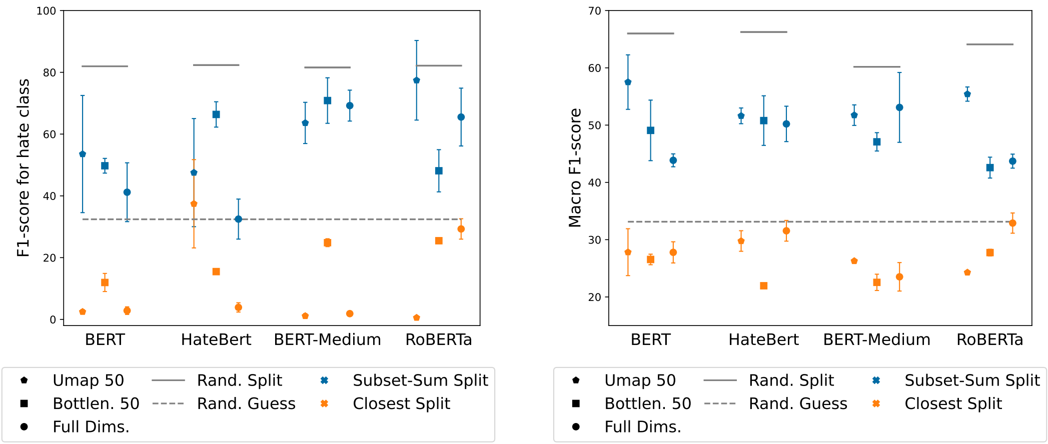
We now turn to evaluating models' performance on our newly proposed splits.

5.1 Performance on Challenging Splits

We compare the performance of models trained on a random split to models trained on the CLOSEST-SPLIT and SUBSET-SUM-SPLIT. The random split performances are presented in Table 1. For the binary Reddit dataset, performance on random splits is high for all four models with F1-scores for the hate class of around 82%. The performance on the three-way HateXplain dataset is comparably lower, with macro F1-scores of around 65%. For both datasets, these results are on par with (or surpass) baselines from prior work, upon which we elaborate in Appendix D.1.[3]

Hyperparameter Estimation For both splits, we conduct a hyperparameter estimation to select d_U

[3]Note that these results are obtained with 90% of the data as explained in Section 4.2. The reader is referred to Table 5 and Table 6 for accuracy results, results on 100% of the data and results on the standard split.



(a) Reddit Dataset (b) HateXplain Dataset

Figure 3: Performance of models trained on the SUBSET-SUM-SPLIT and CLOSEST-SPLIT . The errorbars show the standard error between cluster seeds. Horizontal lines indicate performance for models trained and tested on a random split.

and d_B using the Reddit dataset, for which the results are shown in Fig. 9, Appendix D.2. Across the board, all values considered challenge the models more than the random split, but full dimensions, $d_U = 50$ and $d_B = 50$ lead to a large decrease with relatively small variance between cluster seeds.

In addition to varying the dimensionalities, we consider using the models' pretrained representations (without further finetuning) to examine whether the latent features must be task-specific to challenge our models. Task-specific representations are, indeed, vital, as is shown in Fig. 8, Appendix D.2.

New Data Splits Reveal Catastrophic Failure Both SUBSET-SUM-SPLIT and CLOSEST-SPLIT lead to an under-representation of parts of the latent space in the model's training set and we hypothesised that this leads to a challenging data split. Indeed, the empirical results show significant performance drops when training models on these splits in comparison to random splits.

Fig. 3a shows the performance drops for the Reddit dataset. For the SUBSET-SUM-SPLIT, F1-scores for the hate class drop significantly for all four models, but with a high variation between different cluster seeds. For the CLOSEST-SPLIT, test set performance drops even further and more consistently without much variation between cluster seeds: F1-scores for the hate class are mostly between 0 and 25%. [4]

[4] These results are not specific to the examination of F1-scores; the same tendencies can be observed when looking at the accuracy (Appendix D.3).

Fig. 3b displays performances for HateXplain, which similarly shows a drop in performance for SUBSET-SUM-SPLIT and CLOSEST-SPLIT. CLOSEST-SPLIT leads to F1-scores that are on par with or below random guessing, resulting from drops of around 36%.

Overall, the CLOSEST-SPLIT is more challenging than the SUBSET-SUM-SPLIT. Moreover, the bottleneck-based splits generally lead to the most stable results, i.e., the variance between different cluster seeds is the lowest. In some cases performance drops below the random guessing baseline; this happens when a model fails to predict some class completely, defaulting instead to one of the other classes. In summary, the new splits lead to drastic performance drops for both datasets and across all four models.

5.2 Independent Test Set Performance

We now take the most challenging split observed (CLOSEST-SPLIT with $d_B = 50$) and further analyse the behaviour of models trained on this split for the HateXplain dataset, which is the most widely used dataset as well as the most challenging one.

From the results in Section 5.1 it is clear that CLOSEST-SPLIT reveals weaknesses in these models, since the models struggle to generalise to the split's test data. The question remains whether the test set obtained by the new splitting methods is harder or whether the new splitting method leads to very simple or perhaps even incomplete training sets, thereby preventing the models from learning the task. To this end, we evaluate the models trained on the training data obtained from

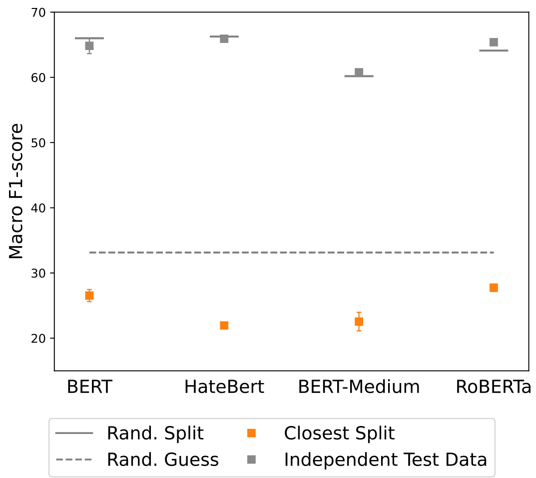


Figure 4: Performance of models trained on training data determined by the CLOSEST-SPLIT and evaluated on the test data of the CLOSEST-SPLIT and on independent test data (HateXplain dataset). Horizontal lines indicate performance for models trained and tested on a random split. Errorbars show the standard error between cluster seeds.

a CLOSEST-SPLIT on the 10% independent test data that was set aside earlier (Section 4.1). The results show that models achieve similar performance on the independent test data as the models trained and tested on random data, strengthening the hypothesis that CLOSEST-SPLIT training data is informative enough to learn the task. Results for these experiments are reported in Fig. 4.[5]

5.3 Cross-Model Generalisation

The previous results have shown that CLOSEST-SPLIT leads to challenging test sets. To show the robustness of these splits, we also examine whether these test sets are generally difficult or only for the model used to develop the split—i.e. we examine cross-model generalisation. The results of the cross-model evaluations can be seen in Fig. 5. They show that data splits developed using one model are indeed also challenging for other models, although the personalised splits are slightly more challenging. These results do not only strengthen the robustness of the challenging data split, but have also practical implications: The data-splitting pipeline only needs to be carried out with one model and multiple models can be assessed and compared with the same split.

[5]The validation accuracy for the models trained on CLOSEST-SPLIT is for most splits around 5 points higher than the accuracy on the validation set of the random data split—i.e. the models perform normally during training as suggested by the validation data.

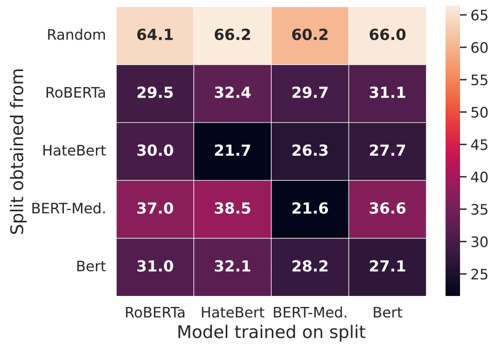


Figure 5: F1-scores for HateXplain on a CLOSEST-SPLIT ($d_B = 50$). Comparison of models trained on the data split obtained with their respective hidden representations (diagonal) and on data splits obtained from representations of other models.

6 Analysis

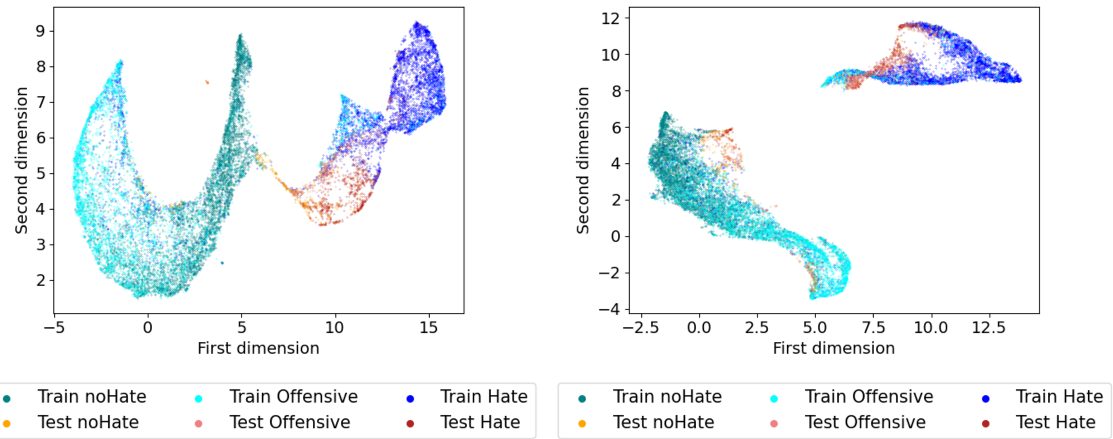
The performance of models deteriorates heavily when using the proposed splits. This section analyses the generated splits; first examining the surface-level properties of the resulting train and test sets, and then taking a closer look at two specific splits by visualising the datapoints in the train and test sets. Additionally, an analysis of the topics in the train and test sets can be found in Appendix E.2.

6.1 Correlation Analysis: Relating Splits' Features to Performance Drop

For the most challenging split variant, CLOSEST-SPLIT, we investigate the correlation of performance drops compared to the random splits (including three random splits with 0 drop) and surface-level properties of the data split. The properties' implementation is explained in detail in Appendix E.1. We firstly consider *task-agnostic* features: 1) the unigram overlap between the train and test set, 2) the input length in the test set and 3) the number of rare words in the test set.

Secondly, *task-specific* properties are computed: 1) The number of under-represented hate keywords from the lists used by the dataset's creators (see Section 3), 2) the number of under-represented target communities retrieved from the HateXplain annotations, and 3) a quantification of the distributional shift of data sources (Twitter and Gab are present in HateXplain) in the train and test set using the Kullback-Leibler Divergence of token distributions (Kullback and Leibler, 1951).

Table 2 presents the results of this analysis. For the Reddit Dataset, the only significant correlation (bold) is the number of under-represented key-



(a) RoBERTa's representations (b) BERT's representations

Figure 6: Hidden representations for tertiary classification using the CLOSEST-SPLIT for the HateXplain dataset.

	Feature	Reddit	HateXplain
task-agnostic	unigram overlap	0.24	**-0.51***
	sentence length	0.12	0.26
	# Rare words	0.13	0.44*
task-specific	# under-represented keywords	**0.47***	0.32*
	# under-represented targets	—	0.21
	KL-Div. data source	—	0.05

Table 2: Pearson correlation between data split properties and models' F1-score drops in comparison to random splits. Correlations with a p-value < 0.05 are marked with *. Some analysis methods are dataset-specific and cannot be computed for both datasets.

word categories in the training data. Task-agnostic features do not correlate with the decreased performance of models on the CLOSEST-SPLIT for the Reddit data. In contrast, for the HateXplain dataset, task-agnostic features do play a role: The biggest (negative) correlation can be observed for the unigram overlap (bold): The higher the unigram overlap between train and test set, the closer the performance is to the random split F1-score. Another smaller correlation exists concerning the number of rare words in the test set: The more rare words, the more challenging the split. Similar to the Reddit dataset, a significant, albeit weak, correlation exists between the decreased performance and the number of keyword categories that are under-represented in training data.

Taken together, these results suggest that the properties associated with performance drops differ from dataset to dataset. This implies that CLOSEST-SPLIT cannot easily be replicated based on task-specific or task-agnostic features. Using latent representations instead helps uncover weaknesses in models that are otherwise not easily identified.

6.2 Visualisation of Hidden Representations

We now take a closer look at two specific data splits for the HateXplain dataset by visualising their hidden representations. For this analysis, we select the CLOSEST-SPLITS obtained with representations with $d_B = 50$ for BERT and RoBERTa, which are more commonly used than HateBERT or BERT-medium. We make these splits available via the GenBench Collaborative Benchmarking Task.

The CLOSEST-SPLIT assigns clusters of hidden representations that are spatially close to the test set. While the clustering is conducted on high-dimensional representations, a 2-dimensional projection by UMAP (McInnes et al., 2020) can give an intuition about why these data splits are challenging. Fig. 6a shows RoBERTa's representations for the HateXplain dataset. A decision boundary can be observed, with mostly *offensive* examples on the left, *noHate* examples in the middle and *hate* examples on the right. Based on this illustration, the CLOSEST-SPLIT picks a pocket of (mixed) examples between the *noHate* (dark blue) and *hate* (dark green) regions to be the test set. This is mirrored in the F1-scores of the different classes. The *hate* test examples lie closest to the corresponding region, and the F1-score is the highest at 47.0. Similarly, for the *noHate* class, the F1-score is relatively high at 38.28. The offensive class, with test examples farther away, only has an F1-score of 11.88. The same phenomenon can be observed for a BERT-based CLOSEST-SPLIT (Fig. 6b). This suggests that the model overfits its decision boundaries to train set-specific features and, therefore, fails to predict the correct classes in the test set. Developing models using CLOSEST-SPLIT in addition to random splits might thus lead to models that are more robust to such overfitting.

7 Conclusion

Hate speech detection systems are prone to overfitting to specific targets of hate speech and specific keywords in the input, complicating the detection of more implicit hatred and harming the generalisability to unseen demographics. Yet, in addition to those *known* and *interpretable* vulnerabilities, systems may have less obvious weaknesses. The data splitting method we developed aims to highlight those. Our splitting method is based on the clustering of internal representations of finetuned models, thus making the splits task- and dataset-specific. We proposed two variants (SUBSET-SUM-SPLIT and CLOSEST-SPLIT) that differ in how they assign clusters to the train and test set.

The latter variant, in particular, led to consistent catastrophic drops in test set performance, when compared to a random split. Moreover, while each split was developed using the hidden representations from a specific model, we identified that this result generalises when developing the split using one model, and evaluating it using another. The analyses of the resulting data splits showed that the properties of the train and test sets differ from dataset to dataset. Since no property clearly correlates with decreased model performance for both datasets, CLOSEST-SPLIT cannot be easily replicated based on data splits' surface-level properties, and using latent representations is crucial to reveal the weaknesses we observed in the models.

We encourage future work to consider evaluations using the CLOSEST-SPLITS we release for HateXplain, in order to develop more robust systems, but also emphasise that even though our results were specific to hate speech detection, the methodology can be more widely applied. To challenge models beyond i.i.d. evaluation, we do not need costly data annotations. Instead, we can start by relying on systems' latent features to simulate train-test distribution shifts.

8 Limitations

We identify three main limitations of our work:

1. **The scope of our work**: the splitting methodology we developed can be applied to a wide range of tasks, but we only experimented with hate speech detection. Future work is required to confirm the method's wider applicability. Moreover, even though we aim to use the challenging split to improve generalisation, we have not yet made efforts in this direction.

2. **Generality of conclusions**: We experimented with a limited set of model architectures, all of which resemble one another in terms of their structure and the (pre-)training data used. Different models or training techniques could lead to less challenging splits, or splits with significantly different properties. At the same time, we did demonstrate that the split's difficulty is not model-specific (see Section 5.3), and observed that under variation of random seeds CLOSEST-SPLIT consistently leads to performance drops across four models and two datasets.

3. **Naturalness of the experimental setup**: we created an artificially partitioned data split and have no guarantee that the generalisation challenges that language models encounter when deployed in real-world scenarios resemble our splits. However, given that our approach simulated a worst-case scenario, demonstrated by catastrophic failure in performance, we are hopeful that models that are more robust to our train-test shift are also more robust to real-world variations in test data.

9 Ethics Statement

By its very nature, hate speech detection involves working closely with hurtful and offensive content. This can be difficult for researchers. However, considering the severe consequences when hate speech models fail on unseen data and people are confronted with harmful content, it is all the more important to improve the generalisation ability of models and protect others.

While our work intends to contribute to generalisation evaluation in a positive way, we do not recommend using our data splits as representative of generalisation behaviour 'in the wild', but recommend them for academic research instead. While standard and random splits often overestimate real-world performance, our splits are likely to underestimate it, and can in this way reveal real weaknesses. Our splits are designed to improve academic research on the robustness of language models and contribute to improving the generalisation ability for NLP tasks.

Prior to conducting work with potentially harmful hate speech data, this project obtained approval from the Research Ethics committee at the authors' local institution.

Acknowledgments

We thank Agostina Calabrese for helpful suggestions in the early stages of this project. VD is supported by the UKRI Centre for Doctoral Training in Natural Language Processing, funded by the UKRI (grant EP/S022481/1) and the University of Edinburgh, School of Informatics and School of Philosophy, Psychology & Language Sciences. IT is supported by the Dutch National Science Foundation (NWO Vici VI.C.212.053).

References

Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. 2001. On the surprising behavior of distance metrics in high dimensional space. In *Database Theory — ICDT 2001*, pages 420–434, Berlin, Heidelberg. Springer Berlin Heidelberg.

Aymé Arango, Jorge Pérez, and Barbara Poblete. 2019. Hate speech detection is not as easy as you may think: A closer look at model validation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 45–54, New York, NY, USA. Association for Computing Machinery.

Udit Arora, William Huang, and He He. 2021. Types of out-of-distribution texts and how to detect them. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10687–10701, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. 1999. When is "nearest neighbor" meaningful? In *Database Theory — ICDT'99*, pages 217–235, Berlin, Heidelberg. Springer Berlin Heidelberg.

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in NLI: Ways (not) to go beyond simple heuristics.

Tom Bourgeade, Patricia Chiril, Farah Benamara, and Véronique Moriceau. 2023. What did you learn to hate? A topic-oriented analysis of generalization in hate speech detection. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3495–3508, Dubrovnik, Croatia. Association for Computational Linguistics.

Samuel Broscheit, Quynh Do, and Judith Gaspers. 2022. Distributionally robust finetuning BERT for covariate drift in spoken language understanding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1970–1985, Dublin, Ireland. Association for Computational Linguistics.

Marc Brysbaert and Boris New. 2009. Moving beyond Kučera and Francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for american english. *Behavior research methods*, 41:977–90.

Tommaso Caselli, Valerio Basile, Jelena Mitrović, and Michael Granitzer. 2021. HateBERT: Retraining BERT for abusive language detection in English. In *Proceedings of the 5th Workshop on Online Abuse and Harms (WOAH 2021)*, pages 17–25, Online. Association for Computational Linguistics.

Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. 2019. CODAH: An adversarially-authored question answering dataset for common sense. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 63–69, Minneapolis, USA. Association for Computational Linguistics.

Patricia Chiril, Endang Wahyu Pamungkas, Farah Benamara, Véronique Moriceau, and Viviana Patti. 2022. Emotionally informed hate speech detection: A multi-target perspective. *Cognitive Computation*, 14(1):322–352.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *Proceedings of the International AAAI Conference on Web and Social Media*, 11(1):512–515.

Ona de Gibert, Naiara Pérez, Aitor García Pablos, and Montse Cuadros. 2018. Hate speech dataset from a white supremacy forum. *CoRR*, abs/1809.04444.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '18, page 67–73, New York, NY, USA. Association for Computing Machinery.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Aparna Elangovan, Jiayuan He, and Karin Verspoor. 2021. Memorization vs. generalization : Quantifying data leakage in NLP performance evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1325–1335, Online. Association for Computational Linguistics.

Mai ElSherief, Shirin Nilizadeh, Dana Nguyen, Giovanni Vigna, and Elizabeth Belding. 2018. Peer to peer hate: Hate speech instigators and their targets. *Proceedings of the International AAAI Conference on Web and Social Media*, 12(1).

Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. Latent hatred: A benchmark for understanding implicit hate speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Paula Fortuna, Ilaria Bonavita, and S érgio Nunes. 2018. Merging datasets for hate speech classification in Italian. *EVALITA Evaluation of NLP and Speech Tools for Italian*, 12:218.

Paula Fortuna and Sérgio Nunes. 2018. A survey on automatic detection of hate speech in text. *ACM Comput. Surv.*, 51(4).

Ameya Godbole and Robin Jia. 2022. Benchmarking long-tail generalization with likelihood splits.

Maarten Grootendorst. 2022. BERTopic: Neural topic modeling with a class-based tf-idf procedure.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Xiaolei Huang and Michael J. Paul. 2019. Neural user factor adaptation for text classification: Learning to generalize across author demographics. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 136–146, Minneapolis, Minnesota. Association for Computational Linguistics.

Md Saroar Jahan and Mourad Oussalah. 2023. A systematic review of hate speech automatic detection using natural language processing. *Neurocomputing*, 546:126232.

Amita Kamath, Robin Jia, and Percy Liang. 2020. Selective question answering under domain shift. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5684–5696, Online. Association for Computational Linguistics.

Hans Kellerer, Ulrich Pferschy, and David Pisinger. 2004. *The Subset Sum Problem*, pages 73–115. Springer Berlin Heidelberg, Berlin, Heidelberg.

Brendan Kennedy, Xisen Jin, Aida Mostafazadeh Davani, Morteza Dehghani, and Xiang Ren. 2020. Contextualizing hate speech classifiers with post-hoc explanation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5435–5442, Online. Association for Computational Linguistics.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2019. Measuring compositional generalization: A comprehensive method on realistic data. *CoRR*, abs/1912.09713.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, Online. Association for Computational Linguistics.

Najoung Kim and Tal Linzen. 2020. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language*

Processing (EMNLP), pages 9087–9105, Online. Association for Computational Linguistics.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Jana Kurrek, Haji Mohammad Saleem, and Derek Ruths. 2020. Towards a comprehensive taxonomy and large-scale annotated corpus for online slur usage. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 138–149, Online. Association for Computational Linguistics.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *35th International Conference on Machine Learning, ICML 2018*, pages 4487–4499. International Machine Learning Society (IMLS).

Angeliki Lazaridou, Adhi Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder, Dani Yogatama, Kris Cao, Susannah Young, and Phil Blunsom. 2021. Mind the gap: Assessing temporal generalization in neural language models. In *Advances in Neural Information Processing Systems*, volume 34, pages 29348–29363. Curran Associates, Inc.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

S. Lloyd. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.

Binny Mathew, Ritam Dutt, Pawan Goyal, and Animesh Mukherjee. 2019. Spread of hate speech in online social media. In *Proceedings of the 10th ACM Conference on Web Science*, WebSci '19, page 173–182, New York, NY, USA. Association for Computing Machinery.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. HateXplain: A benchmark dataset for explainable hate speech detection. *CoRR*, abs/2012.10289.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, Minneapolis, Minnesota. Association for Computational Linguistics.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Leland McInnes, John Healy, and James Melville. 2020. Umap: Uniform manifold approximation and projection for dimension reduction.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Isar Nejadgholi and Svetlana Kiritchenko. 2020. On cross-dataset generalization in automatic detection of online abuse. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 173–183, Online. Association for Computational Linguistics.

Nedjma Ousidhoum, Zizheng Lin, Hongming Zhang, Yangqiu Song, and Dit-Yan Yeung. 2019. Multilingual and multi-aspect hate speech analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4675–4684, Hong Kong, China. Association for Computational Linguistics.

Alexis Palmer, Christine Carr, Melissa Robinson, and Jordan Sanders. 2020. COLD: Annotation scheme and evaluation data set for complex offensive language in english. *Journal for Language Technology and Computational Linguistics*, 34(1):1–28.

Jing Qian, Anna Bethke, Yinyin Liu, Elizabeth Belding, and William Yang Wang. 2019. A benchmark dataset for learning to intervene in online hate speech. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4755–4764, Hong Kong, China. Association for Computational Linguistics.

Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the behaviors of BERT in ranking. *CoRR*, abs/1904.07531.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot numerical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 840–854, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Joni Salminen, Maximilian Hopf, Shammur A. Chowdhury, Soon-gyo Jung, Hind Almerekhi, and Bernard J. Jansen. 2020. Developing an online hate classifier for multiple social media platforms. *Human-centric Computing and Information Sciences*, 10(1):1.

Joan Serra, Ilias Leontiadis, Dimitris Spathis, Gianluca Stringhini, Jeremy Blackburn, and Athena Vakali. 2017. Class-based prediction errors to detect hate speech with out-of-vocabulary words. In *Proceedings of the first workshop on abusive language online*, pages 36–40.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Anders Søgaard. 2020. Some languages seem easier to parse because their treebanks leak. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2765–2770, Online. Association for Computational Linguistics.

Anders Søgaard, Sebastian Ebert, Jasmijn Bastings, and Katja Filippova. 2021. We need to talk about random splits. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1823–1832, Online. Association for Computational Linguistics.

Robyn Speer. 2022. rspeer/wordfreq: v3.0.

Zeerak Talat, James Thorne, and Joachim Bingel. 2018. *Bridging the Gaps: Multi Task Learning for Domain Transfer of Hate Speech Detection*, pages 29–55. Springer International Publishing, Cham.

Ming Tan, Yang Yu, Haoyu Wang, Dakuo Wang, Saloni Potdar, Shiyu Chang, and Mo Yu. 2019. Out-of-domain detection for low-resource text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3566–3572, Hong Kong, China. Association for Computational Linguistics.

Laure Thompson and David Mimno. 2020. Topic modeling with contextualized word representation clusters. *CoRR*, abs/2010.12626.

William Timkey and Marten van Schijndel. 2021. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4527–4546, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962.

Dusan Varis and Ondřej Bojar. 2021. Sequence length is a domain: Length-based overfitting in transformer models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8246–8257, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bertie Vidgen, Alex Harris, Dong Nguyen, Rebekah Tromble, Scott Hale, and Helen Margetts. 2019. Challenges and frontiers in abusive content detection. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 80–93, Florence, Italy. Association for Computational Linguistics.

Bertie Vidgen, Dong Nguyen, Helen Margetts, Patricia Rossini, and Rebekah Tromble. 2021. Introducing CAD: the contextual abuse dataset. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2289–2303, Online. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. *CoRR*, abs/1905.00537.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461.

Linyi Yang, Shuibai Zhang, Libo Qin, Yafu Li, Yidong Wang, Hanmeng Liu, Jindong Wang, Xing Xie, and Yue Zhang. 2022. GLUE-X: Evaluating natural language understanding models from an out-of-distribution generalization perspective.

Wenjie Yin and Arkaitz Zubiaga. 2021a. Towards generalisable hate speech detection: a review on obstacles and solutions. *PeerJ Computer Science*, 7:e598.

Wenjie Yin and Arkaitz Zubiaga. 2021b. Towards generalisable hate speech detection: a review on obstacles and solutions. *CoRR*, abs/2102.08886.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Trans. Intell. Syst. Technol.*, 11(3).

Caleb Ziems, Jiaao Chen, Camille Harris, Jessica Anderson, and Diyi Yang. 2022. VALUE: Understanding dialect disparity in NLU. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3701–3720, Dublin, Ireland. Association for Computational Linguistics.

A GenBench Eval Card

Motivation			
Practical	*Cognitive*	*Intrinsic* □	*Fairness*

Generalisation type					
Compo-sitional	*Structural*	*Cross Task*	*Cross Language*	*Cross Domain*	*Robust-ness* □

Shift type			
Covariate □	*Label*	*Full*	*Assumed*

Shift source			
Naturally occuring	*Partitioned natural* □	*Generated shift*	*Fully generated*

Shift locus			
Train–test	*Finetune train–test* □	*Pretrain–train*	*Pretrain–test*

Our work proposes a data split that evaluates the generalisation ability of hate speech detection models. Our motivation is an **intrinsic** one, we aim to understand better what kind of data is most challenging for hate speech detection models.

We focus on testing the **robustness** of such models, especially when it comes to out-of-distribution (o.o.d.) generalisation. However, it is not straightforward to define and detect o.o.d. data (Arora et al., 2021). Moreover, data properties that might seem challenging for humans (Varis and Bojar, 2021; Ebrahimi et al., 2018) might not be equally challenging for models or rely on costly annotations (Arango et al., 2019; Nejadgholi and Kiritchenko, 2020; Bourgeade et al., 2023).

Therefore, we create a train test split by only relying on a model's hidden representations. This **partitioned natural** splitting method yields a **covariate shift**, since we re-split existing data sets. The resulting train test splits indeed challenge hate speech detection models in a **finetune train-test** locus.

B Clustering

Our proposed data split creates a train-test split by assigning whole clusters of latent representations to either the train or the test set. We use k-means clustering (Lloyd, 1982) to perform the clustering. The used hyperparamters can be found below.

Parameter	Value
n clusters	3-50
n initializations with different centroids	10
max. iterations for a run	300
random state	42, 62, 82
algorithm	LLoyd

Table 3: K-Means hyperparameters

C Language Models

We use four transformer language models to obtain and evaluate the data splits: BERT-Base(-Cased) (Devlin et al., 2019), its smaller variant BERT-Medium (Turc et al., 2019; Bhargava et al., 2021), HateBERT (Caselli et al., 2021), a BERT-Base-Uncased model that was further pretrained on abusive Reddit data using the MLM objective, and RoBERTa-Base (Liu et al., 2019). The hyperparamters for finetuning can be found below. They are generally adopted from the finetuned models from Caselli et al. (2021), but due to computational restrictions, the models had to be trained with reduced batch sizes. To compensate for this, models were trained with more epochs with the option of early stopping.

Hyperparameter	Value
batch size	4 (biggest possible)
early stopping	after 5 epochs
maximum epochs	10 (20 for the larger RoBERTa models)
optimizer	AdamW
learning rate	2e-5
adam epsilon	1e-8
scheduling	linear schedule with warmup
warm up steps	0
random seeds	42, 55, 83
max. sequence length	512

Table 4: Hyperparameters for finetuning the language models are adopted from the finetuned models from Caselli et al. (2021).

D Detailed Results

The following section presents detailed results including baselines, hyperparameter selections and further results.

D.1 Baselines

We compare the performance of models trained on our proposed data splits (CLOSEST-SPLIT and SUBSET-SUM-SPLIT) to a random split. We obtain random splits not only from 100% of the data but also from 90% of the data. This is necessary to compare the random split to the CLOSEST-SPLIT and SUBSET-SUM-SPLIT, as these use only 90% of the data. The random split performances are presented below.

model	valid acc.	test acc.	hate f1
SVM*	–	–	75.7
RNN*	–	–	77.5
BERT-base (100%)	94.6 ± 0.21	91.55 ± 0.13	82.24 ± 0.34
BERT-base (90%)	91.69 ± 0.07	91.25 ± 0.11	81.96 ± 0.5
BERT-med. (100%)	94.3 ± 0.23	91.63 ± 0.2	82.27 ± 0.45
BERT-med. (90%)	91.84 ± 0.07	91.2 ± 0.15	81.58 ± 0.66
HateBert (100%)	94.12 ± 0.06	91.87 ± 0.16	82.72 ± 0.38
HateBert (90%)	92.02 ± 0.07	91.51 ± 0.13	82.34 ± 0.59
RoBERTa (100%)	94.4 ± 0.12	91.67 ± 0.2	82.5 ± 0.49
RoBERTa (90%)	91.8 ± 0.09	91.37 ± 0.16	82.15 ± 0.61

Table 5: Results for the Reddit dataset on random splits using 100% and 90% of the data. Random splits are generated using three different seeds and models are trained with three initialisation seeds; mean and standard errors are reported. Results marked with * are taken from Qian et al. (2019).

split	model	valid acc	test acc	Macro f1
stand.	BERT-base *	–	69.0	67.4
stand.	BERT-base	67.45 ± 0.36	68.38 ± 0.35	66.06 ± 0.44
	BERT-med.	63.93 ± 1.2	64.58 ± 0.99	62.32 ± 1.45
	HateBert	68.12 ± 0.16	68.0 ± 0.37	65.97 ± 0.36
	RoBERTa	67.32 ± 0.3	67.83 ± 0.42	65.98 ± 0.26
rand.	BERT-base	67.66 ± 0.31	68.25 ± 0.28	66.0 ± 0.36
	BERT-med.	62.46 ± 0.49	62.85 ± 0.42	60.18 ± 0.42
	HateBert	67.91 ± 0.32	68.51 ± 0.28	66.25 ± 0.35
	RoBERTa	66.45 ± 0.51	66.4 ± 0.56	64.1 ± 0.9

Table 6: Results for the HateXplain dataset on the standard (stand.) split and on random (rand.) splits using 90% of the data. Random splits are generated using three different seeds and models are trained with three initialisation seeds; mean and standard errors are reported. Results marked with * are taken from Mathew et al. (2020).

D.2 Hyperparameter Selection for Proposed Split

We analyse the effects of two hyperparameters. First, we analyse whether task-specific, finetuned representations are needed for challenging data splits or whether task-agnostic, pretrained representations also lead to difficult splits. The results can be found in Fig. 7 and Fig. 8. The second hyperparameter we analyse is the dimensionality of the representations, as displayed in Fig. 9.

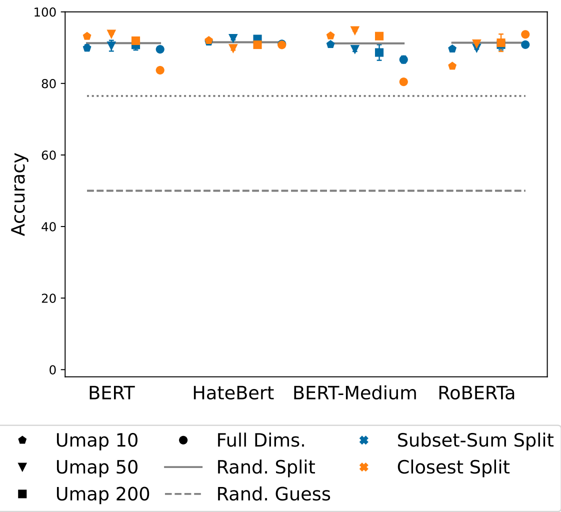


Figure 7: Performance of language models trained on the **pretrained** SUBSET-SUM-SPLIT and pretrained CLOSEST-SPLIT of the Reddit data. The errorbars show the standard error between cluster seeds.

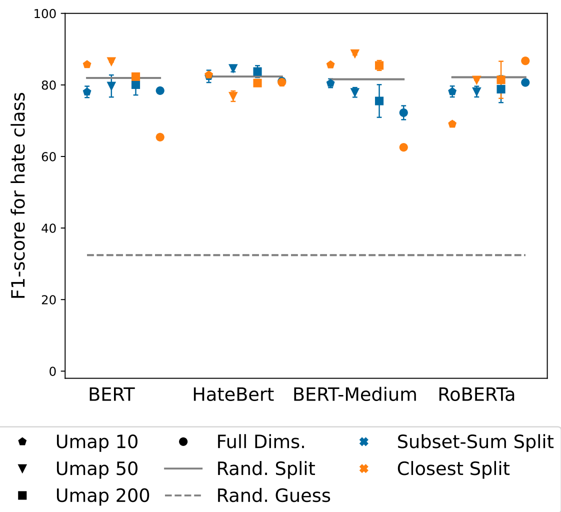
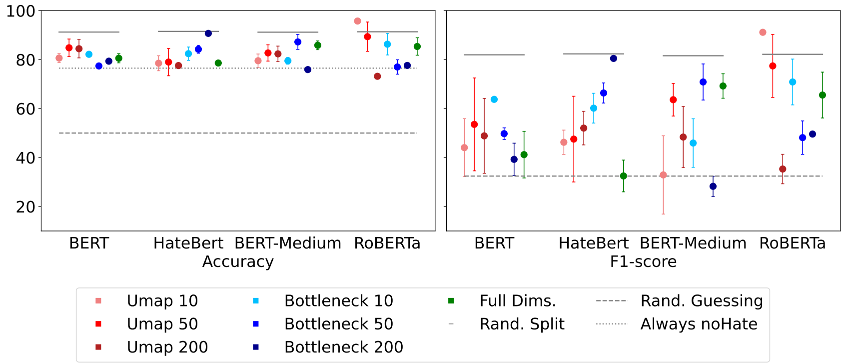
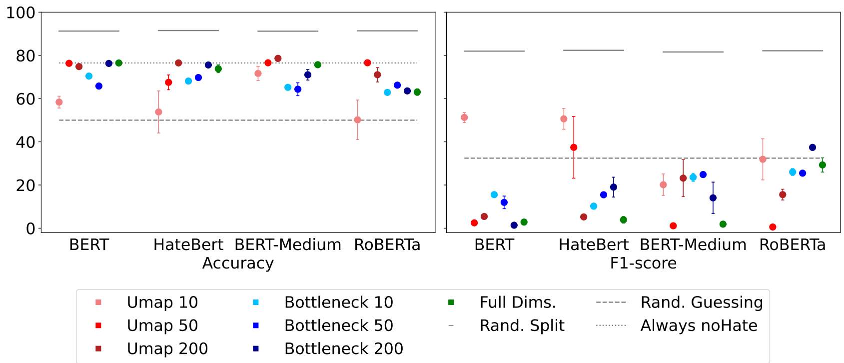


Figure 8: Performance of language models trained on the **pretrained** SUBSET-SUM-SPLIT and pretrained closest split of the Reddit data. The errorbars show the standard error between cluster seeds.



(a) SUBSET-SUM-SPLIT



(b) CLOSEST-SPLIT

Figure 9: Performance of language models trained on the SUBSET-SUM-SPLIT and CLOSEST-SPLIT of the Reddit dataset. Random split performance, indicated by the solid horizontal lines, is used as a baseline. The error bars show the standard error between cluster seeds.

D.3 Subset-Sum and Closest Split

SUBSET-SUM-SPLIT and CLOSEST-SPLIT both lead to a decreased performance. The performance on the Reddit dataset in terms of accuracy can be found below in Fig. 10.

The HateXplain accuracy can be found in Fig. 11. For both datasets, models fail to predict some class completely, defaulting instead to one of the other classes. Note that HateXplain is a balanced dataset, while Reddit is highly unbalanced (75% noHate).

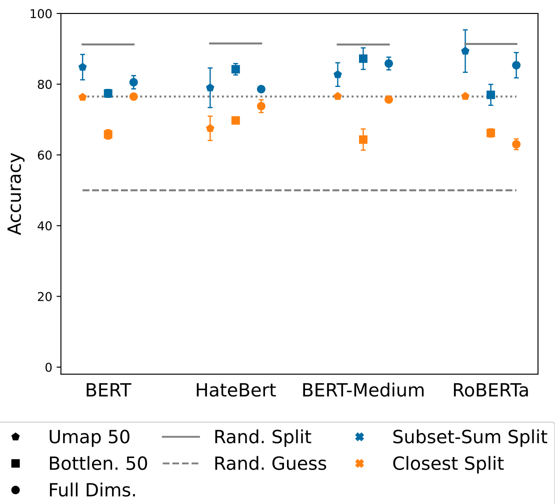


Figure 10: Performance of language models trained on the SUBSET-SUM-SPLIT and CLOSEST-SPLIT of the Reddit data. The errorbars show the standard error between cluster seeds.

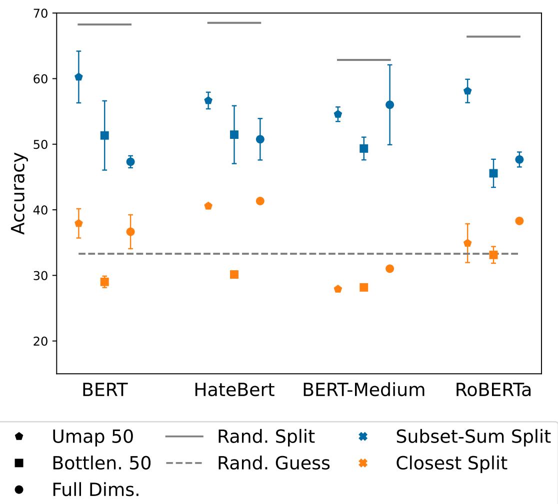


Figure 11: Performance of language models trained on the SUBSET-SUM-SPLIT and CLOSEST-SPLIT of the HateXplain data. The errorbars show the standard error between cluster seeds.

E Analysis

E.1 Data split properties

This section presents a detailed description of the features used for the analysis in Section 6. The following task-agnostic features are included in the analysis:

Unigram Overlap Following the word overlap algorithm in Elangovan et al. (2021), the word overlap o_i for a given test example $test_i$ is the word overlap with the most similar training example $train_k$. The word overlap of the whole test set is then the average over the word overlap of the test examples o_i. For this computation, examples are represented as a vector with unigram counts (ignoring stopwords), and similarity is computed as the cosine similarity.

Sentence Length in the Test Set We use the average length of input examples in the test set in terms of characters.

Number of Rare Words in the Test Set Rare words are defined following the definition of Godbole and Jia (2022): Rare words are words that are not common (i.e. occur at most once per million words) and are not misspelled (i.e. appear in the word list of common words[6]). For word frequency statistics, Godbole and Jia (2022) rely on Brysbaert and New (2009). We use the word frequencies more recently collected by Speer (2022) instead.

Moreover, we compare the dropped performance on the proposed data splits to the following task-specific features:

Number of under-represented keywords in the train set The Reddit and HateXplain dataset have been created by filtering posts based on hate keywords by simply string-matching the posts with the keywords. These keywords can be understood as hate speech categories. We calculate the number of hate speech categories that are under-represented in the train set, i.e. have less than 50% of their occurrences in the train set. Keywords that occur in less than 3% of the data set are excluded.

Number of under-represented targets in the train set This method aims to analyse the different targets of hate speech. For the HateXplain dataset, these targets are annotated as explained in Section 3. We calculate the number of under-represented targets in the train set using the same concept as for the under-represented keywords.

Difference of the data source distribution in the train and test set As described in Section 3, the HateXplain dataset consists of two data sources, Gab (46%) and Twitter (54%). We calculate the distributional shift between the data source distribution in the train and test set. The Kullback-Leibler Divergence (Kullback and Leibler, 1951) is calculated for the two data sources in the dataset and then the average is taken over both classes, weighted by the occurrence of the class in the dataset. Since there is no upper bound for the KL Divergence, it is scaled to be between 0 and 1 by the function

$$f(x) = 1 - e^{-x}. \quad (1)$$

E.2 Topic analysis

Set	Class	Topics RoBERTa
Train	Hate	nigger, kike, white, jews
	Offens.	retarded, bitch, white, ghetto
	noHate	white, people, women, raped
Test	Hate	jews, faggot, muslim, white
	Offens.	faggot, jews, nigger, white
	noHate	white, jews, people, retarded

Table 7: Top 4 topics for different classes in the HateXplain dataset. The topics are obtained from train and test sets of the Closest Split with latent representations from RoBERTA.

We extract topics for each class in the train and test sets using c-TF-IDF (Grootendorst, 2022).

As an example, Table 7 summarises the topics with the highest c-TF-IDF scores. There seems to be a tendency for the offensive and noHate classes to have different topics in the train and test sets, while the hate class is more consistent across the split. A manual analysis of cluster topics for all cluster splits did not lead to conclusive results: Topics are not clearly separated across all classes between the train and test sets. Many of the topics found by c-TF-IDF seem to coincide with the targets that were annotated, and used for the analysis in the previous section. No strong correlation between targets and performance was observed then, which strengthens the result that different targets in the train and test sets are not the reason for the decreased performance.

[6] https://github.com/dwyl/english-words