Ling 83600 Final Project
Identifying and Classifying Hate Speech

Livia Clarete and **Yaffa Tziporah** and **Rose Wong**
Department of Computational Linguistics
CUNY Graduate Center

Abstract

This research aims to inspire change and encourage reappraisal of potentially harmful language by identifying written hate speech based on social media Twitter data using hierarchical classification (HC). HC has historically been used in the field of genetics, and recently on e-commerce product classifications. It has also shown to be effective in sentiment analysis, emotion recognition, and hate speech detection in English, and Indonesian. Previous research has found that classification methods, specifically HC, are effective in detecting hate speech, but annotated data is often only available in English. This study uses an annotated dataset of 5,668 tweets in Brazilian Portuguese Tweets classified using a binary label of 'hate' vs 'no-hate' and a multi-label framework of 81 categories.

Three sets of experiments were completed. The first experiment replicates the findings of Fortuna et al. This was successful. The second set of experiments was to explore flat classification using four different algorithms. These results were then used as the baseline for comparing the results of hierarchical classification using Extreme Text classification models. The results of the hierarchical work are not yet successful, with key metrics lower than baseline from flat classifications. Future work would include improving models hyperparameters, and gathering a larger dataset to improve the results.

1 Introduction

Hate speech can be defined as any form of expression through which speakers intend to vilify, humiliate, or incite hatred against a group or a class of persons on the basis of race, religion, skin color sexual identity, gender identity, ethnicity, disability, or national origin[1]. These forms of expression can be speech, conduct, writing, behavior, or expression that may incite violence or prejudicial action against or by a particular individual or group, or because it disparages or intimidates a particular individual or group.

[1] Kenneth Ward, Free Speech and the Development of Liberal Virtues: An Examination of the Controversies Involving Flag-Burning and Hate Speech, 52 U. Miami L. Rev. 733 (1998)

The objective of this research is to identify written hate speech in social media data using hierarchical classification (HC). The goal of this research is to inspire change. Since many platforms are free speech platforms, and we want to respect that right, maybe a suggestion can pop up when hate speech is detected, to encourage reappraisal and reconsideration. Many methods for detecting hate speech in English already exist, but since hate speech is not bound to any language, we are exploring detecting hate speech in Portuguese to help tackle this global issue.

HC has been used in genetics for genes classification, and later on expanded to text classification. Literature shows that HC is promising to the linguistics field, with preliminary studies concluding that hierarchical algorithms perform better than flat classification for sentiment analysis, emotion recognition, and hate speech. There is a lot of this type of detection and labeling in English, but it is important for other languages as well, and different languages have different slang, manners of speech, and subtleties which need to be taken into consideration. In this project, we are going to identify hate speech using Brazilian Portuguese Twitter data, implementing the algorithm using Python.

2 Previous Work

1. Li (2015): Hierarchical classification is used to compute sentiment analysis using multi-tier filtering schemas and various classification methods. Maximum entropy and Naive Bayes classifiers gave the highest accuracy and Winnow performed the worst. Additionally, a basic positive negative binary classifier outperformed classifiers with more options.

In this research, they discovered that words connected to a specific topic do not have any relationship with a specific sentiment. Additionally, removing words with strong sentiment increases accuracy since the model can now detect more subtle sentiment and will not be subject to over-fitting. This paper compares various classification models in depth to detect the most accurate one, which is very helpful. One weakness is that removing words with strong sentiment may make it harder to detect overall sentiment. While this paper discusses sentiment analysis as a whole in English, our project specifically discusses negative sentiment in Portuguese.

2. Fortuna (2018): Machine Learning algorithms are a great technology to assess hate speech. Classification methods usually return the most accurate results, compared to clustering methods. However, it requires annotated data, which is resource intensive. Most of the available hate speech annotated data is available in English. In this research, we attempt to use Brazilian Portuguese. We are going to use the annotated dataset by Fortuna (2019), who based her master's degree in scraping and classifying 5,668 Tweets. The method used to classify the data was split into two:
   (a) a non-experts binary label classification of 'hate' vs 'no-hate'.
   (b) experts classification using a multi-label framework of 81 categories.

3. Prabowo et al (2019) found that hierarchical classification performed better than flat algorithms by 2.27%. Using word unigram for feature extraction and Support Vector Machine(SVM) for classification, the authors achieved a 68.43% accuracy on a Twitter dataset. Other classification methods studied were Random Forest Decision Tree (RFDT) and Naive Bayes (NB). In addition to word n-gram for feature extraction, they also looked at character n-gram. In terms of feature categories, the authors looked at 5 different levels of category granularity: 3, 5, 8, 9, 12, with the 9 label construct performing the best. Based on their learning, it is very important to construct the categories carefully. Too few categories approach the flat structure in performance and give little to no additional insight from the flat structure. Too many categories can lead to meaningless sub-groupings.

3 Method

The project methodology is structured into 2 parts: data and baseline. In data, we ran some basic exploratory data analysis with the preprocessed data. For the baseline, we pre-processed the data, then trained and tested models for both flat and hierarchical classification (HC) and evaluated the results.

Hierarchical Classification (HC) aims to organize large amounts of information into taxonomies/classes. HC is a classification problem in which the classification algorithm is defined based on a predetermined class taxonomy. Taxonomies are semantically called IS-A ($\prec$) hierarchy, represented using a tree or DAG (Direct Acyclic Graph):

- The only one greatest element "R" is the root of the tree.
- $\forall$ c_i, c_j $\in$ C, if $c_i \prec c_j$, then $c_j \prec c_i$.
- $\forall$ c_i $\in$ C, $c_i \prec c_i$.
- $\forall$ $c_i, c_j, c_k \in C$, $c_i \prec c_j$ and $c_j \prec c_k$ imply $c_i \prec c_k$

This project will be the first HC analysis applied to hate speech in Brazilian Portuguese. We hope to build on the learnings of previous work to improve performance and insights vs. prior work. The work is intended to answer some questions, such as:

- Is it possible to identify hate speech nuances based on an annotated dataset in Portuguese?
- Is it possible to fine-grain hate speech classification based on a multi-label annotated dataset in Portuguese?

Our hypothesis is that HC performs better than flat classification in Portuguese.

4 Data

The dataset "A Hierarchically Labeled Portuguese Hate Speech Dataset" (2019) is composed of two data sets collected from Twitter in Brazilian Portuguese. The first data set is composed of 5,667 Tweets with binary classification, hate speech or non-hate speech. The data was annotated by naive readers. This data will not be used in this analysis.

The second data set is composed of 5,668 Tweets, annotated by experts based on 81 hate speech categories, in addition to a binary variable classified into hate speech and non-hate speech. The taxonomy class labels are organized into a directed acyclic graph (DAG), multi-layer framework (MLP) of 81 labels, and 4 levels of depth. A leaf node can be connected to more than one parent node. The first depth of the data is the binary variable (hate speech or non-hate speech). The second level is composed of 11 categories: sexism, body, origin, homophobia, racism, ideology, religion, health, aging, migrants, and other-lifestyle. The full structure can be visualized here.

4.1 Dataset Statistics

There are 47,712 words in the corpus from a vocabulary size of 11,103. The content of the Tweets is characteristic of the internet, composed of humor, sarcasm, and irony. Sexism is the most popular category, followed by Women, and Homophobia. Among the top 10 words, "mulher" (woman), "ir" (go), "dia" (day), and "querer" (want) are the most frequent ones. There are 4,440 non hate speech cases, and 1,228 hate speech cases.

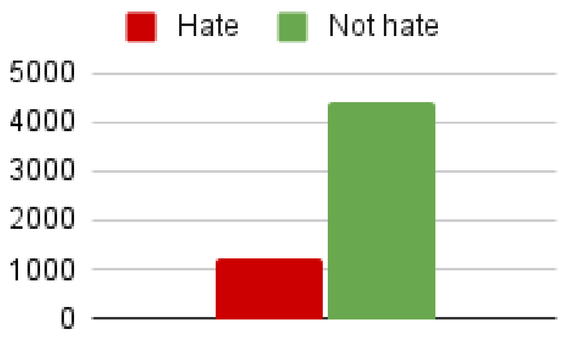


Figure 1: Approximately 20% of the data set is hate speech

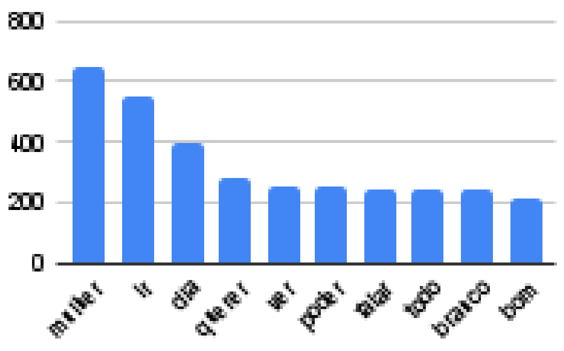


Figure 2: Top 10 words

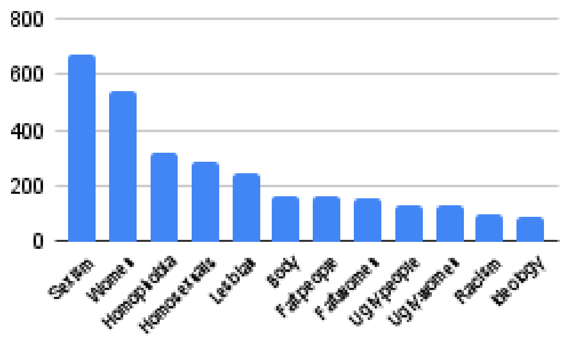


Figure 3: Top 12 categories make up approximately 80% of hate speech

4.2 Preprocessing Steps

We first removed mentions, and links. Then we used a lemmatizer specific to Portuguese which also tokenizes the data in the process. Next, we removed punctuation, normalized to lowercase, and removed all stopwords, including a list of stopwords we compiled manually. Flat classification did not require any additional processing. HC FastText expects the label to be in a specific format when read by the library, so we added the labels as a string with two dashes: `label__<X> __label__<Y> ... <Text>`.

Then, TF-IDF was applied to vectorize the text. TF-IDF means term frequency-inverse document frequency. It is a vectorizer technique that aims to quantify the importance of a given word relative to other words in the document and in the corpus. It assigns a weight to each word based on its frequency. This metric shows how relevant a word is in a document (if TF-IDF is higher, the word is more important). Instead of using raw frequency, TF-iDF scales down the impacts and weights of a token that occur more often in the document. Empirically such tokens are less informative compared to features occurring in a small fraction of the corpus. It should also be noted that it takes into account the number of documents in which the words can be found. It can be calculated by multiplying the term frequency and the inverse document frequency, as shown in the following formula, $w_{x,y} = tf_{x,y} * log\frac{N}{df_x}$, where $tf_{x,y}$ is the frequency of the term x within the document y, df_x is the number of documents containing the term x; N is the total number of documents. However, in the Scikit-Learn library, the metric is computed slightly different, i.e., the Euclidean norm is applied to the vectors, $IDF(t) = log\frac{1+n}{1+df(t)} + 1$

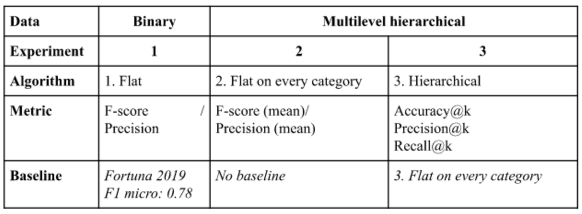
5 Baseline

The analysis is divided into three experiments, as shown in Figure 4. Experiment 1 uses a binary variable of hate speech/ non hate speech categories. Experiments 2 and 3 use the multilevel hierarchical variables: flat algorithms are applied to Experiment 2, and a hierarchical algorithm is applied to Experiment 3.

In experiment 1, CV f-score results are compared against the Fortuna 2018 results (0.78). In experiment 2, there is no baseline. Despite that, its results will be used as a baseline for experiment 3.

The flat algorithms are applied to experiment 1, and 2 following a pipeline of four traditional machine learning models: ComplementNB, Random Forest Decision Tree (RFDT), Logistic Regression, and xgBoost. Data was stratified into 10 folds, and we used the mean of the results of those 10 folds as our final results. xgBoost returned the highest accuracy (0.87) which is higher than the author's results. The model was built using sklearn[2].

Experiment 3 uses a hierarchical classification, taking into consideration metrics such as Hierarchical precision (hP), and Hierarchical recall (hR), which are based on ranking measures including Precision@k, and Recall@k, in which k represent the number of top predictions the model is evaluated on.



Data	**Binary**	**Multilevel hierarchical**	
Experiment	**1**	**2**	**3**
Algorithm	1. Flat	2. Flat on every category	3. Hierarchical
Metric	F-score / Precision	F-score (mean)/ Precision (mean)	Accuracy@k Precision@k Recall@k
Baseline	*Fortuna 2019 F1 micro: 0.78*	*No baseline*	*3. Flat on every category*

Figure 4: Top 12 categories make up approximately 80% of hate speech

6 Metrics

The cases in which the algorithms correctly predicted the outcomes considering the dataset used in this research. The TP are the news cases that were rightly classified as hate speech. The TN are the news cases that were rightly classified as NOT hate speech - they were not hate speech, and the algorithm predicted it to not be hate speech.

The cases in which the algorithms mistakenly predicted the outcomes. The FP are the news cases that were wrongly classified into hate speech, despite not being hate speech. The FN are the news cases that were wrongly classified as not hate, despite being hate speech.

Precision and recall are some of the metrics that can be used to evaluate the model's results. Precision helps to understand the relevant. Precision is defined as: Precision = TP/(TP + FP). In this case, precision measures the proportion of number of correctly hate speech, divided by the total number of news classified as hate speech, despite being correct or wrong. Recall is defined as: Recall = TP/ (TP + FN) In this case, recall measures the proportion of the number of correctly classified as hate speech, divided by the total number of (real) hate speech cases. Another metric that is also important to mention is F1 score, which combines both precision and recall. F1 score is calculated based on the harmonic mean of the metrics precision and recall. There are two other metrics based on F1 scores. The first one is the micro F1, and the second one is the macro F1. F1 Micro considers the average of F1 in each class, weighting the proportion of documents in each class. F1 Macro is the simple mean of the F1 scores in each class.

7 Algorithms

7.1 Flat Algorithms

Complement Naive Bayes (CNB) is a variant of the Naive Bayes classifier that reduces bias towards the majority class by weighting the features differently depending on the class they belong to, with a higher weight given to features that are more indicative of the minority class. This is specially useful for unbalanced datasets, like the one analyzed in this research. Logistic regression is a binary classification algorithm that calculates its probability using a sigmoid function to map results using the following equation: p = 1 / (1 + exp(-z)). Random Forest Decision Tree (RFDT) is a type of ensemble learning method for classification and regression. It creates a set of decision trees from a randomly selected subset of the training set, combining the predictions made by each tree to come up with the final predictions. XGBoost (eXtreme Gradient Boosting) is an open source library that is also based on ensemble decision tree methods. Its goal is to optimize the loss function through using a gradient descent algorithm.

[2]Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

7.2 Hierarchical algorithms

The HC model was built based on XFT models (Extreme Text Classification models), a multi-label classification method suitable for a large number of labels, similar to Word2Vec in regards to training, the main difference is that FastText trains a character n grams instead of words. Extreme FastText is an open-source library for text classification and representation learning. It is built on top of the popular FastText library and provides several additional features such as pre-trained models, vector representations of words, sentence embeddings, and more. XFT allows users to quickly and easily build high-performance models for text classification tasks. It is designed to be fast, efficient, and easy to use.

ExtremeText performs two operations: preprocesses by vectorizing the data, then inputs it into a single hidden layer based on a Probabilistic Labels Tree (PLT) loss with top-down hierarchical clustering (k-means) for tree building. TF-IDF was extracted as features; stochastic gradient descent as an optimizer; learning rate of 0.05, L2 regularization of 0.003, 100 dimensions, and 300 epochs.

8 Experiments

8.1 Experiment 1: Flat algorithm based on the binary variable of hate speech/ non hate speech

The first experiment is based on the binary variable of hate speech/ non hate speech. In this case, the hierarchical classification is not considered. The intention is to replicate the experiment done by Fortuna 2018. The highest score provided by the author is a CV f-score of 0.78.

8.2 Experiment 2: Flat algorithm applied to each to category

In the second experiment, the same set of algorithms of experiment 1 (ComplementNB, Logistic Regression, RFDT, and XGBoost) was applied. The difference is the data used. In this case, these algorithms were performed in each variable of the hierarchical dataset. The goal here is to use the results as a comparative to experiment 3.

8.3 Experiment 3: Multilevel hierarchical algorithm

XFT models (Extreme Text Classification models) are also evaluated with metrics such as precision, recall, and F1 score. K@1 is a metric that measures the top-1 accuracy of a model, which is the percentage of times the model correctly predicts the most likely class for a given input. K@2 is similar to k@1, but involves evaluating the model's top-2 predictions rather than just its top prediction, and so on, up to k@n. These metrics can be useful for understanding the model's performance at different levels of granularity, and for making tradeoffs between precision and recall depending on the specific task and the desired performance.

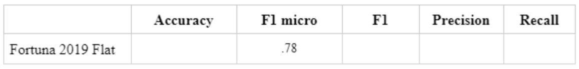
9 Results/Evaluation

9.1 Experiment 1 Results

The overall accuracy of the RFDT classifier was 0.877, followed by XGboost (0.865), Logistic Regression (0.860), and ComplementNB (0.833). The RFDT classifier also had the greatest F1 micro score (0.877), which measures the harmony between accuracy and recall. RFDT classifier had the greatest F1 score, which is a weighted average of precision and recall, at 0.654, followed by XGboost at 0.590, logistic regression at 0.570, and ComplementNB at 0.598.

RFDT classifier had the highest precision (measured as the percentage of genuine positive results out of all positive outcomes) at 0.838, followed by logistic regression (at 0.853), XGboost (at 0.856), and ComplementNB (at 0.627). Recall was greatest for XGboost at 0.450, followed by logistic regression at 0.429, ComplementNB at 0.573, and RFDT classifier at 0.537.

Recall represents the percentage of true positive outcomes out of all real positive results. Overall, it appears that the XGboost classifier had the highest recall while the RFDT classifier had the highest accuracy and F1 scores. It is vital to keep in mind that the results may be affected by the particular dataset and classification job, thus it is crucial to take these factors into account while selecting a method.

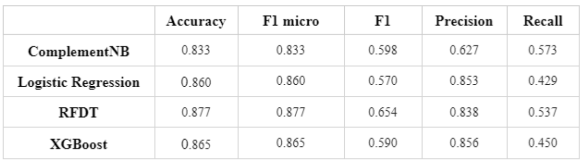


	Accuracy	F1 micro	F1	Precision	Recall
Fortuna 2019 Flat		.78			

Figure 5: Fortuna 2019 Results/Baseline. Only F1 micro score was reported

9.2 Experiment 2 Results

The results in this section are based on the average of accuracy, F1, F1 micro, precision, and re-



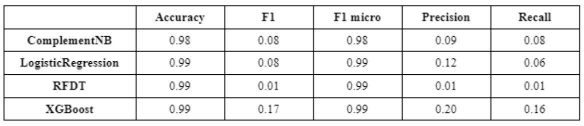
	Accuracy	F1 micro	F1	Precision	Recall
ComplementNB	0.833	0.833	0.598	0.627	0.573
Logistic Regression	0.860	0.860	0.570	0.853	0.429
RFDT	0.877	0.877	0.654	0.838	0.537
XGBoost	0.865	0.865	0.590	0.856	0.450

Figure 6: Flat algorithm based on binary variable of hate/non hate. F1 micro scores are reported for purpose of comparison with Fortuna result. F1 scores will be used to compare binary results with hierarchical results

call across all categories after the set of algorithms ComplementNB, Logistic Regression, RFDT, and XGBoost) were applied in each one of them.

The models appear to be doing well in terms of accuracy in the findings you supplied, with values ranging from 0.98 to 0.99. Their performance, however, varies more in terms of F1, accuracy, and recall. The LogisticRegression and XGBoost models have somewhat higher F1, accuracy, and recall values than the ComplementNB model, which has quite low values. The RFDT model performs poorly on these parameters, as seen by its F1, accuracy, and recall values of 0.

When assessing a model's performance, it's crucial to take the accuracy vs. recall trade-off into account. Conversely, a model with more precision may also have worse recall. The best model for a given application will rely on how important accuracy and recall are in relation to one another.



	Accuracy	F1	F1 micro	Precision	Recall
ComplementNB	0.98	0.08	0.98	0.09	0.08
LogisticRegression	0.99	0.08	0.99	0.12	0.06
RFDT	0.99	0.01	0.99	0.01	0.01
XGBoost	0.99	0.17	0.99	0.20	0.16

Figure 7: Average metrics based on all categories

9.3 Experiment 3 Results

Experiment 3 uses a PLT implemented by XFT models (Extreme Text Classification models) to perform the analysis. A few scenarios were considered in this case: preprocessed data, raw data/ non preprocessed, categories with more than 5 cases, and categories with more than 100 cases.

These results show the performance of a model on a classification task at different values of "k". The metric "A@k" indicates the accuracy at k, meaning the proportion of predictions that are correct, "P@k" (precision at k) is the proportion of predicted labels, "R@k" (recall at k) is the proportion of actual labels that are correctly predicted, and "f1@k" is the harmonic mean of precision and recall at k.

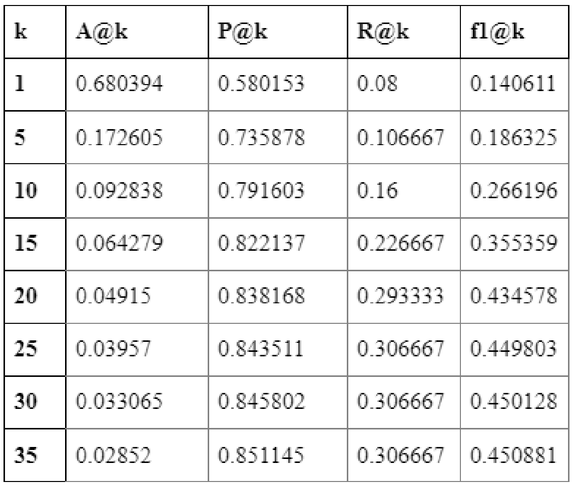
9.3.1 Preprocessing the categories

In this first experiment using HC XFT models, it will use preprocessed data as an input. The preprocessing follows the same process described in the section data of this research.

The model shows that A@k values dramatically decrease when k => 5, from A@1 = 0.68 to A@2 = 0.17, until 0.029 in k = 35. This indicates that the model is struggling to correctly classify examples in the dataset in extreme categories. Now, let's consider A@k in conjunction with other evaluation metrics, such as P@k and R@k, to get a complete picture of the model's performance.

R@k ranges from 0.08 to 0.307, which means that the classifier is performing poorly, as the values of recall are generally low across different values of "k". On the other hand, it retrieves relatively high P@k values (ranging from 0.58 to 0.85), especially when considering more labels (k $\geq$ 5). This could mean that the model is confident in its predictions, even when labels are not correctly classified. Finally, F1@k also indicates poor results, with indices ranging from 0.14 to 0.45.

Overall, this first experiment using hierarchical classification with a XFT model resulted in poor performance. It can be explained by a few reasons, such as optimizing the feature engineering process. In this case, the second experiment will be based on raw data to test if results could be improved.



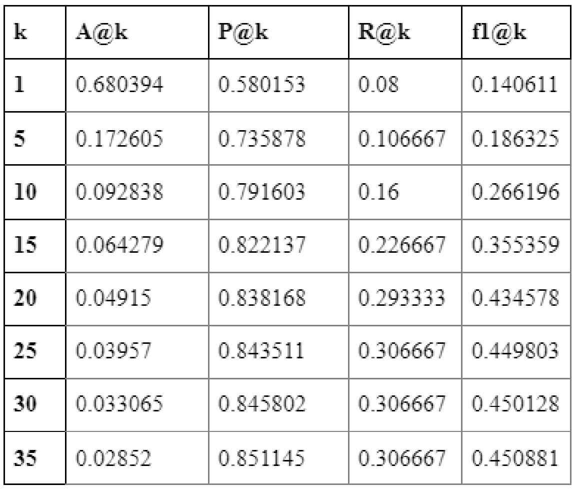
k	A@k	P@k	R@k	f1@k
1	0.680394	0.580153	0.08	0.140611
5	0.172605	0.735878	0.106667	0.186325
10	0.092838	0.791603	0.16	0.266196
15	0.064279	0.822137	0.226667	0.355359
20	0.04915	0.838168	0.293333	0.434578
25	0.03957	0.843511	0.306667	0.449803
30	0.033065	0.845802	0.306667	0.450128
35	0.02852	0.851145	0.306667	0.450881

Figure 8: All data preprocessed

9.3.2 Raw text/ not preprocessed

In the second experiment using HC XFT models using raw data, A@k results are slightly better than the preprocessing data, ranging from 0.71 to 0.03, and P@k, that goes from 0.6 to 0.91. However, the classifier has higher values of R@k (ranging from 0.04 to 0.33) when trained on raw data compared to when it is trained on preprocessed data. One possible reason to explain it is that the pre-processing can remove some important context-based tokens, which jeopardize the classifier's ability to perform on positive results.

In the second experiment using HC XFT models using raw data, A@k results are slightly better than the preprocessing data, ranging from 0.71 to 0.03, and P@k, that goes from 0.6 to 0.91. However, the classifier has higher values of R@k (ranging from 0.04 to 0.33) when trained on raw data compared to when it is trained on preprocessed data. One possible reason to explain it is that the pre-processing can remove some important context-based tokens, which jeopardize the classifier's ability to perform on positive results.

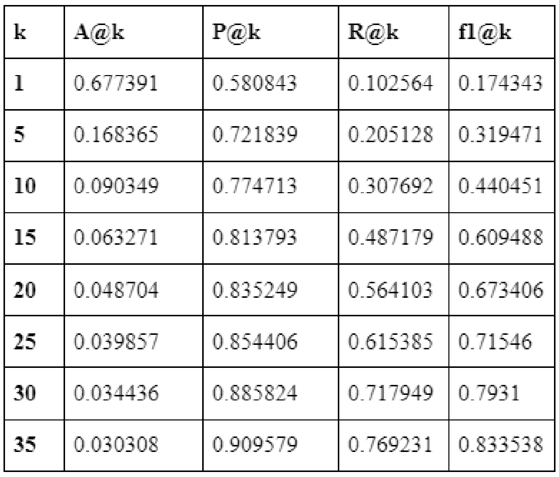


k	A@k	P@k	R@k	f1@k
1	0.680394	0.580153	0.08	0.140611
5	0.172605	0.735878	0.106667	0.186325
10	0.092838	0.791603	0.16	0.266196
15	0.064279	0.822137	0.226667	0.355359
20	0.04915	0.838168	0.293333	0.434578
25	0.03957	0.843511	0.306667	0.449803
30	0.033065	0.845802	0.306667	0.450128
35	0.02852	0.851145	0.306667	0.450881

Figure 9: Raw text/data not preprocessed

9.3.3 Excluding categories with less than 5 instances

In the third experiment using HC XFT, categories with less than 5 cases were excluded from the dataset with preprocessed data. A@k presented the worst results compared to the two previous experiments, ranging from 0.68 to 0.03. P@k was slightly better compared to the whole dataset experiment (3.1). However, poor results taking into account raw data (3.2). On the other hand, R@k was improved, ranging from 0.10 to 0.767, resulting in the best recall so far.



k	A@k	P@k	R@k	f1@k
1	0.677391	0.580843	0.102564	0.174343
5	0.168365	0.721839	0.205128	0.319471
10	0.090349	0.774713	0.307692	0.440451
15	0.063271	0.813793	0.487179	0.609488
20	0.048704	0.835249	0.564103	0.673406
25	0.039857	0.854406	0.615385	0.71546
30	0.034436	0.885824	0.717949	0.7931
35	0.030308	0.909579	0.769231	0.833538

Figure 10: Excluded categories with less than 5 instances

9.3.4 Excluding categories with less than 100 instances

In the fourth experiment using HC XFT, categories with less than 100 cases were excluded from the dataset with preprocessed data. These are the best results compared to the previous ones. Here, the classifier is performing well in terms of precision and recall, as the values of P@k and R@k are relatively high across different values of k, higher than 0.7 in both cases. The f1 scores are also high, suggesting a good balance between precision and recall. However, accuracy is still low when $k \geq 5$, which means it's not resulting in correct predictions.

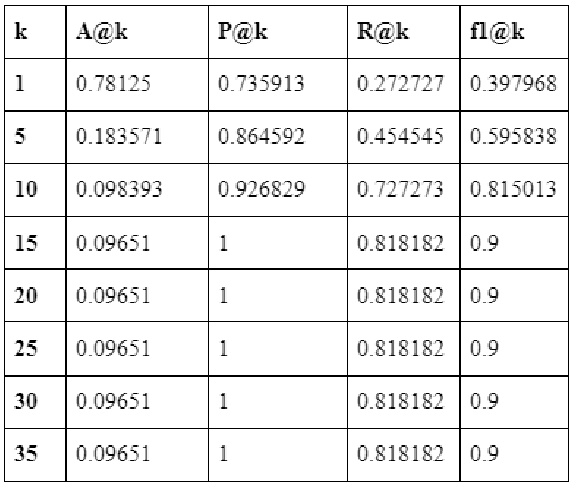
Overall, the trade-off of excluding so many categories improved precision, recall, and F1, but it had little effect on accuracy.

9.3.5 Using second level categories

In the fifth and last experiment using HC XFT, only categories that are in the second level of the DAG were considered: Aging, Body, Health, Homophobia, Ideology, Migrants, Origin, Other, Lifestyle, Racism, Religion, and Sexism. R@k and f1@k returned the best results so far, and P@k is slightly similar to experiment 3.4. However, A@k is still low for $k \geq 5$.

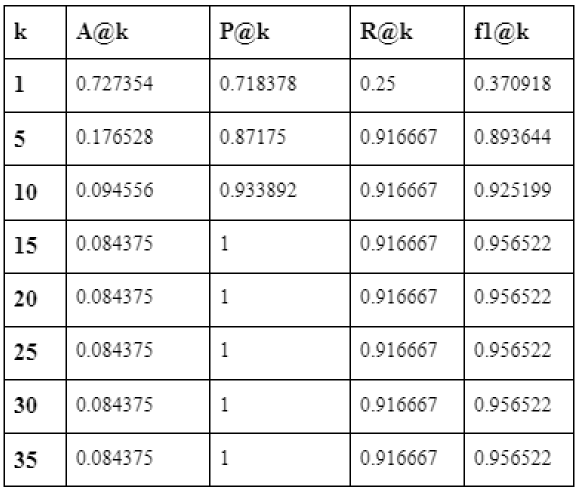
10 Discussion

This research was successful in replicating the experiment of Fortuna 2019 using traditional machine



k	A@k	P@k	R@k	f1@k
1	0.78125	0.735913	0.272727	0.397968
5	0.183571	0.864592	0.454545	0.595838
10	0.098393	0.926829	0.727273	0.815013
15	0.09651	1	0.818182	0.9
20	0.09651	1	0.818182	0.9
25	0.09651	1	0.818182	0.9
30	0.09651	1	0.818182	0.9
35	0.09651	1	0.818182	0.9

Figure 11: Excluded categories with less than 100 instances



k	A@k	P@k	R@k	f1@k
1	0.727354	0.718378	0.25	0.370918
5	0.176528	0.87175	0.916667	0.893644
10	0.094556	0.933892	0.916667	0.925199
15	0.084375	1	0.916667	0.956522
20	0.084375	1	0.916667	0.956522
25	0.084375	1	0.916667	0.956522
30	0.084375	1	0.916667	0.956522
35	0.084375	1	0.916667	0.956522

Figure 12: Second level categories

learning methods applied to the binary variable of hate speech/ no hate speech categories. It also outperformed the author's results in terms of F1 micro, reaching 0.877 using RFDT, compared to 0.78 mentioned by Fortuna 2019.

In general, the second experiment of applying a traditional algorithm to each category of the hierarchical dataset resulted in higher F1 micro compared to a single algorithm applied to the binary variable (> 0.98). Despite that, precision and recall were less than 0.20, reaching only 0.1 in the RFDT model.

In terms of hierarchical classification (HC XFT model), the results were dubious, pointing to an overall poor accuracy performance of the XTF model. In terms of model accuracy, traditional machine learning methods reported better results compared to HC XFT. Specifically, XGBoost returned the best accuracy, precision, recall, and F1 score combined.

Despite that, the HC XFT model generally outperforms traditional machine learning models in terms of precision and recall. However, the HC XFT model only showed significant improvements in terms of precision and recall when excluding categories with less than a certain number of cases. However, it goes against the purpose of the extreme hierarchical classification, that is considering a large number of categories, even when they are scarce in the data.

11 Conclusion

In conclusion, the HC XFT model is not an effective method in terms of quality predictions considering this specific dataset. The reason can be explained by the lower number of training set cases, which could result in poor predictions.

12 Future work

Some next steps might include using more data to train and test the model, especially more data labeled under the less common categories and subcategories. We also would be interested in experimenting more with the preprocessing to see which steps improve or worsen our results since it seems our results are better when we do not preprocess the data. We could also try running the models with different parameters to see how that impacts our results. One idea would be to recreate the extreme text training model in our code to debug through each step of the process and help us better understand our results.

13 References

Ricardo Cerri, & Rodrigo C. Barros, & André de Carvalho (2013, March 22). Hierarchical multi-label classification using local neural networks. Journal of Computer and System Sciences. Retrieved December 21, 2022, from https://www.sciencedirect.com/science/article/pii/S0022000013000718.

Silla, C. N., & Freitas, A. A. (2010, April 7). A survey of hierarchical classification across different application domains - data mining and knowledge discovery. SpringerLink. Retrieved December 21, 2022, from https://link.springer.com/article/10.1007/s10618-010-0175-9.

Miranda, F. M., Köhnecke, N., & Renard, B. Y. (2022, December 5). HiClass: A python library for local hierarchical classification compatible with scikit-learn. arXiv.org. Retrieved December 21, 2022, from https://arxiv.org/abs/2112.06560/ .

Fortuna, P., Silva, J. R. da, Soler-Company, J., Wanner, L., & Nunes, S. (n.d.). A hierarchically-labeled Portuguese hate speech dataset. ACL Anthology. Retrieved December 21, 2022, from https://aclanthology.org/W19-3510/ .

F. A. Prabowo, M. O. Ibrohim and I. Budi, "Hierarchical Multi-label Classification to Identify Hate Speech and Abusive Language on Indonesian Twitter," 2019 6th International Conference on Information Technology, Computer and Electrical Engineering (ICITACEE), 2019, pp. 1-5, doi: 10.1109/ICITACEE.2019.8904425.

J. Li, S. Fong, Y. Zhuang and R. Khoury, "Hierarchical Classification in Text Mining for Sentiment Analysis," 2014 International Conference on Soft Computing and Machine Intelligence, 2014, pp. 46-51, doi: 10.1109/ISCMI.2014.37.

Ghazi, D., Inkpen, D., & Szpakowicz, S. (n.d.). Hierarchical versus flat classification of emotions in text. ACL Anthology. Retrieved December 21, 2022, from https://aclanthology.org/W10-0217/ .