Multi-Lingual Contextual Hate Speech Detection Using Transformer-Based Ensembles

Maria Luisa Ripoll[1], Fadi Hassan[1], Joseph Attieh[1], Guillem Collell[1] and Abdessalam Bouchekif[1]

[1]*Huawei Technologies Oy., Finland*

Abstract
We present three different transformer-based approaches to contextual hate speech detection, submitted respectively to the three tasks of the HASOC 2022 competition [1]. To deal with the scarce dataset we use an ensemble of cross-validation trained models (CV ensemble) that makes use of all available data while still having validation sets to apply early stopping. We try different methods such as ensemble of ensembles, cross validation trained ensembles and embedding isotropy optimization. Furthermore, we compare different base models for the ensembles as well as different solution proposals to the tasks. Our team, *hate-busters*, ranked 3rd in task 1, 5th in task 2, 3rd in task 3A, 1st in task 3B and 4th in task 3C.

Keywords
Ensemble, Multi-Lingual NLP, Hate Speech, HASOC 2022

1. Introduction

The automatic detection of hate speech in internet forums and platforms has the capacity to significantly improve the online safety of millions of users. Yet, abusive text detection is a challenging task.

Firstly, language is dynamic, especially conversational language in social media, where words are often generated or given new meanings. The constantly changing vocabulary requires an offensive language detection system to have an architecture capable of dealing with unknown words.

Secondly, the rise of social media in recent decades has also led to an increase in global inter-connectivity and a rich exchange of cultures and languages. It is not uncommon to find "*macaronic*" texts, namely texts written in hybrid mixtures of more than one language. While some natural language processing systems are already able to recognise and classify comments in multi-lingual settings, this code-mixed language in single text samples is an added difficulty for current algorithms. Furthermore, dialects and what are known as low-resource languages, often lack the large amounts of annotated data required to train deep neural networks, which leads to unbalanced levels of moderation and protection against hate speech across languages.

Thirdly, language and semantics are both very complex and highly contextual. Sometimes, external knowledge is needed to understand the meaning of a sentence. In the case of a

Forum for Information Retrieval Evaluation, December 9-13, 2022, India
maria.luisa.ripoll@huawei.com (M. L. Ripoll); fadi.hassan@huawei.com (F. Hassan); joseph.attieh@huawei.com (J. Attieh); guillem.collell1@huawei.com (G. Collell); abdessalam.bouchekif@huawei.com (A. Bouchekif)

conversation, a machine should take into account the previous text inputs when classifying a sample since they could have an effect on the samples label. For instance, a text with only neutral content should be moderated if it is a reply agreeing with a hateful comment. Because of this, a good hate speech detection system should include contextual knowledge. Finally, an extra task for abusive text detection is identifying the target of the hate speech, which provides more fine-grained information of the text.

Facing all of these issues is important if we are to fully solve the problem of automatic hate speech detection and moderation. This is one of the main purposes behind the HASOC 2022 competition [2] assignments presented this year. The competition is motivated by the discussed challenges and asks participants to solve tasks using twitter data:

The first task is the (1) Identification of Conversational Hate-Speech in Code-Mixed Languages (**ICHCL**) [3]. This task tests the model's ability to identify hate speech in a multi-lingual setting with the languages being german and hinglish (macaronic text of hindi and english). For this task we test different ensemble architectures and compare results.

The second task is (2) multiclass ICHCL for hinglish text and it consists in identifying whether a sample is standalone hate, contextual hate or no hate [3]. Thus, the task adds on the challenges related to context. Our second system uses an ensemble of cross-validation trained models to solve the problem.

The final task is (3) Offensive Language Identification in Marathi and it addresses the problems for low-resource languages as well as the added complexity of identifying targeted hate speech [1, 4]. This task is divided into three sub-tasks. Each of which dedicated to different levels of multi-class classification of targeted hate speech. We present a single model capable of solving all three sub-tasks. By pre-processing the data and merging the multiple labels per sample into one, we convert the multi-class multi-label problem into a multi-class, single-label problem.

This work is structured into six sections. Section 1 is the introduction. Section 2 outlines the related work. Section 3 describes the data for every task and section 4 gives an overview of the processing steps and system architectures per task. The results are presented in section 5, followed by the conclusion in section 6 and the Appendix.

2. Related Work

The automatic detection of hate speech is a research area that has been growing over the last few years. There have been numerous surveys detailing the available data and methodologies used in hate speech detection [5, 6, 7, 8] and most of them conclude that the definition of hate speech itself is still unclear [5, 8]. This uncertainty leads to disparate annotations guidelines across datasets, languages and domains and makes the task of hate speech detection very complex.

Due to its complexity and relevance, there have been a series of data challenges posted online to encourage researchers to solve particular hate-speech detection tasks. Among these challenges were: Prolifing Hate Speech Spreaders on Twitter - PAN 2021 [9], SemEval-2019 Task 5: Detection of Hate Speech Against Immigrants and Women in Twitter (hateval) [10] or HASOC 2021: Hate Speech and Offensive Content Identification in English and Indo-Aryan Languages and Conversational Hate Speech [11]. This year we participate in HASOC 2022 [1] which builds upon the tasks presented in HASOC 2021. As detailed by Modha et al [12], the top submissions from last years challenges included various technologies such as graph convolutional networks [13], transformers [14] and ensembles of transformers [15]. HASOC 2021 concluded that the transformer architecture is state-of-the-art for automatic hate speech detection.

3. Data

The data used was provided by HASOC 2022 [2]. The text samples for the first and second tasks were organized in a 3 level tree structure with tweets, tweet comments and comment replies. Task 1 included 6284 samples in two languages: german and hinglish and had binary labels for hateful (HOF) and non-hateful text (NOT). Task 2 used the same hinglish text samples as task 1, excluding the german training data and included three sets of labels for none hateful data (NONE), standalone hate (SHOF) and contextual hate (CHOF), the latter being those comments that are only hateful due to context with its parent text.

The data for task 3 [16, 17] is no longer in a tree structure. All sample tweets and respective labels are independent from each other. There are a total of 3013 training samples, all of which are used in subtask A to identify offensive posts (HOF) from non-offensive (NOT) ones. Subtask B, however, only uses those 1068 samples that are classified as offensive in subtask A and determines whether they are targeted (TIN) or untargeted (UNT) insults. The 740 data samples that are classified as targeted insults in subtask B form the training data for subtask C, which consists in classifying the target of the insult as an individual (IND), a group (GRP) or other (OTH).

Table 1 presents a data summary of the languages, number of train samples, number of test samples and class labels per task and subtask.

Task	Language	Train Samples	Test Samples	Full Dataset	Classes
	German	307	81	388	
1	Hinglish	4901	995	5896	Binary (NOT/HOF)
	Both	5208	1076	6284	
2	Hinglish	4901	995	5896	Multi (NONE/SHOF/CHOF)
3A	Marathi	3013	510	3523	Binary (NOT/OFF)
3B	Marathi	3013 (1068)	510	1578	Binary (TIN/UNT)
3C	Marathi	3013 (740)	510	1250	Multi (IND/GRP/OTH)

Table 1
Data Summary of data language, number of samples and class labels per task. Train samples in parenthesis are the subset of samples which do not have NaN label values for the corresponding subtask.

4. Methodology

4.1. Task 1: ICHCL Binary Classification

For the first task, the team worked on 3 different solutions in parallel and then merged them into one ensemble model as can be seen in Figure 1.

The first solution used an XLM-Roberta huggingface model [18, 19] as a base and added an extra layer as a head. The extra layer was trained to jointly optimize the loss function on the classification task as well as the isotropy property of the last hidden layer's embeddings. In a non isotropic space, the information in the embeddings is not uniformly distributed amongst all directions in the space. This is not desirable as these representations vary the most in top directions, which limits the expressiveness of the space. To improve the isotropy property of the space, a penalty term is added to the loss which improves the embedding space, making the embeddings less similar and as shown in previous works [20], better prepared to learn the downstream task. The input data was pre-processed by concatenating the text with any possible parents (the tweet in the case of the comment and the tweet and comment in the case of the reply), then removing all emojis, redundant spaces, URLs, underscores and hashtags and finally tokenizing the sample using the models corresponding tokenizer. The data was divided into training and validation sets using an 75 - 25 split in order to be able to use early stopping. For the rest of this work we refer to this model as the *Isotropy Model*.

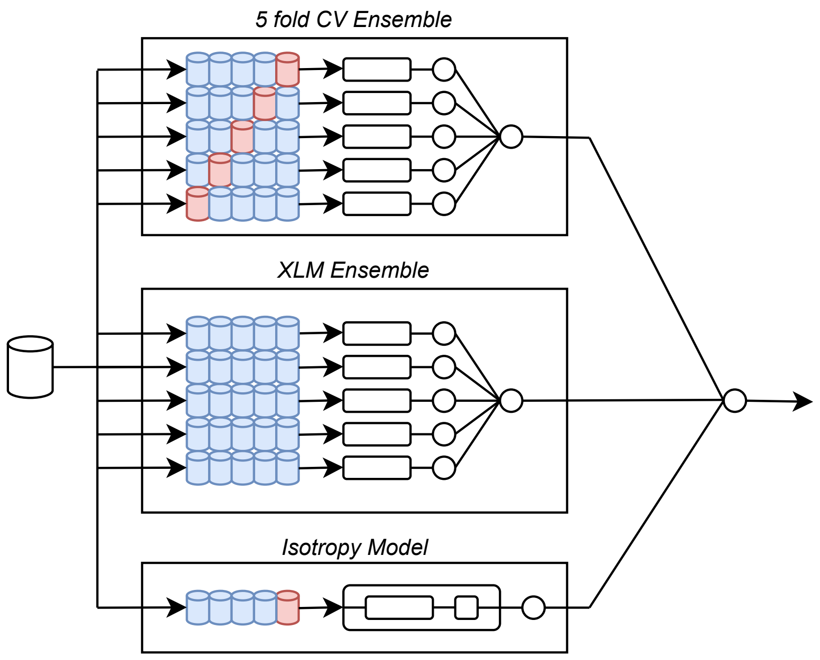

Figure 1: System Architecture for Task 1. In this figure, cylinders represent data. Smaller cylinders are color coded subsets with blue representing train data and red representing evaluation data. Rectangles correspond to models, squares to individual layers and circles to scores.

The second solution used the same XLM-Roberta huggingface model and tokenizer as the ones used in the isotropy model. In this case, the samples were also concatenated with all parent texts yet the data was not normalized - nothing was removed. Furthermore, since the amount of data for training such a large model was quite scarce, it was decided that this model would use the entirety of the available data for training and spare no samples for validation. The model was trained for 5 epochs. This model architecture was then run using 5 different seeds. The scores of the five trained models were averaged resulting in an ensemble, which we refer to as the *XLM Ensemble*.

The final solution used a different XLM-Roberta huggingface base model and tokenizer, namely Microsoft's InfoXLM [21]. We use InfoXLM since it is a framework that focuses on improving cross-lingual representations, which is relevant for task 1's multilingual classification assignment. Using the InfoXLM tokenizer, every text was aggregated with its parents using separator tokens. In this case scenario, the aim was to be able to apply early stopping on the models to avoid overfitting, while also using all the available data for training. To do this, we applied a form of *cross-validation training* where the dataset was divided into 5 folds. We trained 5 models using every fold once for validation and the other 4 for training. Then we averaged the scores of all 5 models together into an ensemble. This ensemble is hereafter referred to as a *K-fold CV (Cross Validation) Ensemble*.

The scores of all three solutions are averaged into one ensemble. Since two out of the three models are ensembles by themselves we call this last ensemble the *ensemble of ensembles*.

4.2. Task 2: ICHCL Multiclass Classification

The system architecture for the model submitted to task 2 is a 5-fold CV Ensemble as described in section 4.1 and uses the same data pre-processing steps. Diagram 4 in the appendix shows Task 2's model architecture.

4.3. Task 3: Offensive Language Identification in Marathi

The third task is divided into three subtasks as described in section 3. Our aim is to train a joint model capable of solving all three subtasks at once. To achieve this we can look at every subtask as a categorical label, and the labels within that subtask as classes. From this point of view task 3 is reduced to a multi-label, multi-class problem.

We can further simplify the task by merging all labels into a single categorical label, thus converting the multi-label, multi-class problem into a single-label, multi-class problem. Since only the HOF class is used in subtask B and only TIN is used in subtask C, the combinations of labels across subtasks is small and the new label will only have five possible classes: NOT, OFF-UNT, OFF-TIN-IND, OFF-TIN-GRP and OFF-TIN-OTH. Figure 2 shows the data distribution across classes and subtasks with the original labels. We can compare this distribution to that of Figure 3, which shows the data distribution after merging the labels into one.

Once the labels are merged, we tokenize the data and feed it into our CV Ensemble architecture. For task 3 we tested two different huggingface models: InfoXLM [21] and L3Cube [22] with the latter being a language-specific model trained on Hindi as well as the target language Marathi. We also compared results when using 5 and 10 fold cross validation training.

5. Results

The results for task 1 are shown in table 2. The 5 fold CV ensemble received the best scores among the different architectures. The two most notable differences in this system are on the

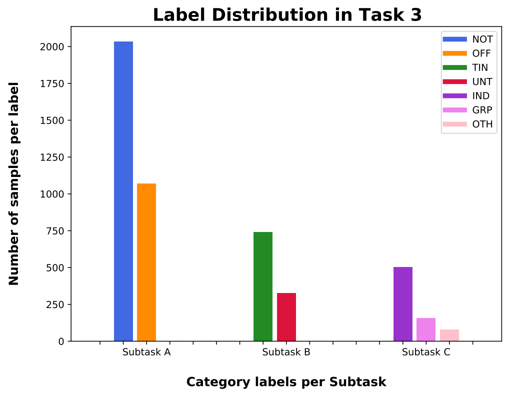

Figure 2: Label Distribution for Task 3

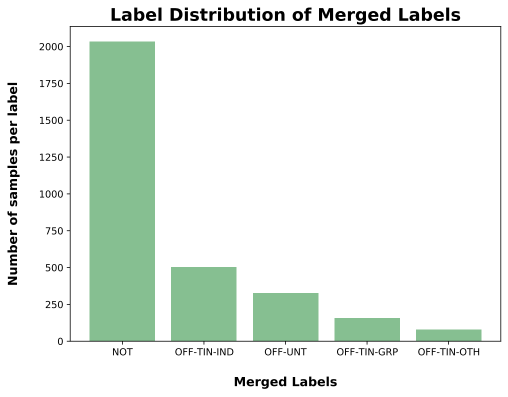

Figure 3: Label Distribution of Merged Labels

first hand the model used - InfoXLM and on the other hand, the cross validation training format, which allows the use of early stopping while still applying the entire dataset for training. Due to a lack of submission tries we were not able to test the performance for the cross validation training across different models, which would give us more insight into the architectures effects on the task. Also due to the lack of submission tries we did not test the isotropy model by itself.

The other systems tested in task 1 were also ensembles and achieved very similar scores to the 5 fold CV ensemble results. The final ensemble model (the *ensemble of ensembles*) included the 5 fold CV ensemble, the XLM ensemble and the isotropy model yet it did not achieve better results than the 5 fold CV ensemble by itself (+0.3% Macro F1), proving that this form of ensemble stacking is not effective.

Our team achieved 3rd position in this task with a F1 difference of 4.73% to the top submission and a difference of 0.11% to the second position.

Table 2
Test Results for Task 1 and 2. Run Names can be found in table 4 in the Appendix.

Task	Model	Macro F1	Macro Precision	Macro Recall
1	XLM ensemble	65.69%	65.74%	65.71%
	Ensemble of ensembles	65.79%	65.80%	65.79%
	5 fold CV ensemble	**66.10%**	**66.34%**	**66.19%**
2	Single XLM (InfoXLM)	43.87%	48.03%	43.87%
	5 fold CV ensemble	**46.51%**	**53.48%**	**47.31%**

Table 2 also shows the results for task 2. The overall results of the submissions presented by all teams are lower than those from task 1 due to the increased complexity of the task. Our team ranked 5th and achieved a score of 46.51%, which was 2.88% lower than the top score (49.39%). Within our submissions, we compare our 5 fold CV ensemble method with a single InfoXLM model and we observe an increase of almost 3% when using the cross validation ensemble.

For task 3 we compare between three different architectures tested on all three subtasks.

When comparing the performance of the 5 fold CV ensemble using InfoXLM as base against the L3cube we see a relatively significant increase in performance for the marathi trained L3Cube across all tasks. The increase is of +4.21%, +12.78% and +8.19% in F1 for tasks A, B, and C respectively.

The second comparison we can observe is the difference between the 5 fold CV ensemble and the 10 fold CV ensemble. Increasing the value of k was beneficial across all tasks, yielding improvements of +1.17%, +8.71% and +12.13% in F1 for tasks A, B and C. In this case, the difference in the magnitude of the improvements could also be due to the change in the number of samples available per task (see figure 3). Subtask A has almost 3 times as many samples as subtask B and almost 4 times as many as subtask C. Taking away a fifth of the data for validation vs a tenth makes a significant difference for such scarce data.

Another observation is the degrading performance as tasks become more fine-grained with the overall performance of submissions for task A being higher than those for task B, who are themselves higher than those for task C. Withal, the results from task 3 indicate clearly that both language-specific training and a higher value of k in the k-fold CV ensemble consistently improve model performance for said tasks.

For subtask A our team ranked 3rd scoring 1.77% F1 lower than the top submission and 0.21% F1 lower than the second submission. For subtask B, our team ranked 1st with an improvement of 0.58% F1 over the second position. And finally, for subtask C, our team ranked 4th. The variation of results for this task was larger than for the other tasks with the top result achieving a score 16.78% F1 higher than the second result. Our team scored 23.52% F1 lower than the top rank.

Table 3
Test Results for Task 3. The run names for every model can be found in table 5 in the Appendix.

Subtask	Model	Macro F1	Macro Precision	Macro Recall
A	5 fold CV ensemble - InfoXLM	90.39%	90.39%	90.40%
	5 fold CV ensemble - L3cube	94.51%	94.52%	94.50%
	10 fold CV ensemble - L3cube	**95.68%**	**95.92%**	**95.62%**
B	5 fold CV ensemble - InfoXLM	70.58%	71.11%	70.27%
	5 fold CV ensemble - L3cube	83.36%	83.10%	84.86%
	10 fold CV ensemble - L3cube	**92.07%**	**91.14%**	**93.42%**
C	5 fold CV ensemble - InfoXLM	52.23%	47.16%	59.90%
	5 fold CV ensemble - L3cube	60.42%	54.89%	67.77%
	10 fold CV ensemble - L3cube	**72.55%**	**79.99%**	**77.05%**

6. Conclusion

We have presented transformer-based ensemble solutions for each of the tasks in the HASOC 2022 competition. Our method of cross validation (CV) training has shown to be beneficial compared to our XLM Ensemble approach in this low resource setting for hate speech detection. The advantage of this CV method is that - by alternating the validation and training sets - it allows us to use the entire available data for training as well as to implement early stopping. From our experiments with InfoXLM and L3Cube in task 3 we can also conclude that using the evaluation language in the training set increases performance significantly, therefore using language-specific models instead of state of the art cross lingual and multilingual models is still in several cases a better option.

References

[1] T. Ranasinghe, K. North, D. Premasiri, M. Zampieri, Overview of the HASOC subtrack at FIRE 2022: Offensive Language Identification in Marathi, in: Working Notes of FIRE 2022 - Forum for Information Retrieval Evaluation, CEUR, 2022.

[2] S. Satapara, P. Majumder, T. Mandl, S. Modha, H. Madhu, T. Ranasinghe, M. Zampieri, K. North, D. Premasiri, Overview of the HASOC Subtrack at FIRE 2022: Hate Speech and Offensive Content Identification in English and Indo-Aryan Languages, in: FIRE 2022: Forum for Information Retrieval Evaluation, Virtual Event, 9th-13th December 2022, ACM, 2022.

[3] S. Modha, T. Mandl, P. Majumder, S. Satapara, T. Patel, H. Madhu, Overview of the HASOC Subtrack at FIRE 2022: Identification of Conversational Hate-Speech in Hindi-English Code-Mixed and German Language , in: Working Notes of FIRE 2022 - Forum for Information Retrieval Evaluation, CEUR, 2022.

[4] M. Zampieri, T. Ranasinghe, M. Chaudhari, S. Gaikwad, P. Krishna, M. Nene, S. Paygude, Predicting the type and target of offensive social media posts in marathi, Social Network Analysis and Mining 12 (2022) 77. URL: https://doi.org/10.1007/s13278-022-00906-8. doi:10.1007/s13278-022-00906-8.

[5] F. E. Ayo, O. Folorunso, F. T. Ibharalu, I. A. Osinuga, Machine learning techniques for hate speech classification of twitter data: State-of-the-art, future challenges and research directions, Computer Science Review 38 (2020) 100311.

[6] N. Chetty, S. Alathur, Hate speech review in the context of online social networks, Aggression and violent behavior 40 (2018) 108–118.

[7] M. A. Paz, J. Montero-Díaz, A. Moreno-Delgado, Hate speech: A systematized review, Sage Open 10 (2020) 2158244020973022.

[8] F. Alkomah, X. Ma, A literature review of textual hate speech detection methods and datasets, Information 13 (2022). URL: https://www.mdpi.com/2078-2489/13/6/273. doi:10.3390/info13060273.

[9] F. Rangel, G. L. De la Peña Sarracén, B. Chulvi, E. Fersini, P. Rosso, Profiling hate speech spreaders on twitter task at pan 2021., in: CLEF (Working Notes), 2021, pp. 1772–1789.

[10] V. Basile, C. Bosco, E. Fersini, D. Nozza, V. Patti, F. M. Rangel Pardo, P. Rosso, M. Sanguinetti, SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter, in: Proceedings of the 13th International Workshop on Semantic Evaluation, Association for Computational Linguistics, Minneapolis, Minnesota, USA, 2019, pp. 54–63. URL: https://aclanthology.org/S19-2007. doi:10.18653/v1/S19-2007.

[11] S. Modha, T. Mandl, G. K. Shahi, H. Madhu, S. Satapara, T. Ranasinghe, M. Zampieri, Overview of the hasoc subtrack at fire 2021: Hate speech and offensive content identi-

fication in english and indo-aryan languages and conversational hate speech, in: FIRE 2021: Forum for Information Retrieval Evaluation, Virtual Event, 13th-17th December 2021, ACM, 2021.

[12] S. Modha, T. Mandl, G. K. Shahi, H. Madhu, S. Satapara, T. Ranasinghe, M. Zampieri, Overview of the hasoc subtrack at fire 2021: Hate speech and offensive content identification in english and indo-aryan languages and conversational hate speech, in: Forum for Information Retrieval Evaluation, 2021, pp. 1–3.

[13] N. Bölücü, P. Canbay, Hate speech and offensive content identification with graph convolutional networks, in: Forum for Information Retrieval Evaluation (Working Notes)(FIRE), CEUR-WS. org, 2021.

[14] M. Nene, K. North, T. Ranasinghe, M. Zampieri, Transformer models for offensive language identification in marathi, in: Forum for Information Retrieval Evaluation (Working Notes)(FIRE), CEUR-WS. org, 2021.

[15] A. Glazkova, M. Kadantsev, M. Glazkov, Fine-tuning of pre-trained transformers for hate, offensive, and profane content detection in english and marathi, arXiv preprint arXiv:2110.12687 (2021).

[16] T. Mandl, S. Modha, G. K. Shahi, H. Madhu, S. Satapara, P. Majumder, J. Schäfer, T. Ranasinghe, M. Zampieri, D. Nandini, A. K. Jaiswal, Overview of the HASOC subtrack at FIRE 2021: Hate Speech and Offensive Content Identification in English and Indo-Aryan Languages, in: Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation, CEUR, 2021. URL: http://ceur-ws.org/.

[17] S. Gaikwad, T. Ranasinghe, M. Zampieri, C. M. Homan, Cross-lingual offensive language identification for low resource languages: The case of marathi, in: Proceedings of RANLP, 2021.

[18] J. Plu, tf-xlm-roberta-large, 2020. URL: https://huggingface.co/jplu/tf-xlm-roberta-large.

[19] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Transformers: State-of-the-art natural language processing, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Online, 2020, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[20] D. Biś, M. Podkorytov, X. Liu, Too much in common: Shifting of embeddings in transformer language models and its implications, in: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Online, 2021, pp. 5117–5130. URL: https://aclanthology.org/2021.naacl-main.403. doi:10.18653/v1/2021.naacl-main.403.

[21] Z. Chi, L. Dong, F. Wei, N. Yang, S. Singhal, W. Wang, X. Song, X.-L. Mao, H. Huang, M. Zhou, InfoXLM: An information-theoretic framework for cross-lingual language model pre-training, in: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Online, 2021, pp. 3576–3588. URL: https://aclanthology.org/2021.naacl-main.280. doi:10.18653/v1/2021.naacl-main.280.

[22] R. Joshi, L3cube-hindbert and devbert: Pre-trained bert transformer models for devanagari

based hindi and marathi languages (2022). doi:10.13140/RG.2.2.14606.84809.

A. Appendix

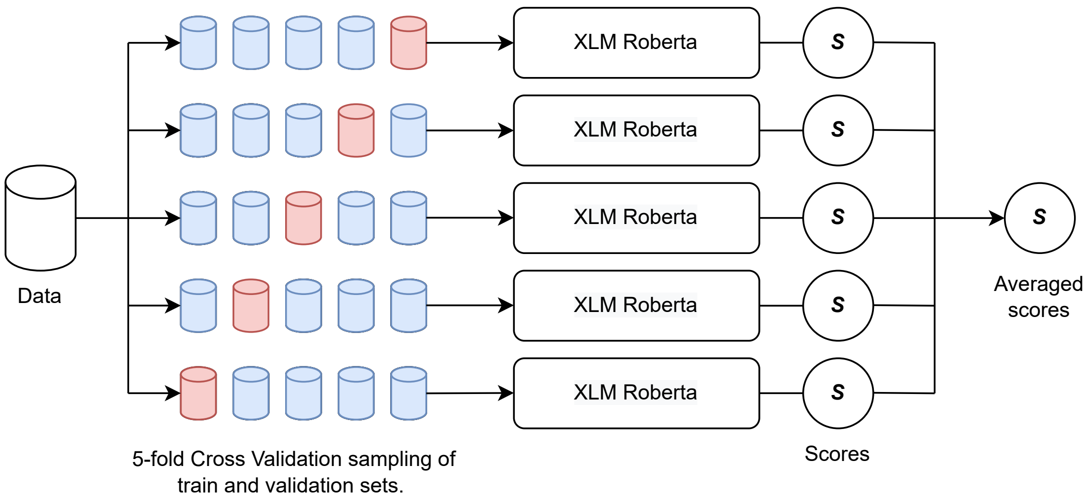
Data
XLM Roberta
S
5-fold Cross Validation sampling of train and validation sets.
Scores
Averaged scores

Figure 4: System Architecture of a 5 fold CV Ensemble. This was the main architecture submitted for tasks 2 as well as one of the models used in the ensemble solution for task 1. The final and best system for task 3 was a 10 fold version of this architecture.

Table 4
Run names for tasks 1 and 2

Task	Model	Run Name
1	XLM ensemble	traineval ensemb
	Ensemble of ensembles	ensemble_3models_task1.csv
	5 fold CV ensemble	infoxlm-5folds-fixed-joint_avg_task1
2	Single XLM (InfoXLM)	infoxlm_large-v2-single
	5 fold CV ensemble	infoxlm-5folds-joint_fixed_task2

Table 5
Run names for task 3

Subtask	Model	Run Name
A	5 fold CV ensemble - InfoXLM	infoxlm_large-5folds-joint
	5 fold CV ensemble - L3cube	task3-5fold-concat_a
	10 fold CV ensemble - L3cube	task3-10fold-concat_avg_a
B	5 fold CV ensemble - InfoXLM	infoxlm_large-5folds-concat_b
	5 fold CV ensemble - L3cube	task3-5fold-concat_avg_b
	10 fold CV ensemble - L3cube	task3-10fold-concat_avg_b
C	5 fold CV ensemble - InfoXLM	infoxlm_large-5folds-concat_c
	5 fold CV ensemble - L3cube	task3-5fold-concat_avg_c
	10 fold CV ensemble - L3cube	task3-10fold-concat_avg_c