Exploration of Multi-Corpus Learning for Hate Speech Classification in Low Resource Scenarios

Ashwin Geet d'Sa, Irina Illina, Dominique Fohr, Awais Akbar

Exploration of Multi-Corpus Learning for Hate Speech Classification in Low Resource Scenarios

Ashwin Geet D'Sa, Irina Illina, Dominique Fohr, and Awais Akbar

Université de Lorraine, CNRS, Inria, LORIA, F-54000, Nancy, France
{irina.illina@loria.fr, dominique.fohr@loria.fr}

Abstract. The dramatic increase in social media has given rise to the problem of online hate speech. Deep neural network-based classifiers have become the state-of-the-art for automatic hate speech classification. The performance of these classifiers depends on the amount of available labelled training data. However, most hate speech corpora have a small number of hate speech samples. In this article, we aim to jointly use multiple hate speech corpora to improve hate speech classification performance in low-resource scenarios. We harness different hate speech corpora in a multi-task learning setup by associating one task to one corpus. This multi-corpus learning scheme is expected to improve the generalization, the latent representations, and domain adaptation of the model. Our work evaluates multi-corpus learning for hate speech classification and domain adaptation. We show significant improvements in classification and domain adaptation in low-resource scenarios.

Keywords: hate speech detection · multi-task learning · low-resource text classification.

1 Introduction

An increase in online social media usage has led to a rise in hate speech. Hate speech is an anti-social behavior that targets a small part of the society, based on race, gender, etc. [9]. In many countries, hate speech is prohibited by the law and has to be filtered from social media platforms. However, manually analyzing the user contents is time-consuming and expensive. Natural language processing techniques can be used to automatically detect and filter hate speech content. Hence, there is an increased interest in automatic hate speech classification. Deep learning-based approaches have become the state-of-the-art for this task [3, 8, 17, 19, 14]. However, the performance of these classifiers depends on the amount of available labelled training data [2].

Typically, hate speech datasets are collected from sources such as Twitter [7, 13, 4], Wikipedia [26], etc. Characteristics of the dataset, such as the sampling strategy, the time frame [11] of the comments, and the definition of class labels [12], often bias the models trained on each dataset. Particularly, a model trained on one dataset can be inefficient on another dataset [25], resulting in the restricted generalizability of the model. Furthermore, these datasets have a small number of labelled samples. In order to bring diversity in the training data, and increase the number of samples to train the model, multiple hate speech corpora

can be harnessed to consider the corpus diversity and reduce the data sparsity issue. In this paper, we investigate a multi-task learning (MTL) approach, instead of a simple combination of different corpora.

MTL aims to jointly learn from multiple related tasks. MTL combines the domain-specific information and shares representations between related tasks, hence, can improve the generalization capabilities of the model on the target task [6]. MTL has applications in various domains, such as computer vision, bioinformatics, speech, natural language processing (NLP), etc. [27, 23].

MTL has been explored for hate speech classification. An MTL architecture having shared and private task-specific layers to capture shared and task-specific features, respectively, from different hate speech classification tasks is proposed in [15]. A joint model of emotion and abusive language detection, that allows one task to receive relevant information from the other tasks is introduced in [21]. They combine the features of single task-learning and MTL using an attention mechanism. Although these prior works have shown the effectiveness of MTL architectures, they haven't exploited the pre-trained models.

In this article, we design an MTL approach based on the work in [18], wherein the authors combined a range of NLP tasks using shared layers represented by the pre-trained BERT model and several groups of task-specific layers; each group corresponding to a single task. Compared to this work, we adapt the paradigm of multi-task learning to multi-corpus learning. In our approach, *a task* corresponds to *a corpus*. Compared to the works in [15, 21], we use the pre-trained Bidirectional Encoder Representations from Transformers (BERT [10]) model for our MTL to benefit from extensive knowledge learned by BERT pre-training. A Spanish BERT model in an MTL setup has showed improvements for hate speech classification tasks in [20]. However, they incorporate sentiment analysis and emotion analysis tasks in their MTL. Instead, we exploit the relatedness of hate speech classification tasks by using five well-known hate speech datasets extracted from Twitter and Wikipedia. Furthermore, these prior works do not study the performance of the MTL approach in low-resource scenarios. Thus, we explore a low-resource domain adaptation scenario in the framework of multi-corpus learning.

Our contributions are summarized as follows:

1. We adapt MTL approach to *multi-corpus learning* (MCL) for hate speech classification and validate it on widely used hate speech corpora.
2. We study the robustness of the proposed MCL in low-resource scenarios.
3. We study low-resource domain adaptation.

2 Proposed Methodology

In this section, we first describe MTL. This is followed by our approaches for MCL and domain adaptation in low-resource scenario.

2.1 Objective of Multi-Task Learning

Given T related tasks $\{t_i\}_{i=1}^{T}$, MTL aims to jointly learn these tasks to improve the model performance on each task t_i. Let us consider supervised learning task

t_i, with n_i samples $(x_1^i, x_2^i \dots x_{n_i}^i) \in X^i$ and labels $(y_1^i, y_2^i \dots y_{n_i}^i) \in Y^i$. For the task t_i, a MTL model learns the parameter set $\{\theta^s, \theta^i\}$ using a function f^i as follow:

$$f^i(X^i; \theta^s, \theta^i) : X^i \rightarrow Y^i \tag{1}$$

where θ^s are the model parameters shared between all the tasks in $\{t_i\}_{i=1}^T$, and θ^i represents the task-specific model parameters. The objective is to minimize the overall loss L:

$$L(\theta^s, \theta^1, \theta^2, \dots \theta^T) = \sum_{i=1}^{T} L^i(\theta^s, \theta^i) \tag{2}$$

where $L^i(\theta^s, \theta^i)$ is the loss for task t_i, and, in the supervised case, can be evaluated as follow:

$$L^i(\theta^s, \theta^i) = \frac{1}{n_i} \sum_{j=1}^{n_i} \mathcal{L}(f^i(x_j^i; \theta^s, \theta^i), y_j^i) \tag{3}$$

Where, $\mathcal{L}$ is a loss function measuring how well the function f^i fits the training data (X^i, Y^i). The objective of MTL is to reduce the overall loss L, by optimizing the task-specific parameters $\{\theta^i\}_{i=1}^{Ts}$, and the parameters shared across all the tasks θ^s. In a single task learning approach, $T = 1$ and the dataset of task t_1 is processed by a model with parameters θ^1.

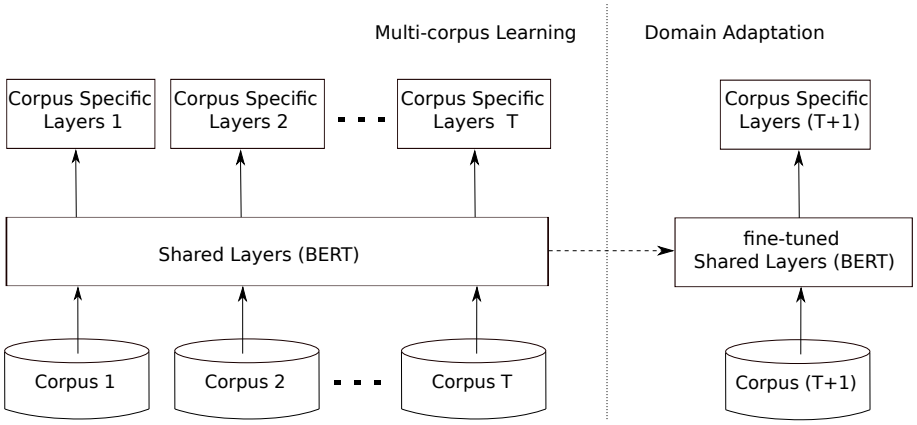


Fig. 1: Architecture of the multi-corpus model (left part) and the procedure of domain adaptation (right part).

2.2 Our Approach for Multi-Corpus Learning

MTL can be done with either *hard* or *soft* parameter sharing of hidden layers [23]. In our work, we use the most common approach of MTL: the hard parameter sharing. In this case, all the datasets are first processed by the shared layers having learnable parameters θ^s. These layers learn a shared representation for all the tasks from all the available input data. The outputs of the shared layers are passed into the task-specific layers with parameters θ^i when the model input corresponds to the data of task t_i.

Our methodology is based on MTL proposed in [18], where a pre-trained BERT model is incorporated. We adapt this model for our task of low-resource hate speech classification and apply it in a multi-corpus scenario. Usually, supervised classification approaches require a large amount of annotated data. By combining several corpora, MCL mitigates the problems of less amount of training data to efficiently train a model and reduces the overfitting problem.

Figure 1 (left part) shows the architecture of our approach. We consider *a corpus as a task*. The number of tasks corresponds to the number of available annotated corpora used to train the model. The MCL model consists of two parts: (a) the shared layers; (b) a set of corpus-specific layers.

Shared layers: The shared layers are shared by all the tasks. We chose the pre-trained BERT model [10] as shared layers. The training samples from all the tasks are passed as input to the shared layers. These layers benefit from an implicit data augmentation as they process the data from all the tasks. This enriches the representations learned by the shared layers.

Corpus-specific layers: The outputs of the shared layers are used as input to the corpus-specific layers. The objective of the corpus-specific layers is to optimize the model for a given corpus.

2.3 Domain Adaptation using Multi-Corpus Learning

The goal of an efficient model is to generalize to unseen data. When the distribution of train and test sets differ (domain shift) [22], the ability of a model trained on one domain to perform on another domain reduces. Supervised domain adaptation techniques allow a model trained on the source domain to adapt to a target domain with a limited amount of labelled data.

The procedure for domain adaptation using the MCL is presented in Figure 1 (right). We first train an MCL model with all the available corpora except one, which is our target corpus for adaptation. Then we adapt the trained MCL model to our target corpus. After adding new corpus-specific layers for the new target corpus, during adaptation, we update the shared layers along with the newly added corpus-specific layers using the target corpus.

3 Experimental Setup

In this section, we briefly describe the considered datasets, the text pre-processing, and the choice of model parameters for our MCL approach.

3.1 Corpora

We consider five widely used hate speech corpora to train our MCL model. Four of these corpora are tweets sampled from Twitter, namely *Davidson* [7], *Founta* [13], *Hateval* [4], and *Waseem* [24]. The fifth corpus is sampled from *Wikipedia* talk pages [26]. We perform binary classification for the Hateval, Waseem and

Wikipedia datasets. The Davidson and Founta corpora are used for the multi-class classification of hate speech. The statistics of these corpora are provided in Table 1.

Table 1: Corpus statistics: number of tweets or comments.

Corpus	Total	Class labels		
		Normal	Abusive	Hateful
Davidson	24.8k	4.2k	19.2k	1.4k
Founta	86.0k	53.8k	27.2k	5.0k
		Non-hateful		Hateful
Hateval	13.0k	7.5k		5.5k
Waseem	10.9k	8.0k		2.9k
		Non-Toxic		Toxic
Wikipedia	159.7k	131.7k		28.0k

Davidson: This dataset is collected by sampling tweets using some keywords from the hatebase lexicon.[1] The corpus is annotated into three classes *neither*, *offensive language*, and *hate speech*. We refer to these classes as *normal*, *abusive*, *hateful*, respectively.

Founta: The Founta corpus has four classes, namely, *normal*, *abusive*, *hateful*, and *spam*. We removed samples labelled as *spam* class, which reduced the size of this dataset from 100k tweets to 86.9k tweets.

Hateval: This corpus was designed for the 'SemEval-2019' shared task. For our study, we have used the English part of the dataset. The corpus is annotated into two classes, namely, *hateful* and *non-hateful*. The corpus provides 9k, 3k, and 1k samples for training, development, and test sets, respectively.

Waseem: This dataset is sampled using keywords containing racial and sexual slurs. This dataset has three classes, *racism*, *sexism*, and *none* with 2.0k, 3.4k, and 11.6k samples, respectively. Due to the filtering strategy of Twitter to remove hateful content, we retrieved only 20, 2.9k, and 8.0k samples for *racism*, *sexism*, and *none* classes, respectively, using the tweet-ids provided by the authors, as in [5]. We refer to the *sexism* class as *hateful*, and the *none* class as *non-hateful*. We discard samples from the *racism* class due to very few samples.

Wikipedia: This corpus contains comments from the user talk pages. We use the 'toxicity' part of the dataset, annotated with five labels - *very toxic*, *toxic*, *neither*, *healthy*, and *very healthy*. Each comment is annotated by approximately ten annotators. We chose to split the corpus into two classes: *toxic* versus *non-toxic* for each comment. We consider the comment as toxic if at least two annotators have labelled the comment as *toxic* or *very toxic*, and if the number of annotations as *toxic* or *very-toxic* is higher than the number of annotations as *healthy* and *very-healthy*. The dataset provides 95.7k, 32.1k, and 31.9k samples for training, development, and test sets, respectively.

[1] `https://www.hatebase.org`

3.2 Dataset Split

For Davidson, Founta, and Waseem, we randomly split the datasets into three parts, training, validation, and test sets, each containing 70%, 10%, and 20%, respectively. For Hateval and Wikipedia corpora, we utilize the splits provided by the datasets. The training set is used to train the model, the validation set to adjust the model parameters, and the test set to evaluate the model performance.

3.3 Input Pre-processing

For Twitter corpora, the user handles are changed to '@USER'. The '#' symbol in the hashtag is removed, and the multi-word hashtags are split based on the presence of the uppercase characters. For example, '#leaveThisPlace' is changed to 'leave This Place'. The tweets containing the term 'RT' indicating re-tweet are also removed. For all the datasets, we remove all numbers, newlines, and special characters except '.', ',', '!', '?', and *apostrophes*. The repeated occurrences of the same special character are reduced to a single one. All the URLs and emoticons are also removed. Finally, all the data is lower-cased.

3.4 Multi-Corpus Model and Training Description

The shared layers consist of the pre-trained English 'bert-base-uncased' model. We use five sets of corpus-specific layers as we have five corpora. The output of the [CLS] token of the BERT model is used as input to the corpus-specific layers. We define a single dense layer with 768 hidden units as our corpus-specific layer. The outputs of this hidden layer are passed through a softmax classifier with the number of units equal to the number of classes of the respective corpus. We use ReLU[1] activation for the dense layers, a learning rate of $1e-5$, Adam optimizer [16], a maximum of 30 epochs, mini-batch size of 32, and early stopping.

Compared to the standard way of a random selection of training samples for a mini-batch, we perform a task-specific selection of mini-batches. All the samples of a given mini-batch are extracted from a single corpus. For example, given two datasets, for one mini-batch, we select a fixed number of random training samples from one dataset, and for the other mini-batch we select the same number of samples from the other dataset. This procedure is repeated for the remaining mini-batches. When one corpus has fewer samples compared to another corpus, the samples from the smaller dataset are repetitively selected. This kind of mini-batch selection ensures that the multi-task learning model is trained with an equal number of samples from all the corpora. Our source code for MCL is made available[2].

4 Results and Discussion

In this section, we report the classification performance. We compute average macro-F1 and standard deviation over five runs of the model with different random initialization.

[2] `https://gitlab.inria.fr/adsa/multitasklearning_lrec`

4.1 Multi-Corpus Learning

We evaluate the following configurations:

Single-Corpus Learning (SCL): We create five models, one for each corpus. Each model is obtained by fine-tuning the pre-trained BERT on the training part of the corresponding corpus. The test set is used to evaluate the model.

Multi-Corpus Learning (MCL): We create a single model using all the training corpora (see Section 2.2). The test set of each corpus is used separately to evaluate the model.

Multi-Corpus Learning with corpus-specific fine-tuning (MCL$_{finetuned}$): The model learned using the MCL setup, is further fine-tuned using five target corpora. We create five models, one for each corpus. In the beginning, one MCL model is learned. Then, this model is fine-tuned using five training corpora separately. This results in five models. The test part of each corpus is used to evaluate the corresponding fine-tuned MCL model.

The results obtained on the five corpora are presented in Table 2.

Table 2: Macro-F1 results on test sets for the different approaches. Average column presents the mean on five test corpora.

	Davidson	Founta	Hateval	Waseem	Wikipedia	Average
SCL	76.0 ± 0.6	75.8 ± 0.4	49.3 ± 1.8	84.0 ± 0.5	86.9 ± 0.1	74.4
MCL	76.3 ± 1.1	75.5 ± 0.2	50.4 ± 3.0	84.1 ± 0.4	86.4 ± 0.2	74.5
MCL$_{finetuned}$	75.7 ± 1.0	75.8 ± 0.7	52.1 ± 2.6	84.6 ± 0.6	86.7 ± 0.2	**75.0**

We observe that the average macro-F1 obtained for the SCL approach is 74.4%. The average macro-F1 of the MCL approach 74.5% is close to the SCL approach. We note that for the two smaller training corpora (Davidson and Hateval) the performance slightly increased, but for the two larger training corpora (Founta and Wikipedia) the performance marginally reduced, thus showing higher improvements in low-resource corpora. For the MCL$_{finetuned}$ setup, we obtain an average macro-F1 of 75.0%. This shows an improvement compared to SCL and MCL approaches. We observe that all the corpora, except Davidson, benefit from the fine-tuning of the MCL model. This improvement observed for the MCL$_{finetuned}$ approach can be due to the fact that the MCL model is not fully optimized for every considered corpus. Hence, fine-tuning the MCL model on a specific corpus can help.

We would like to highlight that, although we obtain poor classification results on the Hateval dataset, our results are higher than the average macro-F1 of 44.84% obtained by the participants of the SemEval-2019 Task 5 challenge [4].

In the MCL approach, only the shared layers benefit from jointly training with several corpora. Whereas, each corpus-specific layer is trained only with the corpus-specific data. To increase the amount of data used to train corpus-specific layers, we *combine the training sets* of related corpora. To achieve this, we merge the training sets of the three-class datasets together, and similarly, the training sets of the two-class datasets. We represent the combined training sets as {Davidson & Founta} and {Hateval & Waseem & Wikipedia} in Table 3. This

setup reduces the number of corpus-specific layers used for the MCL architecture, and the the number of parameters to train. Compared to the standard MCL approach, which consists of five sets of corpus-specific layers, by combining the training sets, we have only two sets of corpus-specific layers. In this setup, for all the configurations, we fine-tune the MCL model using the combined training sets. However, the model is evaluated on a specific test set and the results are reported separately to allow their comparison with the previous results.

Table 3: Macro-F1 results on test sets for the different approaches by combining tasks. Average column presents the mean on five test corpora.

Train set	{Davidson & Founta}		{Hateval & Waseem & Wikipedia}			Average
Test set	Davidson	Founta	Hateval	Waseem	Wikipedia	
SCL	82.1 ± 4.7	77.2 ± 0.7	39.8 ± 2.3	80.4 ± 1.7	86.7 ± 0.2	73.2
MCL	82.2 ± 3.3	77.7 ± 1.1	42.5 ± 3.6	80.0 ± 1.2	86.4 ± 0.3	73.7
MCL$_{finetuned}$	88.1 ± 1.7	78.4 ± 0.2	42.3 ± 2.5	81.9 ± 0.7	86.0 ± 0.3	**75.3**

Table 3 presents the results obtained using the MCL by combining corpora. For SCL, we obtain better results for Davidson and Founta test sets compared to the SCL approach without combining the training sets (results in Table 2). Perhaps this is because Davidson and Founta datasets have a similar label definition. However, we observe a reduced performance for Hateval and Waseem test sets. This can be due to the fact that abusive speech and toxic speech are close but represent different concepts and bias the system.

The MCL approach by combining the tasks provides a small improvement compared to the SCL approach (73.7% versus 73.2%). Furthermore, for MCL$_{finetuned}$ approach, we obtain the best results (75.3%). In conclusion, we note that corpus-specific fine-tuning of the trained MCL model shows improvements compared to the MCL approach.

4.2 Multi-Corpus Learning in Low-Resource Scenarios

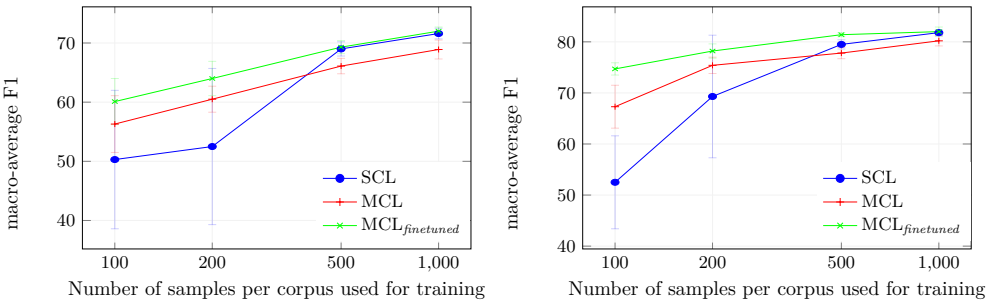
We explore the MCL approach in low-resource training scenarios. We down-sample the available training sets of all the corpora to 100, 200, 500, and 1000 samples. We then perform the training using SCL, MCL, and MCL$_{finetuned}$ approaches on the reduced training data.

Table 4 presents the average macro-F1 on the five datasets in low-resource scenarios. Figure 2 shows the macro-average F1 for SCL, MCL, and MCL$_{finetuned}$ setup for the Founta and Wikipedia test sets. For illustration, we plot the results only for the Wikipedia and Founta datasets, as examples of two-class and three-class classification performance. We obtained similar results for other datasets.

From Table 4 and Figure 2, we can note that MCL and MCL$_{finetuned}$ setups show similar or better results than SCL. However, MCL and MCL$_{finetuned}$ give higher performance gains in very low-resource scenarios. When we use 100 samples for the training sets, we obtain a relative improvement of 16.8% and 23.9% for MCL and MCL$_{finetuned}$, respectively, compared to SCL (61.2%, 57.7% versus 49.4%). For 200 samples, we obtain a relative improvement of 18.5% and 22.3%

Table 4: Average macro-F1 results on test sets for five corpora in low-resource scenarios.

Approaches	Number of training samples			
	100	200	500	1000
SCL	49.4	53.4	67.7	70.2
MCL	57.7	63.3	67.0	69.6
$\text{MCL}_{finetuned}$	**61.2**	**65.3**	**68.7**	**70.6**



(a) Founta (b) Wikipedia

Fig. 2: Macro-average F1 results on test sets for low-resource scenarios.

for MCL and $\text{MCL}_{finetuned}$, respectively (65.3% and 63.3% versus 53.4%). These improvements are statistically significant.

This shows that when the number of available samples is low, the MCL gains from jointly training the model using several datasets. Furthermore, the results also indicate that corpus-specific fine-tuning of the MCL model gives significant performance improvements in low-resource scenarios.

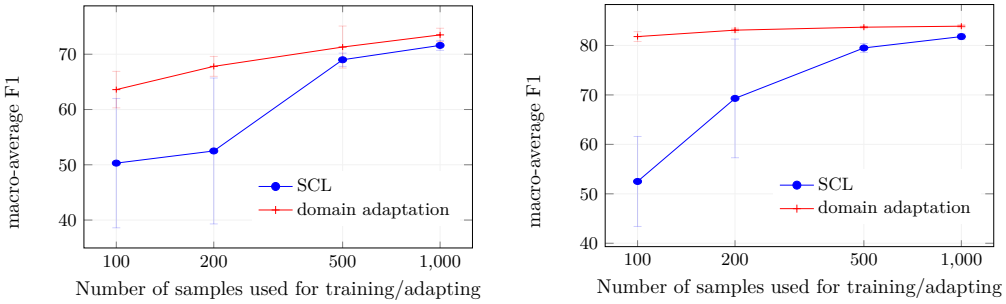
4.3 Domain Adaptation using Multi-Corpus Learning Approach

We perform supervised domain adaptation for hate speech classification as described in Section 2.3. We simulate low-resource scenarios for domain adaptation. We train the MCL model with entire training sets of four tasks. Whereas, for the target corpus, we use only 100, 200, 500, and 1000 training samples. The average macro-F1 results obtained on five target corpora in low-resource scenarios are presented in Table 5. The results of domain adaptation are compared against low-resource single-corpus learning, where the SCL model is fine-tuned with the varying amount of training data of the target set (same model as in section 4.2). Figure 3 presents the results obtained for domain adaptation for Founta and Wikipedia as target datasets.

Compared to the SCL, for domain adaptation, we obtain a significant relative improvement of 37% and 31.5%, using 100 and 200 training samples for the target datasets, respectively (67.7% versus 49.4% and 70.2% versus 53.4%). The improvement is higher when the amount of available data is lower. This can be because the shared layer of MCL captures information from multiple

Table 5: Average of macro-F1 results for five test datasets as target datasets for domain adaptation: low-resource scenario and all training samples.

Approaches	Low-resource scenario				All training samples for adaptation
	Number of adaptation samples				
	100	200	500	1000	
SCL (without adaptation)	49.4	53.4	67.7	70.2	73.2
MCL domain adaptation	**67.7**	**70.2**	**71.2**	**72.1**	**74.7**



(a) Macro-F1 on **Founta** test set. The MCL model is trained using Davidson, Hateval, Waseem, and Wikipedia training sets. Model adapted using varying amount of Founta training set.

(b) Macro-F1 on **Wikipedia** test set. The MCL model is trained using Davidson, Founta, Hateval, and Waseem training sets. Model adapted using varying amount of Wikipedia training sets.

Fig. 3: Macro-average F1 results for domain adaptation.

related corpora, that can be transferred to a new corpus. Figure 3b shows a considerable amount of improvements in the low-resource domain adaptation for the Wikipedia dataset, although the MCL model was trained with four Twitter datasets. This indicates that the MCL approach can still be helpful when the task is related but the corpora come from different domains. Using the entire training set as a target corpus for domain adaptation, an average macro-F1 of 74.7% is obtained. This result is better than macro-F1 of 73.2% obtained using SCL. Thus, from Table 5, we conclude that domain adaptation in low-resource scenarios gives better performance than the SCL approach.

5 Conclusion

In this article, we explored multi-corpus learning(MCL) for low-resource hate speech classification. Our approach for MCL is based on the paradigm of multi-task learning. Our idea is to utilize the shared layers of MCL to learn a common representation for several corpora, and corpus-specific layers to take into account the corpus-specific characteristics. We showed that the fine-tuning of the MCL model improves the performance compared to the SCL model.

In very low-resource scenarios, the MCL showed significant performance improvement when compared to SCL. We also used the MCL approach to perform

domain adaptation. Compared to fine-tuning a pre-trained BERT, our adaptation approach showed significant improvements, especially when the amount of available adaptation data is very low. Overall, we experimentally demonstrated the efficiency of MCL for low-resource hate speech classification and domain adaptation.

6 Acknowledgments

This work was funded by the M-PHASIS project supported by the French National Research Agency (ANR) and German National Research Agency (DFG) under contract ANR-18-FRAL-0005. Experiments presented in this article were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations.

References

1. Agarap, A.F.: Deep learning using rectified linear units (ReLU). arXiv preprint arXiv:1803.08375 (2018)
2. Alwosheel, A., van Cranenburgh, S., Chorus, C.G.: Is your dataset big enough? sample size requirements when using artificial neural networks for discrete choice analysis. Journal of choice modelling **28**, 167–182 (2018)
3. Badjatiya, P., Gupta, S., Gupta, M., Varma, V.: Deep learning for hate speech detection in tweets. In: Proceedings of the 26th International Conference on World Wide Web Companion. pp. 759–760 (2017)
4. Basile, V., Bosco, C., Fersini, E., Nozza, D., Patti, V., Rangel Pardo, F.M., Rosso, P., Sanguinetti, M.: 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. pp. 54–63. Association for Computational Linguistics (Jun 2019)
5. Bodapati, S., Gella, S., Bhattacharjee, K., Al-Onaizan, Y.: Neural word decomposition models for abusive language detection. In: Proceedings of the Third Workshop on Abusive Language Online. pp. 135–145 (2019)
6. Caruana, R.: Multitask learning. Machine learning **28**(1), 41–75 (1997)
7. Davidson, T., Warmsley, D., Macy, M., Weber, I.: Automated hate speech detection and the problem of offensive language. In: Proceedings of the International Association for the AAAI Conference on Web and Social Media. vol. 11, pp. 512–515 (2017)
8. Del Vigna, F., Cimino, A., Dell'Orletta, F., Petrocchi, M., Tesconi, M.: Hate me, hate me not: Hate speech detection on facebook. In: Proceedings of the First Italian Conference on Cybersecurity. pp. 86–95 (2017)
9. Delgado, R., Stefancic, J.: Hate speech in cyberspace. Wake Forest L. Rev. **49**, 319 (2014)
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: 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). pp. 4171–4186 (2019)

11. Florio, K., Basile, V., Polignano, M., Basile, P., Patti, V.: Time of your hate: The challenge of time in hate speech detection on social media. Applied Sciences **10**(12), 4180 (2020)
12. Fortuna, P., Soler, J., Wanner, L.: Toxic, hateful, offensive or abusive? What are we really classifying? An empirical analysis of hate speech datasets. In: Proceedings of the 12th Language Resources and Evaluation Conference (LREC). pp. 6786–6794 (2020)
13. Founta, A.M., Djouvas, C., Chatzakou, D., Leontiadis, I., Blackburn, J., Stringhini, G., Vakali, A., Sirivianos, M., Kourtellis, N.: Large scale crowdsourcing and characterization of twitter abusive behavior. In: Twelfth International AAAI Conference on Web and Social Media. pp. 491–500 (2018)
14. Gambäck, B., Sikdar, U.K.: Using convolutional neural networks to classify hate-speech. In: Proceedings of the first workshop on abusive language online. pp. 85–90 (2017)
15. Kapil, P., Ekbal, A.: A deep neural network based multi-task learning approach to hate speech detection. Knowledge-Based Systems **210**, 106458 (2020)
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015 (2015)
17. Lee, Y., Yoon, S., Jung, K.: Comparative studies of detecting abusive language on twitter. In: Proceedings of the 2nd Workshop on Abusive Language Online (ALW2). pp. 101–106 (2018)
18. Liu, X., He, P., Chen, W., Gao, J.: Multi-task deep neural networks for natural language understanding. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 4487–4496 (2019)
19. Park, J.H., Fung, P.: One-step and two-step classification for abusive language detection on twitter. In: Proceedings of the First Workshop on Abusive Language Online. pp. 41–45 (2017)
20. Plaza-Del-Arco, F.M., Molina-González, M.D., Ureña-López, L.A., Martín-Valdivia, M.T.: A multi-task learning approach to hate speech detection leveraging sentiment analysis. IEEE Access **9**, 112478–112489 (2021)
21. Rajamanickam, S., Mishra, P., Yannakoudakis, H., Shutova, E.: Joint modelling of emotion and abusive language detection. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 4270–4279 (2020)
22. Ramponi, A., Plank, B.: Neural unsupervised domain adaptation in NLP—A survey. In: Proceedings of the 28th International Conference on Computational Linguistics. pp. 6838–6855 (2020)
23. Ruder, S.: An overview of multi-task learning in deep neural networks. arXiv preprint arXiv:1706.05098 (2017)
24. Waseem, Z., Hovy, D.: Hateful symbols or hateful people? Predictive features for hate speech detection on twitter. In: Proceedings of the NAACL Student Research Workshop. pp. 88–93 (2016)
25. Wiegand, M., Ruppenhofer, J., Kleinbauer, T.: Detection of abusive language: the problem of biased datasets. 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). pp. 602–608 (2019)
26. Wulczyn, E., Thain, N., Dixon, L.: Ex machina: Personal attacks seen at scale. In: Proceedings of the 26th International Conference on World Wide Web. pp. 1391–1399 (2017)
27. Zhang, Y., Yang, Q.: A survey on multi-task learning. IEEE Transactions on Knowledge and Data Engineering pp. 1–1 (2021)