Go Simple and Pre-Train on Domain-Specific Corpora: On the Role of Training Data for Text Classification

Aleksandra Edwards† **Jose Camacho-Collados**†
Hélène de Ribaupierre† **Alun Preece**‡
†School of Computer Science and Informatics, Cardiff University, United Kingdom
‡Crime and Security Research Institute, Cardiff University, United Kingdom
{edwardsai,camachocolladosj,deribaupierreh,preecead}@cardiff.ac.uk

Abstract

Pre-trained language models provide the foundations for state-of-the-art performance across a wide range of natural language processing tasks, including text classification. However, most classification datasets assume a large amount labeled data, which is commonly not the case in practical settings. In particular, in this paper we compare the performance of a light-weight linear classifier based on word embeddings, i.e., fastText (Joulin et al., 2017), versus a pre-trained language model, i.e., BERT (Devlin et al., 2019), across a wide range of datasets and classification tasks. In general, results show the importance of domain-specific unlabeled data, both in the form of word embeddings or language models. As for the comparison, BERT outperforms all baselines in standard datasets with large training sets. However, in settings with small training datasets a simple method like fastText coupled with domain-specific word embeddings performs equally well or better than BERT, even when pre-trained on domain-specific data.

1 Introduction

Language models pre-trained on large amounts of text corpora form the foundation of today's NLP (Gururangan et al., 2020; Rogers et al., 2020). They have proved to provide state-of-the-art performance against most standard NLP benchmarks (Wang et al., 2019a; Wang et al., 2019b). However, these models require large computational resources that are not always available and have important environment implications (Strubell et al., 2019). Moreover, there is limited research in the applicability of pre-trained models in classification tasks with small amount of labelled data. Some related studies (Lee et al., 2020; Nguyen et al., 2020; Huang et al., 2019; Alsentzer et al., 2019) investigate whether it is helpful to tailor a pre-trained model to the domain while others (Sun et al., 2019; Chronopoulou et al., 2019; Radford et al., 2018) analyse methods for fine-tuning BERT to a given task. However, these studies perform evaluation on a limited range of datasets and classification models and do not consider scenarios with limited amounts of training data.

In particular, this paper aims to estimate the role of labeled and unlabeled data for supervised text classification. Our study is similar to Gururangan et al. (2020) where they investigate whether it is still helpful to tailor a pre-trained model to the domain of a target task. In this paper, however, we focus our evaluation on text classification and compare different types of classifiers on different domains (social media, news and reviews). Unlike other tasks such as natural language inference or question answering that may require a subtle understanding, feature-based linear models are still considered to be competitive in text classification (Kowsari et al., 2019). However, to the best of our knowledge there has not been an extensive comparison between such methods and newer pre-trained language models. To this end, we compare the light-weight linear classification model fastText (Joulin et al., 2017), coupled with generic and corpus-specific word embeddings, and the pre-trained language model BERT (Devlin et al., 2019), trained on generic data and domain-specific data. Specifically, we analyze the effect of training size over the performance of the classifiers in settings where such training data is limited, both in few-shot

Proceedings of the 28th International Conference on Computational Linguistics, pages 5522–5529
Barcelona, Spain (Online), December 8-13, 2020

scenarios with a balanced set and keeping the original distributions. In both cases, our results show that a large pre-trained language model may not provide significant gains over a linear model that leverage word embeddings, especially when these belong to the given domain.

2 Supervised Text Classification

Given a sentence or a document, the task of text classification consists of associating it with a label from a pre-defined set. For example, in a simplified sentiment analysis setting the pre-defined labels could be positive, negative and neutral. In the following we describe standard linear methods and explain recent techniques based on neural models that we compare in our quantitative evaluation.

2.1 Supervised machine learning models

Linear models. Linear models such as SVMs or logistic regression coupled with frequency-based hand-crafted features have been traditionally used for text classification. Despite their simplicity, they are considered a strong baseline for many text classification tasks (Joachims, 1998; McCallum et al., 1998; Fan et al., 2008), even more recently on noisy corpora such as social media text (Çöltekin and Rama, 2018; Mohammad et al., 2018). In general, however, these methods tend to struggle with OOV (Out-Of-Vocabulary) words, fine-grained distinctions and unbalanced datasets. FastText (Joulin et al., 2017), which is the model evaluated in this paper, partially addresses these issues by integrating a linear model with a rank constraint, allowing sharing parameters among features and classes, and integrates word embeddings that are then averaged into a text representation.

Neural models. Neural models can learn non-linear and complex relationships which makes them a preferable method for many NLP tasks such as sentiment analysis or question answering (Sun et al., 2019). In particular, LSTMs, sometimes in combination with CNNs for text classification (Xiao and Cho, 2016; Pilehvar et al., 2017), enable capturing long-range dependencies in a sequential manner where data is read from only one direction (referred to as the 'unidirectionality constraint'). Recent state-of-the-art language models, such as BERT (Devlin et al., 2019), overcome the unidirectionality constraint by using transformer-based masked language models to learn pre-trained deep bidirectional representations. These pre-trained models leverage generic knowledge on large unlabeled corpora that can then be fine-tuned on the specific task by using the pre-trained parameters. BERT, which is the pre-trained language model tested in this paper, has been proved to provide state-of-the-art results in most standard NLP benchmarks (Wang et al., 2019b), including text classification.

2.2 Pre-trained word embeddings and language models

Most state-of-the-art NLP models nowadays use unlabeled data in addition to labeled data to improve generalization (Goldberg, 2016). This comes in the form of word embeddings for fastText and a pre-trained language model for BERT.

Word embeddings. Word embeddings represent words in a vector space and are generally learned from shallow neural networks trained on text corpora, with Word2Vec (Mikolov et al., 2013) being one of the most popular and efficient approaches. A more recent model based on the Word2Vec architecture is fastText (Bojanowski et al., 2017), where words are additionally represented as the sum of character n-gram vectors. This allowed building vectors for rare words, misspelt words or concatenations of words.

Language models. A limitation to the word embedding models described above is that they produce a single vector of a word despite the context in which it appears. In contrast, contextualized embeddings such as ELMo (Peters et al., 2018) or BERT (Devlin et al., 2019) produce word representations that are dynamically informed by the words around them. The main drawback of these models, however, is that they are computationally very demanding, as they are generally based on large transformer-based language models (Strubell et al., 2019).

3 Experimental Setting

Datasets. For our experiments we selected a suite of datasets with different domains and nature. These are: SemEval 2016 task on sentiment analysis (Nakov et al., 2019), SemEval 2018 task on emoji prediction (Barbieri et al., 2018), AG News (Zhang et al., 2015), Newsgroups (Lang, 1995) and IMDB (Maas et al., 2011). The main features and statistics of each dataset are summarized in Table 1.[1]

Dataset	Task	Domain	Type	Avg tokens	Labels	# Train	# Dev	# Test
SemEval-16 (SA)	Sentiment analysis	Twitter	Sentence	20	3	5,937	1,386	20,806
SemEval-18 (EP)	Emoji prediction	Twitter	Sentence	12	20	500,000	1,000	49,998
AG News	Topic categorization	News	Sentence	31	4	114,828	624	5,612
20 Newsgroups	Topic categorization	Newsgroups	Document	285	20	11,231	748	6,728
IMDB	Polarity detection	Movie reviews	Document	231	2	28,000	2,560	23,041

Table 1: Overview of the classification datasets used in our experiments.

Comparison models. As mentioned in Section 2, our evaluation is focused on fastText (Joulin et al., 2017, FT) and BERT (Devlin et al., 2019). For completeness we include a simple baseline based on frequency-based features and a suite of classification algorithms available in the Scikit-Learn library (Pedregosa et al., 2011), namely Gaussian Naive Bayes (GNB), Logistic Regression and Support Vector Machines (SVM). Of the three, the best results were achieved using Logistic Regression, which is the model we include in this paper as a baseline for our experiments.

Training. As pre-trained word embeddings we downloaded 300-dimensional fastText embeddings trained on Common Crawl (Bojanowski et al., 2017). In order to learn domain-specific word embedding models we used the corresponding training sets for each dataset, except for the Twitter datasets where we leveraged an existing collection of unlabeled tweets from October 2015 to July 2018 to train 300-dimensional fastText embeddings (Camacho Collados et al., 2020). Word embeddings are then fed as input to a fastText classifier where we used default parameters and softmax as the loss function. As for BERT, we fine-tune it for the classification task using a sequence classifier, a learning rate of 2e-5 and 4 epochs. In particular, we made use of BERT's Hugging Face default transformers implementation for classifying sentences (Wolf et al., 2019) and the hierarchical principles described in Pappagari et al. (2019) for pre-processing long texts before feeding them to BERT. We used the generic base uncased pre-trained BERT model and BERT-Twitter[2], both from Hugging Face (Wolf et al., 2019).

Evaluation metrics. We report results based on standard micro and macro averaged F1 (Yang, 1999). In our setting, since system provide outputs for all instances, micro-averaged F1 is equivalent to accuracy.

4 Analysis

We perform two main types of analysis. First, we look at the effect of training size over the classifier's performance by randomly sampling different sized subsets from the original labeled datasets (Section 4.1). Then, we perform a few-shot experiment where we compare classifier's performance on different sizes of balanced subsets of the training data (Section 4.2).

4.1 Effect of training size

Table 2 shows the results with different sizes of training data randomly extracted from the training set. Surprisingly, classification models based on corpus-trained embeddings achieve higher performance with less labelled data compared to the classifier based on pre-trained contextualised models. However, for cases with more than 5,000 training samples, the performance of fine-tuned BERT significantly outperforms fastText corpus-based classifier, especially when domain-trained BERT model (i.e., BERT (Twitter)) is used. Further to that, the fine-tuned model performance improves at a higher rate than the classifier based on corpus-trained embeddings for training sets with more than 2,000 instances. For instance, for the SE-18 dataset, fastText with domain embeddings improves 0.112 micro-F1 points when the entire dataset is used with respect to using only 200 instances, while BERT-Twitter provides a 0.360 absolute improvement. In contrast, fastText with pre-trained embeddings performs similarly to the baseline. This shows the advantage for pre-trained models to be fine-tuned to the given domain and task.

[1] *# Split* (e.g. *# Train*) in the table indicates the number of instances in the given dataset split.

[2] BERT-Twitter model is available from: `https://huggingface.co/ssun32/bert_twitter_turkle#`

	Model	**Micro-F1**						**Macro-F1**					
		200	**500**	**1000**	**2000**	**5000**	**ALL**	**200**	**500**	**1000**	**2000**	**5000**	**ALL**
SE-16 (SA)	Baseline	.399	.405	.430	.447	.476	.483	.360	.390	.410	.430	.430	.460
	FT (general)	.423	.453	.460	.478	.528	.530	.393	.446	.455	.470	.480	.490
	FT (domain)	**.463**	**.487**	.490	.497	.542	.560	**.446**	**.481**	.484	.490	.500	.520
	BERT (general)	.381	.438	**.547**	**.546**	.600	**.611**	.300	.400	**.530**	**.540**	.580	.600
	BERT (Twitter)	.422	.461	.527	.544	**.603**	**.611**	.330	.450	.520	**.540**	**.590**	**.610**
SE-18 (EP)	Baseline	.108	.116	.133	.139	.154	.201	.100	.110	.130	.140	.150	.190
	FT (general)	.109	.120	.130	.136	.194	.258	.084	.109	.125	.130	.150	.220
	FT (domain)	**.149**	**.153**	**.160**	**.166**	.219	.262	**.108**	**.135**	**.151**	**.160**	**.180**	.220
	BERT (general)	.040	.097	.106	.120	.261	.380	.010	.030	.050	.070	.120	.280
	BERT (Twitter)	.082	.102	.134	.127	**.291**	**.400**	.030	.060	.080	.110	.150	**.380**
AG News	Baseline	.665	.788	.812	.840	.854	.883	.620	.750	.780	.800	.820	.860
	FT (general)	.609	.761	.799	.836	.885	.901	.548	.720	.758	.800	.860	.877
	FT (domain)	**.857**	**.886**	.884	.889	.902	.905	**.831**	**.858**	.857	.860	.880	.881
	BERT (general)	.600	.856	**.910**	**.910**	**.922**	**.954**	.390	.830	**.880**	**.880**	**.900**	**.940**
20 Newsgroups	Baseline	.323	.401	.453	.495	.512	.534	.310	.390	.450	.490	.510	.530
	FT (general)	.311	.409	.510	.567	.620	.633	.275	.398	.490	.560	.610	.624
	FT (domain)	**.458**	**.533**	**.583**	**.621**	.636	.645	**.440**	**.520**	**.573**	**.610**	.630	.630
	BERT (general)	.079	.192	.485	.637	**.700**	**.714**	.040	.110	.420	.590	**.670**	**.700**
IMDB	Baseline	.770	.787	.810	.835	.843	.857	.770	.787	.810	.835	.841	.857
	FT (general)	.750	.770	.811	.845	.859	.878	.750	.771	.811	.845	.859	.878
	FT (domain)	**.814**	**.815**	**.836**	**.862**	**.871**	.879	**.814**	**.815**	**.836**	**.862**	**.871**	.879
	BERT (general)	.543	.783	.834	.850	.850	**.881**	.543	.783	.834	.850	.850	**.881**
AVERAGE	Baseline	.453	.499	.528	.551	.568	.592	.432	.485	.516	.539	.550	.579
	FT (general)	.440	.503	.542	.572	.617	.640	.410	.489	.528	.561	.592	.618
	FT (domain)	**.548**	**.575**	**.591**	.607	.634	.650	**.528**	**.562**	**.580**	**.596**	.612	.626
	BERT (general)	.344	.473	.576	**.613**	**.667**	**.708**	.257	.431	.543	.586	**.624**	**.680**

Table 2: Results by training size: 200, 500, 1000, 2000, 5000 instances and entire training set (ALL), where each subset is extracted from the larger subset.

Sentences vs. documents. In order to avoid confounds such as the type of input data in each of the experiments, we filter the results by sentences and documents (see Table 1 for the actual split of datasets in each category). Figure 1 shows the results for this experiment. As can be observed, training set size affects similarly for both types of input, with BERT being especially sensitive to the training data size.

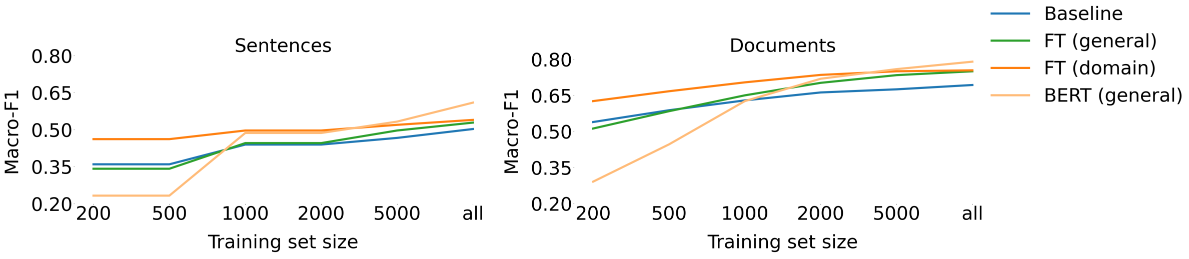

Figure 1: Average Macro-F1 results with randomly sampled training data; the classification datasets are split by type of text: sentence or document.

4.2 Few-shot experiment

A few-shot comparison between the performance of classifiers based on balanced data is shown in Table 3.[3] We balance the dataset for a few shot experiments to ensure the occurrence of instances for all labels within the training set even for datasets with 20 labels when 5-shot and 10-shot experiments are performed. Further, we look at the effect of balanced training data over the classifiers performance. The results show that balancing the dataset lead to improvements in the classification performance with limited training data, especially for BERT. For example, using a subset of 1,000 training instances for

[3] The full balanced set is built by removing random training instances based on the least frequent label's number of instances.

20 Newsgroups corpus, the macro-F1 for random sampled data is 0.42 while the macro-F1 for balanced data (i.e., 50 instances per label) is 0.556.

	2 instances per label				**5 instances per label**				**10 instances per label**			
	FT(gen)	FT(dom)	BERT(gen)	BERT(T)	FT(gen)	FT(dom)	BERT(gen)	BERT(T)	FT(gen)	FT(dom)	BERT(gen)	BERT(T)
SE-16(SA)	.330	**.390**	.230	.320	.370	**.390**	.240	.370	.352	**.384**	.200	.370
SE-18(EP)	.050	**.060**	.020	.030	.080	**.100**	.020	.040	.090	**.110**	.020	.050
AG News	.390	**.700**	.130	-	.520	**.810**	.200	-	.643	**.815**	.410	-
20 News	.090	.200	.010	-	.230	**.430**	.030	-	.294	**.473**	.030	-
IMDB	.411	**.556**	.492	-	.500	**.643**	.547	-	.414	**.567**	.492	-
AVERAGE	.254	**.381**	.176	-	.340	**.475**	.207	-	.359	**.470**	.230	-
	20 instances per label				**50 instances per label**				**Full balanced dataset**			
	FT(gen)	FT(dom)	BERT(gen)	BERT(T)	FT(gen)	FT(dom)	BERT(gen)	BERT(T)	FT(gen)	FT(dom)	BERT(gen)	BERT(T)
SE-16(SA)	.356	**.406**	.320	.370	.416	**.466**	.340	.420	.510	.530	**.610**	.570
SE-18(EP)	.096	**.126**	.020	.070	.121	**.144**	.060	.100	.160	.170	.200	**.280**
AG News	.686	**.838**	.680	-	.752	**.845**	.740	-	.860	.880	**.940**	-
20 News	.406	**.537**	.090	-	.499	**.568**	.500	-	.620	.640	**.680**	-
IMDB	.496	**.641**	.504	-	.660	**.707**	.556	-	.870	.880	**.882**	-
AVERAGE	.408	**.510**	.323	-	.489	**.546**	.439	-	.604	.620	**.662**	-

Table 3: Few-shot Macro-F1 classification results (*gen* refers to general and *dom* refers to domain).

Similarly to the experiments with randomized data samples, fastText based on corpus-trained embeddings is the best performing classification model for very small amounts of balanced labeled data (see Figure 2). However, as the amount of training data increases, BERT model outperforms fastText on average by 0.0442%. As in the previous experiment, the classification model based on pre-trained embeddings perform poorly compared to the corpus-trained embeddings and models fine-tuned to the task. Further, BERT (Twitter) leads to significant improvements over BERT when only 10 instances per label are used (i.e., for SE-16, BERT (Twitter) has macro-F1 = 0.370, similar to domain-based fastText with macro-F1 = 0.384 versus base BERT with macro-F1 = 0.200).

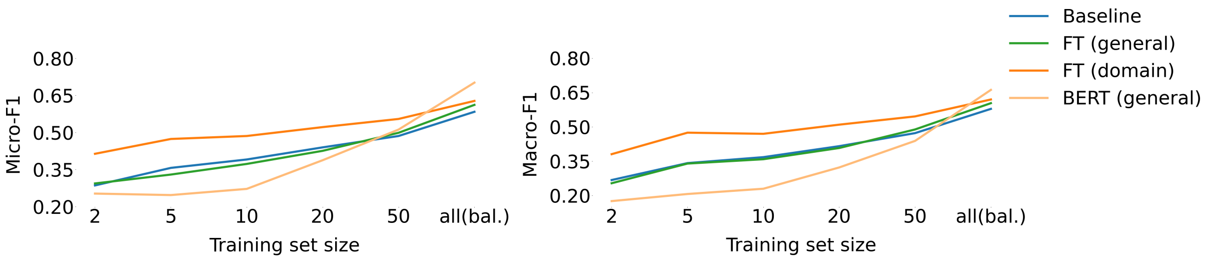

Figure 2: Experiments with balanced data - Micro-F1 results (left), Macro-F1 results (right)

.

5 Conclusion and Future Work

In this paper, we analyzed the role of training and unlabeled domain-specific data in supervised text classification. We compared both linear and neural models based on transformer-based language models. In settings with small training data, a simple method such as fastText coupled with domain-specific word embeddings appear to be more robust than a more data-consuming model such as BERT, even when BERT is pre-trained on domain-relevant data. However, the same classifier with generic pre-trained word embeddings does not perform consistently better than a traditional frequency-based linear baseline.[4] BERT, pre-trained on domain-specific data (i.e., Twitter) leads to improvements over generic BERT, especially for few-shot experiments. For future work it would be interesting to further delve into the role of unlabeled data in text classification, both in terms of word embeddings (e.g., by making use of meta-embeddings (Yin and Schütze, 2016)) and the data used to train language models (Gururangan et al., 2020). Moreover, this quantitative analysis could be extended to more classification tasks and different models, e.g., larger language models such as RoBERTa (Liu et al., 2019) and GPT-3 (Brown et al., 2020), which appear to be more suited to few-shot experiments.

[4] In the appendix we include a qualitative analysis comparing generic and domain-specific word embeddings.

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

Francesco Barbieri, Jose Camacho-Collados, Francesco Ronzano, Luis Espinosa-Anke, Miguel Ballesteros, Valerio Basile, Viviana Patti, and Horacio Saggion. 2018. SemEval-2018 Task 2: Multilingual Emoji Prediction. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, United States. Association for Computational Linguistics.

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

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Jose Camacho Collados, Yerai Doval, Eugenio Martínez-Cámara, Luis Espinosa-Anke, Francesco Barbieri, and Steven Schockaert. 2020. Learning cross-lingual word embeddings from twitter via distant supervision. In *Proceedings of ICWSM*.

Alexandra Chronopoulou, Christos Baziotis, and Alexandros Potamianos. 2019. An embarrassingly simple approach for transfer learning from pretrained language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2089–2095, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Çağrı Çöltekin and Taraka Rama. 2018. Tübingen-oslo at semeval-2018 task 2: Svms perform better than rnns in emoji prediction. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 34–38.

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

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874.

Yoav Goldberg. 2016. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2019. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, April.

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. 2019. Text classification algorithms: A survey. *Information*, 10(4):150.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339, Tahoe City, California.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

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

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics.

Andrew McCallum, Kamal Nigam, et al. 1998. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2019. Semeval-2016 task 4: Sentiment analysis in twitter. *arXiv preprint arXiv:1912.01973*.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Raghavendra Pappagari, Piotr Zelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. 2019. Hierarchical transformers for long document classification. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 838–844. IEEE.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

Mohammad Taher Pilehvar, Jose Camacho-Collados, Roberto Navigli, and Nigel Collier. 2017. Towards a seamless integration of word senses into downstream nlp applications. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1857–1869.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *arXiv preprint arXiv:2002.12327*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy, July. Association for Computational Linguistics.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Proceedings of NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of ICLR*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Yijun Xiao and Kyunghyun Cho. 2016. Efficient character-level document classification by combining convolution and recurrent layers. *arXiv preprint arXiv:1602.00367*.

Yiming Yang. 1999. An evaluation of statistical approaches to text categorization. *Information retrieval*, 1(1-2):69–90.

Wenpeng Yin and Hinrich Schütze. 2016. Learning word meta-embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1351–1360.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

A Analysis word embeddings: Coverage and nearest neighbours

A comparison between the number of out-of-vocabulary (OOV) words for the test datasets between the generic and the domain-trained models (see Table 4) shows that the domain-trained model vocabularies have a larger number of OOV words for the test set than the bigger more generic models except SemEval-18 dataset (Emoji Prediction).

Dataset	# Tokens	# OOV domain	# OOV general
SemEval-16 (SA)	30,467	12,668 (41.6%)	10,558 (34.7%)
SemEval-18 (EP)	66,294	36,846 (55.5%)	37,335 (56.3%)
AG News	23,024	7,766 (33.7%)	4,712 (20.5%)
20 Newsgroups	79,343	39,056 (49.2%)	27,970 (35.2%)
IMDB	105,240	54,395 (51.6%)	36,223 (34.4%)

Table 4: Number of tokens and OOV tokens (for the domain-specific and general-domain word embeddings models) per test set.

However, the generic domain embeddings tend to fail to represent the meaning of more domain-specific words, which may explain their lower performance. This is confirmed by the nearest neighbour analysis (see Table 5) which showed that the generic domain embeddings do not provide accurate representations of more technical words such as *'Windows'* and *'Sun'*. In the IMDB reviews, words such as *'Toothless'*, used within a very specific context are also not correctly represented by the generic model. Moreover, tweets are rich with abbreviations which have domain-specific meaning such as *'SF'* referring to *'San Francisco'*.

Vocabulary	Token	FT (domain)	FT (generic)
Twitter	SF	San Francisco	SciFi
	killing	killin'em	murdering, slaughtering
	arsenal	arsenal fc	armoury
AG News	Sun	Microsystems	Sunlight
	Apple	iTunes	Pear
	capsule	spacecraft	pill
20 Newsgroups	Windows	X Windows	Window, Doors
	DOS	DOS 6, DR-DOS	don't
	backdoor	eavesdrop, algorithm	back-door,loophole
IMDB	Toothless	Worthless	Toothlessness
	slow-pacingly	boringly	fast-pacing
	twist	plot twist	spin

Table 5: Examples of words and their nearest neighbour according to the generic and domain-specific word embedding models.

B Code Availability

Code available at: https://gitlab.com/Aleks-Edwards/coling