Empirical analysis on the state of transfer learning for small data text classification tasks using contextual embeddings

Felipe Carvalho
Post Graduate Program in Electrical Engineering
Federal University of Minas Gerais
Belo Horizonte, Minas Gerais 31270–901
felipefrecar93@gmail.com

Cristiano Castro
Post Graduate Program in Electrical Engineering
Federal University of Minas Gerais
Belo Horizonte, Minas Gerais 31270–901
crislcastro@ufmg.br

***Abstract*—Recent developments in the NLP (Natural Language Processing) field have shown that deep transformer based language model architectures trained on a large corpus of unlabeled data are able to transfer knowledge to downstream tasks efficiently through fine-tuning. In particular, BERT and XLNet have shown impressive results, achieving state of the art performance in many tasks through this process. This is partially due to the ability these models have to create better representations of text in the form of contextual embeddings. However not much has been explored in the literature about the robustness of the transfer learning process of these models on a small data scenario. Also not a lot of effort has been put on analysing the behaviour of the two models fine-tuning process with different amounts of training data available. This paper addresses these questions through an empirical evaluation of these models on some datasets when fine-tuned on progressively smaller fractions of training data, for the task of text classification. It is shown that BERT and XLNet perform well with small data and can achieve good performance with very few labels available, in most cases. Results yielded with varying fractions of training data indicate that few examples are necessary in order to fine-tune the models and, although there is a positive effect in training with more labeled data, using only a subset of data is already enough to achieve a comparable performance with traditional non-deep learning models trained with substantially more data. Also it is noticeable how quickly the transfer learning curve of these methods saturate, reinforcing their ability to perform well with less data available.**

***Keywords*—Small data, text classification, NLP, contextual embeddings, representation learning, deep learning**

I. Introduction

There is a well known bias in the literature about the use of large amounts of data as a sufficient condition for the good performance of deep learning models. This is reasonable since training complex models (with a huge amount of parameters) from small data sets are prone to cause overfitting. Nevertheless, the paradigm for the use of deep models on smaller datasets have shifted with transfer learning [1]. Such technique has enabled to train deep models on a large volume of data and transfer part of this knowledge to solve correlated tasks, sometimes with much less data, by using what was learned in the representation scheme obtained on previous training processes.

Recent advancements in the NLP field showed that deep transformer based language models trained on a large corpus of unlabeled data were able to transfer knowledge efficiently to downstream tasks [2], [3], [4], [5], [6]. These models all achieved state of the art results on several NLP tasks through the process of pre training in a unsupervised fashion on a large corpus of data and fine-tuning on a specific task. Besides that, they all have in common the fact they have great capacity, having millions of parameters, they all make use of the attention mechanism [7] and the transformer architecture [8]. They create a new type of embedding, that is deeper and carries more information than its predecessors, called contextual embedding. Such embedding is able to capture the context of a document and use that in a way that each word has a different meaning depending on the sentence it is in. This enables it to keep important aspects of language such as polysemy, sarcasm and several other important characteristics that are context dependent.

However it is not clear how well these models can transfer knowledge when fine-tuned on very small datasets, which imposes a bigger challenge, given the risk of overfitting becomes higher due to their high model capacity. In Devlin at al. [2] it is discussed how BERT (one of the most successful transformer language models) has a higher probability of generating degenerate results when the dataset is small. This is to be expected of a 340M parameter model that has fine-tuned on little data. The small data transfer ability is specially critical due to its practical value since in many scenarios it is common to have very little labeled data available. It is also unexplored in literature how models like BERT perform with different amount of training data of a same dataset during the fine-tuning process. A result in which the model achieves high performance with small subsets of data would be a great indication of the generalization effectiveness of its contextual embeddings.

In this paper, an empirical analysis is conducted on the two most successful transformer based language models, that have achieved state of the art performance on several tasks at the time they were proposed, BERT [2] and XLNet [9]. In

this analysis the robustness of the transfer learning process of these models on a small data scenario will be put to the test by analysing how they perform when fine-tuned on progressively smaller fractions of training data, for the task of text classification. These two models are tested against two baselines, for comparison purposes. The baselines use simpler forms of text representation, bag of words [10] and TF-IDF [11] combined with an SVM-based classifer [12]. Those were chosen as baseline because they represent the standard statistical learning procedure for dealing with text and also due to the contrast between the simplicity of their way of representing text in comparison to contextual embeddings.

The remainder of the paper is organized as follows: Section II discusses some of the previous works that motivate the experiment and how they relate to the empirical study presented in this paper. Section III presents the models used, their differences and similarities and a theoretical standpoint on their expected behaviour, then in Section IV the datasets and experiment setup are presented. In sections V and VI the results and conclusions taken from the experiments are outlined.

II. Related Work

Prior to the advent of BERT and XLNet models, previous attempts at fine tuning language models on relatively small datasets had failed [13], [14], however authors such as Howard et al [15] and Cer et al [16] were able to, each with its own model architecture, fine tune their language models on small datasets with a certain degree of success.

In Howard et al, his model ULMFit uses a bi-LSTM language model optimized with specific learning rate schedules and shows that fine-tuning his model had much better performance that training it from scratch [15], using some datasets with different fractions of training data available. Although this comparison is very important and shows a transfer capability of his model, a better transfer performance versus training from scratch is somewhat expected, since it is well known that models with high capacity trained with little data tend to overfit. Other differences are that the architecture ULMFit uses is very different from the transformer architecture and the model has orders of magnitude fewer parameters. By looking at his achieved results, the transfer curve of his model is not very steep and takes long to saturate [15], which is an indicative of a not as robust pretraining, which is also evidenced by its worse performance in comparison to BERT and XLNet in most cases [2].

In Cer et al [16], the USE model is presented and it is also shown that it has a good transfer capability and, in that case it is compared against other models pre-trained with word2vec embeddings [17] and trained from scratch. The transfer curves shown in USE are steep but its results are not quite as good as BERT or XLNet and require considerable amounts of extra data in order to be achieved.

Despite their results, these earlier studies have motivated our empirical study in order to observe what is the actual behaviour of BERT and XLNet in a scenario of small data and decreasing dataset sizes. This comparison also aims to find about the transfer capabilities of these two models, inspired by the preceding works before mentioned. The obvious differences are that these two particular models have had, to the knowledge of the authors of this paper, no prior work regarding their transfer capabilities in small data scenarios, specially in regards to datasets with fairly distinct characteristics. Also, these models have considerably different architectures than their predecessors. BERT and XLNet models capacity is vastly superior and their number of parameters far surpasses their predecessors, which makes the fine tuning a much more challenging process and, in this paper, the amount of data used to fine-tune is particularly smaller than what has been used in previous experiments [15], [16]. Being state of the art models, the performance of these methods on many full datasets, including ones used in this paper, is well established and obviously superior to other candidates, but their transfering behaviour and ability to operate on a small data scenario is not known and that is, arguably, one of their main possible benefits, given their main contribution is a novel and more robust way of representing text data [2].

III. Models

A. BERT

The BERT model [2] is based on the transformer architecture described in Vaswani et al [8]. The transformer is a model that uses the attention mechanism [7], originally intended for neural machine translation tasks [7], [8] that, at a very high level, learns to focus on which parts of a document/sentence are important. The transformer is purely based on the attention mechanism and uses no type of recurrent network, as described by Vaswani et al [8]. Instead, the transformer uses multiple attention heads and is very efficient in terms of implementation making good use of highly parallelizable hardware such as GPUs [8]. The transformer consists of an encoder and a decoder part, as illustrated in Fig1.

BERT uses only the encoder part of the transformer architecture which is bidirectional by design [8]. In order to train a language model using a bi-directional encoder, which the authors believe is strictly more powerful than a left-to-right or left-to-right right-to-left shallow concatenated language model [2] the authors come with a novel way of performing the pre-training processes called masked language modeling, which is BERTs main contribution. Through masked language modeling, the bidirectional conditioning problem of allowing each word to indirectly "see itself" in a multilayered context is solved, as described by Devlin et al [2], thus making it possible to train a language model while still being bi-directional. Masked language modeling involves masking a percentage of tokens (replacing them by the [MASK] token) that are fed into the LM (language model) and trying to predict those masked tokens. This creates a mismatch between pre-training and fine-tuning, since the actual [MASK] token is never seen during fine-tuning so, in order to mitigate that, some small percentage of tokens that would be masked are replaced with random tokens instead and, in some cases, they

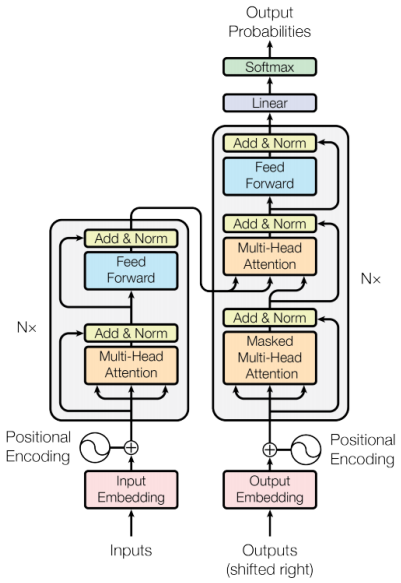

Fig. 1. The transformer - on the left side, the encoder, and on the right side, the decoder [8]

remain unchanged in order to bias the representation to the actual observed word [2]. BERTs input originally takes two sentences separated by the [SEP] token, as it is also intended to be able to transfer knowledge to sentence pair classification tasks [2]. As a secondary task, BERT performs the task of next sentence prediction as well, which is beneficial to some downstream tasks as discussed by Devlin et al [2].

BERT stands for bidirectional encoder representations from transformers, which shows the emphasis on the importance of bidirectionally in LM training, as discussed by the authors [2]. The deep bidirectional representations generated by BERT during its pretraining are powerful contextual embeddings. Contextual embeddings can represent the meaning of the context in each word during its embedding process. As an example, the word bank can refer to a financial institution or a river bank and, through the use contextual embeddings, that confusion would be solved given the words are in different contexts and therefore would have two different representations. This is a very powerful benefit in terms of representational power and allows pre-trained BERT representations to be fine-tuned with just a single output layer [2]. BERTs transformer architecture is also favorable to be fine-tuned, as discussed in Radford et al [6], and it is more robust during fine-tuning than recurrent networks.

BERT has a larger and smaller version, with 340M and 110M parameters respectively, which will both be tested in this paper. Its input is carefully described by Devlin et al [2]. For this work, the [CLS] token, which is part of BERTs input, is specially important since it is the first token in the beginning of every sentence and is used for text classification. The final hidden state corresponding to this token is used as the aggregate sequence representation for classification tasks [2]. This final hidden state is concatenated with a classification layer that consists of a standard Softmax in order to produce the labels for text classification and fine-tuning of the model.

B. XLNet

The way BERT uses masked tokens in its language model pre-training categorizes it as an autoencoder language model, since it reconstructs original data from corrupted input. XLNet changes from the autoencoder architecture to an auto regressive one, while it overcomes some of the AR (autoregressive) language model limitations [9], and attacks some other aspects where, acording to Yang et al [9], BERT is said to be limited. Another significant change XLNet introduces is to use a different transformer architecture, the Transformer-XL [4], that fits XLNets AR framework better since it is itself an AR language model [9]. The Transformer-XL uses segment level recurrence and relative positional embeddings. Those two changes make it able to capture longer contexts, which was one of the limitations of the original transformer, since it had a fixed input size [4], and as a consequence, enables XLNet to do the same [9].

AR (autoregressive) language modeling seeks to estimate the probability distribution of a text corpus with an autoregressive model by factorizing probabilities of words given the previous words through the product rule. The main drawbacks of AR language modeling is that it is not bi directional by design, being able to factorize probabilities in either a forward product or a backward one. As it was seen with BERT, having bi directional context is very important [2]. That motivated the authors of XLNet to propose an AR language model which uses permutation language modeling and keeps benefits of AR modeling while incorporating bi directional context [9].

Permutation language modeling consists of predicting the next token in a way the order of prediction is not necessarily left to right and is sampled randomly instead [9]. This does not break with the AR model, it just changes the order the words are predicted. In other words it permutes the factorization order, not the sequence order. While feeding tokens in a random order for prediction, it is still possible to maintain awareness of their actual right position in the sentence, which is detailed in Yang et al [9]. This is illustrated in Fig 2, where the difference between BERTs autoencoder language model pre-training and XLNets autoregressive language model pre-training can be seen.

According to Yang et al [9], one of the main benefits of the AR model utilization is that it models the join probability of the sentence using the product rule which is something BERT is not able to, and instead, treats each masked token as an independent prediction. That, according to the authors of XLNet, is one of the important advantages it brings [9]. Also, XLNet does not require the use of the [MASK] token specifically, which is argued to also improve performance, since this token in BERT training is only seen during pre-training and not fine tuning, which generates a pre-training/ fine-tuning discrepancy [2], [9].

XLNet also uses an [CLS] token which is used in the same way as BERT for classification. It has a base and large model

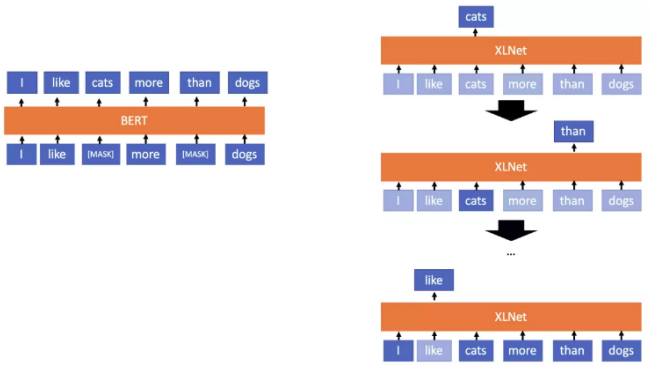

Fig. 2. BERTs autoencoder languange model vs XLNets autoregressive language model [3]

version with the same equivalent number of parameters to BERT, but with the difference XLNet was trained on more data [9].

While the permutation language modeling objective has some benefits, it is a more challenging optimization problem [9]. Also, according to Yang et al., XLNet is able to cover more dependencies than BERT, and the XLNet objective contains more effective training signals [9].

IV. Experiments

In this experiment, both BERT and XLNet are compared on a small data scenario for the task of text classification using progressively smaller fractions of training data. The datasets, training fractions used, test size, total train size, number of classes and average sentence length of each dataset can be seen in Table I.

TABLE I
CHARACTERISTICS OF THE DATASETS USED IN THE EXPERIMENTS

Data	Classes	Avg sentence length	Train size	Test Size	Sampled training sizes
IMDb	2	230	25k	25k	[500 1k 2.5k 5k 10k]
CR	2	19	3016	755	[250 500 1k 2k 3k]
SST-2	2	19	6920	1821	[250 500 1k 1.5k 3k]
MPQA	2	3	8482	2121	[250 500 1k 2k 4k]

The choice of datasets aims to bring diversity, in terms of size and sequence length. The sampled sizes are defined in order to contemplate scenarios with very little data available and progressively increasing it while still being considered small data.

The models compared in the experiment are BERT large and base, which are its 340M and 110M parameter versions respectively, XLNet large and base, that also have 340M and 110M parameters respectively, and two baselines, BOW + SVM and TF-IDF + SVM. The available pre-trained version of BERT and XLNet, provided by its authors [2], [9], are used in this experiment and fine-tuned for text classification in each available dataset.

All models are evaluated on the same test set, for fair comparison (Ref Table I). The datasets had the same pre-processing applied before being fed to the models for training, which involved removing punctuation and lowering the text.

The baselines were trained in a cross-validation process with 3 folds, and had its hyper-parameters searched in a grid-search scheme, including the n-gram range used for the BOW and TF-IDF representations, which used character level tokenization.

BERT and XLNet used the recommended parameters for fine-tuning, given the task of text classification, according to the authors of their respective papers [2], [9]. Those parameters are a batch size of 32, a learning rate of 2e-5, on 3 to 5 epochs. The max sequence length of both BERT and XLNet was set to 256, due to hardware restrictions, which is not a problem in the vast majority of cases. It is also worth mentioning that the base models were trained on a GPU while the large versions were trained on a TPU. No extensive hyperparameter search was performed, since the experiments objective is not to extract the absolute best performance of the methods, but instead to evaluate their behavior on low data scenarios and transfer learning ability.

Each scenario of model, dataset and training fraction was ran multiple times (at least 3) in order to ensure more accurate statistical properties to the experiment. Runs of XLNet and BERT that did not converge or produced degenerate results, as it is possible and described by Devlin et al [2], were not discarded. Instead, the top performance of all models was computed together with the standard deviation of the executions, that help quantify that phenomena. Recording the top results from each scenario also helps debias possible sampling problems during the process, which could compromise any model during the experiments.

V. Results

Results from the experiments proposed in section IV can be seen in Table II and figures 3 and 4. Figure 3 in particular, is the visual representation of the results exposed in Table II, where the best results from each model are shown.

By looking at Figure 3, it is possible to see that the models based on contextual embeddings are able to perform well with low volumes of data and their results are, in most cases, better than the baselines. It is clear that they are able to achieve very high levels of accuracy with few training examples. Although it is not the main focus of this paper, if one was to compare the results achieved in this experiment by BERT and XLNet with results obtained by its predecessors such as USE [16] and ULMfit [15], briefly mentioned in Section II, it is noticeable that they are far better in a small data scenario.

It is also noticeable that there is a positive effect in fine-tuning with more labeled data, however, using a small subset of data is already enough to achieve comparable or better performance than traditional non-deep learning models trained with substantially more data.

BERT base performs very well in all datasets and in all sub-samples of data. BERT large does not and it is possible to see that in scenarios with the lowest availability of training data

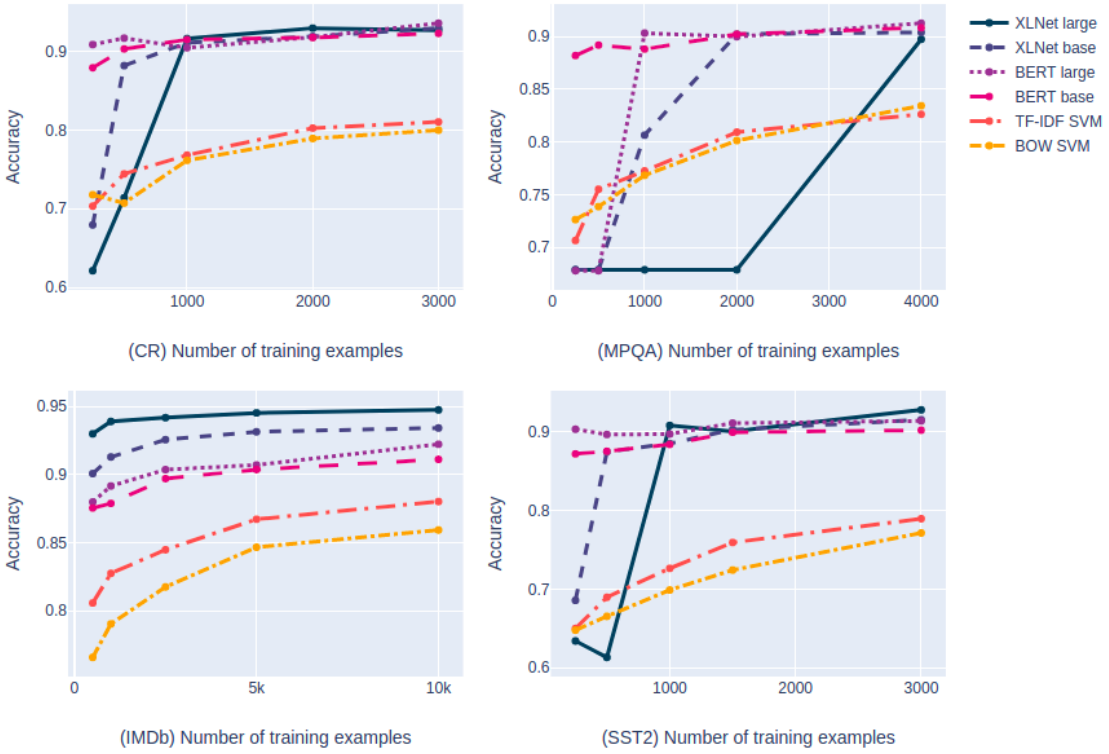

Fig. 3. Accuracy results of the models on the different datasets

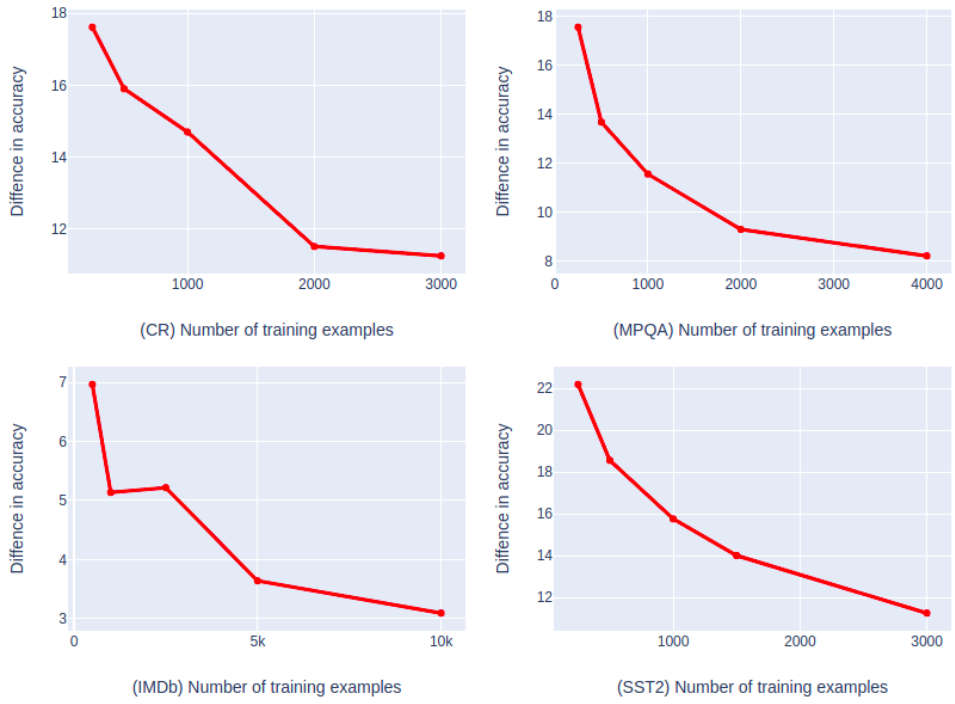

Fig. 4. Difference in accuracy between BERT base, the best performing model in a small data scenario, and the strongest baseline model, TF-IDF + SVM

TABLE II
EXPERIMENT RESULTS - BEST ACCURACY AND STANDARD DEVIATION

CR	**Number of samples**				
Model	**250**	**500**	**1k**	**2k**	**3k**
BERT base	0.8794 0.0022	0.9033 0.0112	0.9152 0.0060	0.9178 0.0026	0.9231 0.0095
BERT large	0.9089 0.1515	0.9171 0.0133	0.9048 0.1631	0.9184 0.0142	0.9361 0.0027
XLNet base	0.6794 0.0440	0.8821 0.0159	0.9112 0.0084	0.9192 0.0037	0.9298 0.0028
XLNet large	0.6211 0.0018	0.7139 0.0674	0.9165 0.0327	0.9298 0.0056	0.9271 0.0037
BOW SVM	0.7178 0.0200	0.7072 0.0272	0.7615 0.0235	0.7894 0.0249	0.8000 0.0072
TFIDF SVM	0.7033 0.0199	0.7443 0.0173	0.7682 0.0094	0.8026 0.0144	0.8105 0.0058
MPQA	**Number of samples**				
Model	**250**	**500**	**1k**	**2k**	**4k**
BERT base	0.8821 0.0043	0.8920 0.0046	0.8882 0.0043	0.9024 0.0053	0.9085 0.0013
BERT large	0.6780 0.0000	0.6780 0.0000	0.9034 0.0944	0.9000 0.1570	0.9128 0.0003
XLNet base	0.6789 0.0000	0.6789 0.0000	0.8066 0.0066	0.9024 0.0020	0.9042 0.0053
XLNet large	0.6789 0.0000	0.6789 0.0000	0.6789 0.0000	0.6789 0.0000	0.8976 0.0333
BOW SVM	0.7265 0.0191	0.7388 0.0097	0.7685 0.0070	0.8015 0.0035	0.8345 0.0095
TFIDF SVM	0.7067 0.0115	0.7553 0.0123	0.7727 0.0074	0.8095 0.0195	0.8264 0.0038
IMDb	**Number of samples**				
Model	**500**	**1k**	**2.5k**	**5k**	**10k**
BERT base	0.8754 0.0038	0.8789 0.0034	0.8970 0.0033	0.9036 0.0016	0.9111 0.0015
BERT large	0.8798 0.0066	0.8916 0.0005	0.9036 0.0012	0.9070 0.0052	0.9222 0.0014
XLNet base	0.9007 0.0097	0.9129 0.0058	0.9256 0.0023	0.9313 0.0009	0.9342 0.0014
XLNet large	0.9286 0.0071	0.9412 0.0011	0.9440 0.0013	0.9452 0.0003	0.9474 0.0004
BOW SVM	0.7659 0.0066	0.7904 0.0069	0.8175 0.0038	0.8467 0.0054	0.8593 0.0033
TFIDF SVM	0.8058 0.0109	0.8275 0.0014	0.8448 0.0098	0.8672 0.0021	0.8802 0.0017
SST2	**Number of samples**				
Model	**250**	**500**	**1000**	**1500**	**3000**
BERT base	0.8720 0.0209	0.8753 0.0248	0.8841 0.0027	0.8995 0.0081	0.9022 0.0023
BERT large	0.9034 0.0098	0.8967 0.0015	0.8973 0.0005	0.9112 0.0323	0.9140 0.0007
XLNet base	0.6858 0.0221	0.8747 0.0069	0.8852 0.0112	0.9028 0.0023	0.9154 0.0007
XLNet large	0.6342 0.0935	0.6133 0.0710	0.9082 0.0896	0.9006 0.0291	0.9280 0.0081
BOW SVM	0.6479 0.0198	0.6655 0.0144	0.6990 0.0190	0.7243 0.0214	0.7715 0.0092
TFIDF SVM	0.6501 0.0338	0.6897 0.0129	0.7265 0.0138	0.7594 0.0121	0.7896 0.0123

BERT large has issues. The difference between BERT base and BERT large is exclusively in the number of parameters [2], therefore it is a reasonable assumption that, given they both are architecturally the same (the only difference being in terms of capacity), by sufficiently regularizing BERT large or maybe by trying extra random restarts of the model, it should be able to achieve good results as well and converge. In Devlin at al. [2], it is discussed how BERT has a higher probability of generating degenerate results when the dataset is small and this is seen in this experiment. BERT large suffers more than BERT base with very small data, as it is to be expected given the bigger capacity makes it more prone to overfitting and also harder to optimize. However, it is worth noticing that reliability is also a factor in a models performance and, therefore, a learned lesson is that, when the dataset is really small, the base version of BERT is, as experiments show, more reliable and recommended.

Nevertheless, despite reliability concerns for some of the models presented, all of them are based on the concept of generating deeper and more robust representations and share a common framework of pre-training and fine-tuning, using transformer based architectures. As it is possible to see by the results of BERT base, that architecture is able to be very data efficient. Noticeably, the MPQA dataset, which has an average sequence length of 3 tokens, still benefits from pre-training and from the use of contextual embeddings (as evidence by the performance of BERT base). This is remarkable given this is a very unfavorable scenario for contextual representations given there is very little context to be captured and, in spite of that, these methods outperform the baselines by a very large margin.

XLNet is also good at dealing with small data. As their authors state and as previously discussed in section III.B XLNet is capable of covering more dependencies than BERT [9]. Noticeably, on the IMDb dataset XLNet thrives, since it contains very long sentences and, even in the smallest subsamples of training data attempted, XLNet outperforms the other models. However, permutation language modeling is a more challenging optimization problem in comparison to BERT, despite its benefits [9]. As evidenced by results on very small data scenarios, XLNet suffers more than the other methods. The empirical results obtained in this paper show that its extra model complexity might present itself as a trade-off when dealing with very small data, in which the cost is lower performance as the available fine-tuning labeled data is less. This can be seen by negative results on datasets other than the IMDb, in which sentences are smaller and labeled examples are fewer. That is not sufficient to say XLNet is not capable of performing well in very small data scenarios, however, given a considerable amount of testing has been made, it is possible to say it is at least not as reliable as BERT.

Figure 4 compares the performance of BERT base, the best performing model in a very small data scenario, and the strongest baseline model, TF-IDF + SVM. It is possible to see a very good transfer behaviour even in scenarios of very small data and it is also noticeable how the model is considerably

better than the baseline in all cases. By looking at Fig 3 and Fig 4 it is remarkable how quickly the transfer learning curves of the transformer based models saturate, which reinforces the finding that these models are able to perform well in scenarios where little data is available.

VI. Conclusion

In this paper, transformer based language models architectures, that have deep text representations in the form of contextual embeddings, more specifically BERT and XLNet, were subject to an empirical analysis that had the objective of evaluating its performance on scenarios with low availability of data for tasks of text classification. The results achieved provide the means of empirically being able to say that the analysed methods are good in small data scenarios and can transfer knowledge well in that same scenario.

It is also possible to see that BERT base was the best performing model with the lowest amount of data available across all datasets. This model has proven to be robust and capable of performing well on small data reliably. This further reinforces the findings that this class of models are good performers in the proposed scenario.

References

[1] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI Global, 2010, pp. 242–264.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[3] keitakurita, "Paper dissected: "xlnet: Generalized autoregressive pretraining for language understanding" explained," 2019. [Online]. Available: https://mlexplained.com/2019/06/30/paper-dissected-xlnet-generalized-autoregressive-pretraining-for-language-understanding-explained

[4] Z. Dai, Z. Yang, Y. Yang, W. W. Cohen, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," *arXiv preprint arXiv:1901.02860*, 2019.

[5] X. Liu, P. He, W. Chen, and J. Gao, "Multi-task deep neural networks for natural language understanding," *arXiv preprint arXiv:1901.11504*, 2019.

[6] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*, 2018.

[7] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[9] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *arXiv preprint arXiv:1906.08237*, 2019.

[10] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: a statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1-4, pp. 43–52, 2010.

[11] S. Robertson, "Understanding inverse document frequency: on theoretical arguments for idf," *Journal of documentation*, vol. 60, no. 5, pp. 503–520, 2004.

[12] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*. Springer, 1998, pp. 137–142.

[13] L. Mou, Z. Meng, R. Yan, G. Li, Y. Xu, L. Zhang, and Z. Jin, "How transferable are neural networks in nlp applications?" *arXiv preprint arXiv:1603.06111*, 2016.

[14] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," in *Advances in neural information processing systems*, 2015, pp. 3079–3087.

[15] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[16] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar *et al.*, "Universal sentence encoder," *arXiv preprint arXiv:1803.11175*, 2018.

[17] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.