Too many classes this semester: Analyzing BERT on a large multi-class classification problem

Maximilian Martin Maurer
maximilian-martin.maurer@ims.uni-stuttgart.de

Urban Knupleš
urban.knuples@ims.uni-stuttgart.de

Abstract

This work investigates the relationship of BERT's performance and number of classes in multi-class text classification for increasing numbers of classes. We conduct experiments on the task of artist classification from lyrics by comparing the performances of fully fine-tuned pre-trained BERT with a classification head, a bag of words (BOW) k-nearest neighbor (kNN) baseline as well as using pre-trained and fine-tuned BERT embeddings as representations for kNN. While we find that for lower numbers of classes, fully fine-tuned BERT outperforms the baseline and the BERT kNN variants for lower numbers of classes, that trend does not hold for greater numbers of classes. For higher numbers of classes, using fine-tuned BERT embeddings with kNN outperforms the other methods. Our code is available on GitHub[1].

1 Introduction

Transformer-based language models such as BERT (Devlin et al., 2019) have shown promising results across tasks in the recent years, from more classical NLP tasks such as POS tagging and morphological prediction (Tsai et al., 2019), to various text classification tasks (Limsopatham, 2021; Xu et al., 2020b). For these tasks, the literature deals with a rather confined number of classes, whereas many classification tasks such as authorship attribution (Fabien et al., 2020) or artist detection from lyrics (Fell and Sporleder, 2014) may require systems to handle hundreds or thousands of classes in real world settings. While multi-label classification with large numbers of non-exclusive labels (Rios and Kavuluru, 2018; Chalkidis et al., 2020) is a relatively well-studied set of problems, that is not necessarily the case for multi-class classification problems on exclusive classes.

[1] https://github.com/mmmaurer/teamlab2022

There are various strategies on how to boost performance for these types of tasks, ranging from task specific ones such as engineering additional features (Fell and Sporleder, 2014) or dividing the problem into sub-problems depending on some hierarchical structure (Shen et al., 2021) to strategies dealing more generally with class/data imbalance (Li et al., 2020) or data augmentation (Xu et al., 2020a). What remains an open question is to which extent performance deteriorates with an increasing number of classes for fine-tuned pre-trained language models such as BERT.

In this work we investigate the relationship of the number of classes and classification performance for fine-tuned BERT on the task of artist detection from lyrics. Following the research questions stated below, we show that while fine-tuning BERT with a classification head is markedly outperforming a k nearest neighbors (kNN) classifier baseline with a set bag of words (BOW) approach and settings with BERT embeddings as input features for a kNN classifier for small amounts of classes, this trend does not hold for larger numbers of classes.

R1: How does performance change for pre-trained language models for increasing numbers of classes?
R2: Can using BERT embeddings with kNN instead of using the classification head for classification improve the results?

The paper's second section discusses relevant methodologies, the third explains the dataset and experimental settings. The fourth presents the results of our experiments. It concludes with a summary and conclusion, and future work.

2 Methods

In this section we provide an overview over the methods we use.

2.1 k nearest neighbors

The kNN algorithm is a non-parametric supervised learning algorithm first described by Fix and Hodges (1951). Given training data tuples of examples X_i and target classes Y_i, $(X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n)$, and a distance metric $||.||$, it assigns the class label of the majority of the k nearest neighbors in terms of the distance metric to an unseen example x. This allows the method, given the right k and distance metric, to perform fairly well with sparse data.

2.2 Distance metrics

For the experiments with kNN we use the Jaccard index for the set-based BOW representations and cosine similarity for real-valued vectorized representations.

Given two sets A and B, the Jaccard index is defined as follows:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Given two feature vectors $A, B \in \mathbb{R}^n$, their cosine similarity is defined as

$$sim_{\cos}(A, B) = \cos(\theta) = \frac{A \cdot B}{||A|| \cdot ||B||}$$

$$= \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Since these capture similarity rather than distance, the distances are defined as $d_J(A, B) = 1 - J(A, B)$ and $d_{\cos}(A, B) = 1 - sim_{\cos}(A, B)$.

2.3 BERT

Bidirectional encoder representations from transformers (BERT) is a transformer-based language representation model introduced by Devlin et al. (2019). As they note, BERT learns deep bidirectional representations by jointly conditioning on contexts on both the left and right side of a given masked (sub-)word in unlabeled text in all layers. This allows the model to learn a contextual representation of (sub-)words without the limitations of directionality. BERT's contextual representations can be used as contextualized embeddings.

Additional to providing contextualized representations, this form of pre-training makes the model easily adaptable to downstream tasks by just adding one fully connected output layer.

3 Experiments

We evaluate the performance of different methods on subsets containing different numbers of classes, using a text classification task of predicting an artist given a lyric. Throughout this section, we are using the term artists and classes interchangeably.

3.1 Dataset

For our experiments we are using a collection of song lyrics with their respective artists[2]. The dataset consists of 57,650 songs, with 643 unique artists in total. It contains no additional meta information, i.e. only the songs textual information is used.

The dataset has an unbalanced distribution of songs per artist as shown in Figure 1. The dataset is pre-split into a train, test and validation set, where the ratio of the set splits are 80/10/10% respectively. Besides tokenization, we do no additional preprocessing of the dataset.

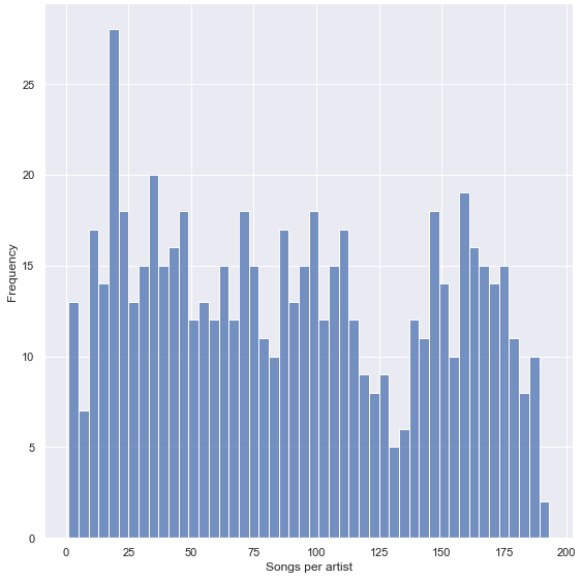

Figure 1: The distribution of songs per artist in our dataset.

3.2 Experimental Setup

To evaluate the performance of different numbers of classes, the main dataset is filtered into subsets that contain the first N artists, where $N \in \{10, 20, 50, 100, 200, 300, 400, 500, 643\}$, as seen in Table 1. The subsets include only songs from

[2]The dataset was provided to us from the lecturers of this course, citing the source from a Kaggle on page https://www.kaggle.com/mousehead/songlyrics/data

those N artists in all the training, testing and validation sets[3]. The same subsets with the respective N number of artists is used throughout all of our experiments for training and testing.

For the experiments using kNN as a classification method, we use a range of different k settings to achieve a better final classification result, where the range is $k \in \{1, 2, ..., 25\}$. For each experiment using different k settings we only report the best overall results.

In all our experiments using BERT we use the 110M-parameter `bert-base-cased` model (Devlin et al., 2019).

We give a more detailed description of our experimental setup on each of the methods described in Section 2. We first describe our two baseline approaches and then continue with BERT in different settings.

Random The first baseline is randomly picking a class from the available artists per amount of classes for each instance in the test set.

kNN + BOW The second baseline represents each song lyric as a set-based BOW representation, i.e. a set of all the unique words in the lyric[4]. We classify each unseen BOW example using the kNN classification algorithm, with Jaccard index as the distance metric. This setting allows comparing neighbouring sets and choosing the majority class.

kNN + Pre-trained BERT embeddings For each lyric we extract the contextual embeddings from the pre-trained BERT model[5] and use kNN to predict classes from unseen examples. The embeddings are collected from the models hidden states in the second to last layer, since they hold the richest contextual representations as reported in Devlin et al. (2019). We calculate the similarity of vectors by using cosine similarity.

Fine-tuning BERT We fully fine-tune the pre-trained BERT model on the task of text classification with an added fully connected output layer. In this setup we do not conduct any additional hyperparameter tuning to improve performance and only use early stopping as a regularization strategy.

kNN + Fine-tuned BERT embeddings This setup matches the one with the pre-trained BERT embeddings and kNN, but instead of using pre-trained embeddings, it uses the contextual embeddings from the fine-tuned model. For each setting with N classes, we use the same model that was trained on that particular subset to preserve class-size consistency.

[3]This approach resulted in our splits not following the original 80/10/10% split.

[4]All transformed BOW lyrics thus have a different set size, which doesn't effect the similarity calculation, because of our choice of a similarity metric.

[5]Some input examples are truncated since the base model allows only a maximum input of 512 tokens. The same limitations apply to all other subsequent BERT experiments.

	random	knn-bow	knn-pre-bert	bert	knn-bert
10	0.098	0.481	0.481	**0.87**	0.614
20	0.035	0.335	0.455	**0.63**	0.44
50	0.022	0.182	0.328	**0.381**	0.28
100	0.009	0.118	0.224	0.219	**0.25**
200	0.004	0.08	0.157	**0.181**	0.174
300	0.003	0.065	0.127	0.155	**0.165**
400	0.002	0.06	**0.116**	0.01	0.055
500	0.002	0.056	0.095	0.065	**0.103**
643	0.001	0.056	0.088	0.073	**0.104**

Table 1: Accuracy of different multi-class classification methods using different class sizes. First column represents the number of classes in each set that the methods are trained/evaluated on. `Knn-bow` refers to BOW representation with a kNN classifier. `Knn-pre-bert` refers to pre-trained BERT embeddings with a kNN classifier. `Bert` refers to fine-tuned BERT model and `knn-bert` using the fine-tuned embeddings with kNN. Best results for each size is highlighted in bold.

4 Results

The results in Figure 2 show the decrease in performance when increasing the number of classes throughout all of our experiments. We see that fine-tuning BERT performs best in almost all cases on class sizes < 300, as seen in Table 1. While fine-tuning BERT is better in these cases, it shows that using fine-tuned BERT embeddings with a kNN classifier achieves better performance on classes ≥ 300. Results also show that using a kNN classifier with both contextual embeddings performs similarly well on class sizes ≥ 500.

Using BOW with a kNN classifier does give the lowest scores to the other methods, but it still shows better performance across classes compared to the random baseline.

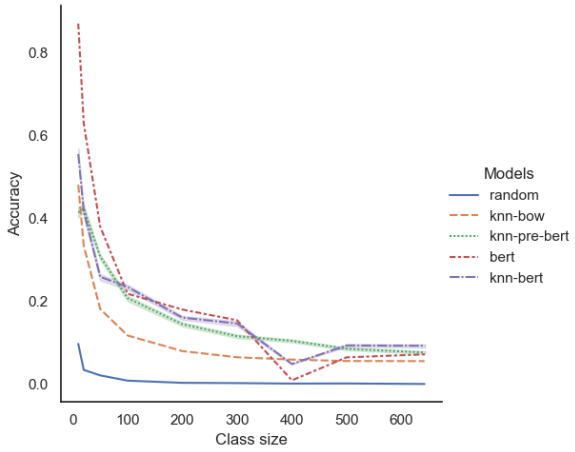

Figure 2: Visual representation of our methods accuracy scores using different class sizes.

5 Summary & Conclusion

This work investigated the relationship between the number of classes and the performance of BERT on the task of artist classification from lyrics. This task is illustrative for a range of problems that in real world settings have hundreds of classes and non-normal/-zipfian distributions. For increasing numbers of classes, we compared two baselines, random and kNN with BOW, with three different BERT-settings: 1) Fine-tuning BERT with a classification head, 2) using fine-tuned BERT embeddings with kNN and 3) using pre-trained BERT embeddings with kNN.

While for lower numbers of classes, fine-tuning BERT outperforms the other methods, the situation is not as clear for higher numbers of classes. In direct contrast, the decrease in performance for BOW kNN baseline and the pre-trained BERT embeddings with kNN follows a smoother curve than for fine-tuning BERT with a classification head and using the fine-tuned BERT embeddings with kNN. We find that for higher numbers of classes using the fine-tuned embeddings outperform their classification head counterpart.

We conclude that, as expected, the performance across methods decreases drastically. While fine-tuning BERT and using the classification head for lower numbers of classes and using the fine-tuned BERT embeddings for higher numbers of classes perform better than the baselines, the decrease in performance points towards the contextual BERT embeddings not holding enough information to classify long-form documents such as lyrics. Moreover, it is conceivable that for this specific task just using lyrics might not be enough with higher numbers of classes. The more artists and songs in the data, the more overlap in the lyrics we expect across classes. For higher numbers of classes, the factor in terms of lyrics that makes it possible to discriminate between artists might not be which words the lyrics use in which structure and which lexical context, but in which musical context they are used. More generally, we argue that for such tasks there is a need for models with representations capable of capturing relationships not only of (sub-)words but phrases, sentences and paragraphs.

6 Future work

Given our observations laid out above, on the one hand the question arises how these findings map to other transformer-based language models and tasks. While we suspect tasks generally become harder with increasing numbers of classes, the situation may differ across tasks. As the task of this work illustrates, with increasing amounts of classes some tasks may require more than the direct context to discriminate between classes. Thus a possible direction for future work is to investigate how the relationship of number of classes and performance differs across tasks and for different models. This in turn could give an idea which models fare more or less well for certain types of classification tasks and might help analyze their shortcomings.

On the other hand, our findings point to the need for methods that are able to deal with noisy, sparse, unbalanced data and classes in tasks where data augmentation is not a viable option. One proposed direction is data-stratifying loss functions (Li et al., 2020). Since these, too, have mostly been used in tasks with rather confined numbers of classes, future work could investigate the usefulness and limitations of loss functions with tasks with more classes.

References

Ilias Chalkidis, Manos Fergadiotis, Sotiris Kotitsas, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. An empirical study on large-scale multi-label text classification including few and zero-shot labels. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7503–7515, Online. Association for Computational Linguistics.

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, Minneapolis, Minnesota. Association for Computational Linguistics.

Maël Fabien, Esau Villatoro-Tello, Petr Motlicek, and Shantipriya Parida. 2020. BertAA : BERT fine-tuning for authorship attribution. In *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 127–137, Indian Institute of Technology Patna, Patna, India. NLP Association of India (NLPAI).

Michael Fell and Caroline Sporleder. 2014. Lyrics-based analysis and classification of music. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 620–631, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Evelyn Fix and Joseph L Hodges. 1951. Discriminatory analysis. *Nonparametric discrimination: Small sample performance. Report A*, 193008.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020. Dice loss for data-imbalanced NLP tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 465–476, Online. Association for Computational Linguistics.

Nut Limsopatham. 2021. Effectively leveraging BERT for legal document classification. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 210–216, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Anthony Rios and Ramakanth Kavuluru. 2018. Few-shot and zero-shot multi-label learning for structured label spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3132–3142, Brussels, Belgium. Association for Computational Linguistics.

Jiaming Shen, Wenda Qiu, Yu Meng, Jingbo Shang, Xiang Ren, and Jiawei Han. 2021. TaxoClass: Hierarchical multi-label text classification using only class names. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4239–4249, Online. Association for Computational Linguistics.

Henry Tsai, Jason Riesa, Melvin Johnson, Naveen Arivazhagan, Xin Li, and Amelia Archer. 2019. Small and practical BERT models for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3632–3636, Hong Kong, China. Association for Computational Linguistics.

Binxia Xu, Siyuan Qiu, Jie Zhang, Yafang Wang, Xiaoyu Shen, and Gerard de Melo. 2020a. Data augmentation for multiclass utterance classification – a systematic study. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5494–5506, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2020b. DomBERT: Domain-oriented language model for aspect-based sentiment analysis. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1725–1731, Online. Association for Computational Linguistics.