arXiv:2109.03777v2 [cs.CL] 23 Sep 2021

Forget me not: A Gentle Reminder to Mind the Simple Multi-Layer Perceptron Baseline for Text Classification

Lukas Galke
University of Kiel / ZBW
Germany
lga@informatik.uni-kiel.de

Ansgar Scherp
University of Ulm
Germany
ansgar.scherp@uni-ulm.de

Abstract

Graph neural networks have triggered a resurgence of graph-based text classification. We show that already a simple MLP baseline achieves comparable performance on benchmark datasets, questioning the importance of synthetic graph structures. When considering an inductive scenario, i. e., when adding new documents to a corpus, a simple MLP even outperforms the recent graph-based models TextGCN and HeteGCN and is comparable with HyperGAT. We further fine-tune DistilBERT and find that it outperforms all state-of-the-art models. We suggest that future studies use at least an MLP baseline to contextualize the results. We provide recommendations for the design and training of such a baseline.

1 Introduction

Text classification is an active area of research as the amount of new methods and recent surveys show (Li et al., 2020; Zhou et al., 2020; Kowsari et al., 2019; Kadhim, 2019). In this work, we refer to text classification as the topical categorization of text. Note that also other tasks such as question answering (Rajpurkar et al., 2016) or natural language inferencing (Wang et al., 2019) can be casted as text classification on a technical level (Devlin et al., 2019). In those cases, the positional information of the sequence is more important than it is when the task is pure topical text classification.

What is interesting to observe is that among the various methods compared for text classification, the good old multi-layer pereceptron (MLP) is rarely among them. This is surprising as MLP-based models were shown to be good performers for classification tasks (Shen et al., 2018; Mai et al., 2018; Galke et al., 2017). MLPs are conceptually simple and have only few hyperparameters (number of layers, hidden dimension), and can be combined with any input representation such as a bag-of-words, optionally with TF-IDF weighting, and/or pretrained word embeddings.

It appears that the MLP model has been forgotten as baseline in the literature. However, considering strong baselines is an important means to argue about *true* scientific advancement (Shen et al., 2018; Dacrema et al., 2019).

We review the key research in text classification and identify the top performing models. We extract the scores reported for these model on established benchmark datasets. We show the lack of using an MLP as strong baseline and run our own experiments with different variants of a simple MLP. Finally, we fine-tune a pretrained language model DistilBERT (Sanh et al., 2019), a size-reduced version of BERT (Devlin et al., 2019), on the text classification datasets.

Our results show that a SimpleMLP with one hidden layer is close or even outperforms recent graph-based approaches (Yao et al., 2019; Liu et al., 2020; Ragesh et al., 2021). We argue that a one-hidden-layer MLP satisfies the universal approximation theorem (Cybenko, 1989): A single hidden layer with nonlinear activation is sufficient to approximate any compact function to an arbitrary degree of accuracy (depending on its width). A fine-tuned DistilBERT sets a new state of the art.

We conjecture that there is a forgotten merit in using MLPs for text classification as a simple and very strong model and that it should be used in future studies. In fact, other fields have reported a resurgence of MLPs. In example, an MLP baseline outperforms various other Deep Learning models for business prediction (Venugopal et al., 2021). In computer vision, Tolstikhin et al. (2021) and Melas-Kyriazi (2021) propose pure MLP models and question the necessity of self-attention in Vision Transformers (Dosovitskiy et al., 2021). Liu et al. (2021) show similar results in natural language processing, while acknowledging that a small attention module is necessary for some tasks. This shows the importance of MLP baselines and we hope with this work to contribute to their resurgence.

2 Selecting State of the Art Models

We discuss related works on text classification to identify the best performing approaches. We base the identification of the models in our comparison on the recent surveys, which cover the range from shallow to deep classification models (Li et al., 2020; Zhou et al., 2020; Kowsari et al., 2019; Kadhim, 2019). We note that the recent surveys from 2020 and 2019 include both classical and Deep Learning models, but none considered a simple 1-layer MLP. A notable exception is the inclusion of DAN (Iyyer et al., 2015), a deep MLP model with n hidden layers, in (Li et al., 2020). We complement our search by checking results and papers on paperswithcode.com. We focus our analysis on models showing strong performance on benchmark datasets (see Table 1). We identify the best performing models per different approach. Furthermore, we explicitly screen for further literature in the key NLP and AI venues. For all models, we have verified that the same train-test split is used and check whether modified versions of the datasets have been used (e. g., less classes).

Embedding-based models Classic machine learning models are extensively discussed in two surveys (Kowsari et al., 2019; Kadhim, 2019) and other comparison studies (Galke et al., 2017). Iyyer et al. (2015) have proposed deep averaging networks (DAN) combining word embeddings and deep feedward networks. Their results suggest to use pretrained embeddings such as GloVe (Pennington et al., 2014) over a randomly initialized neural bag of-words (Kalchbrenner et al., 2014) as input. In fastText (Joulin et al., 2017) a linear layer on top of pretrained embeddings is used for classification, which is often considered as baseline. Furthermore, Shen et al. (2018) explore further embedding pooling variants find that simple word embedding models (SWEM) can rival RNN- and CNN-based approaches. We consider all three models in our comparison and show that a single wide hidden layer is stronger than using pretrained embeddings or more layers. Note that those approaches that rely on a logistic regression on-top of a word embedding, e. g., fastText, share a similar architecture as an MLP with one hidden layer. However, the standard training protocol involves pretraining the word embedding on large amounts of unlabeled text. In our experiments, we show that this is not necessary and may be even harmful for (topical) text classification. Overall, we identify logistic regression and MLP as top performer of classical models to include in this paper.

Recurrent neural networks (RNN) are a natural choice for any NLP task and have been extensively investigated, too. However, it turned out to be challenging to find numbers reported in the literature that can be used as reference. The bidirectional LSTM with two-dimensional max pooling BLSTM-2DCNN (Zhou et al., 2016) has been applied on a stripped-down to 4 classes version of the 20ng dataset. Thus, the high score of 96.5 reported for 4ng cannot be compared with papers applied on the full 20ng dataset. Also TextRCNN (Lai et al., 2015), a model combining recurrence and convolution uses only the 4 major categories in the 20ng dataset. The results of TextRCNN is identical with BLSTM-2DCNN. For the MR dataset, BLSTM-2DCNN provides no information on the specific splitting of the dataset. RNN-Capsule (Wang et al., 2018) is a contemporary model for sentiment analysis, with an accuracy of 83.8 for the MR dataset.

Graph-based text classification has a long history in NLP. It appears also to be the currently most active area, perhaps due to the recent interest in graph-neural networks (GNN) (Hamilton, 2020). An early work is the term co-occurrence graph of the KeyGraph algorithm (Ohsawa et al., 1998). Co-occurence graphs have also been used for automatic keyword extraction such as in RAKE (Rose et al., 2010). Modern approaches exploit this idea in combination with GNNs. Those include TextGCN (Yao et al., 2019), TensorGCN (Liu et al., 2020), HeteGCN (Ragesh et al., 2021), and HyperGAT (Ding et al., 2020). In TextGCN, the authors set up a graph based on word-word connections given by window-based pointwise mutual information and word-document TF-IDF scores. HeteGCN split the adjacency matrix into its word-document and word-word submatrices and fuse the different layers' representations when required. TensorGCN uses multiple ways of converting text data into graph data includic a semantic graph created with an LSTM, a syntactic graph created by dependency parsing, and a sequential graph based on word co-occurrence. Finally, HyperGAT extended the idea of text-induced graphs for text classification to hypergraphs. The model uses graph attention and two kinds of hyperedges. Sequential hyperedges represent the relation between sentences and their words. Semantic hyperedges for word-word connections are derived from topic models (Blei et al., 2003).

In TextGCN's original transductive formulation, the entire graph including the test set needs to be known for training. This may be prohibitive in practical applications as each batch of new documents would require retraining the model. When these methods are adapted for inductive learning, where the test set is unseen, they achieve notably lower scores (Ragesh et al., 2021). GNNs for text classification use corpus statistics, e. g., pointwise mutual information (PMI), to connect related words in a graph (Yao et al., 2019). When these were omitted, the GNNs would collapse to bag-of-words MLPs. Thus, GNNs have access to more information than MLPs. GloVe (Pennington et al., 2014) also captures PMI corpus statistics, which is why we include an MLP on GloVe input representations in our experiments.

Hierarchical methods exploit a hierarchical taxonomy of the classes for text classification. HR-DGCNN (Peng et al., 2018) follows this idea and first converts the text to a word co-occurence graph on which hierarchically regularized convolution operations are applied. The model was applied on two popular benchmark datasets for *multi-label text classification* with a Micro-F1 score of 0.76 for RCV1 and and 0.65 for NYT. One year later, Xiao et al. (2019) proposed the hierarchical text classification model PCEM, which uses path information from the taxonomy in the learning algorithm. Experiments on RCV1 showed a Micro-F1 score of 77.83, slighthly higher than HR-DGCNN (Peng et al., 2018). A direct comparison of PCEM with HR-DGCNN was not performed or reported.

Multi-label text classification often comes naturally with hierarchical taxonomies. However, we focus our study on single-label datasets. PCEM had been applied on the *single-label* 20NG data, where it showed a Micro-F1 score of 70.73. Notably, the authors focus on a weakly-labeled classification task with only 1% of the labels and exploit the path information in the taxonomy. This makes it difficult to compare PCEM with our results (see Table 2). Generally, multi-label classification adds another level of complexity for comparing the model performance, where F1-scores instead of accuracy are reported. This raises further challenges, which we leave as future work.

Transformer-based Models Since our aim is to argue for universally while at the same time seek lightweight models, we look into Transformers that have been distilled into smaller-sized variants. There are several candidates of smaller-sized versions of BERT. We excluded TinyBert (Jiao et al., 2020) since it requires a full BERT model as teacher for fine-tuning. MobileBERT (Sun et al., 2020) would be suitable for our goals, but it relies on a special BERT-large model with inverted-bottleneck modifications. DistilBERT (Sanh et al., 2019) is a distilled version of BERT (Devlin et al., 2019) with 40% reduced parameters while retaining 97% of BERT's language understanding capabilities. We use DistilBERT because its inference times are 60% faster and it is more likely to be reusable by labs with limited resources. Furthermore, DistilBERT (Sanh et al., 2019) can be directly fine-tuned to downstream tasks while showing state-of-the-art performance among size-reduced BERT variants. Thus, we use the DistilBERT in our experiments.

3 Recommendations for Designing an MLP Baseline

We describe our SimpleMLP model and provide recommendations for using it as baseline in future work.

Tokenization Parallel to developments in large-scale language models, also tokenization has evolved in recent years. We borrow the tokenization strategy from BERT (Devlin et al., 2019) along with its uncased vocabulary. The tokenizer relies primarily on WordPiece (Wu et al., 2016) for a high coverage while maintaining a small vocabulary.

Input representation For text classification, the content of the input words is more important than the word order (Conneau et al., 2018). Therefore, we use length-normalized bag-of-words inputs. We further experiment with TF-IDF weighting, normalized to unit L2-norm. Both are viable options. In contrast to conventional wisdom (Iyyer et al., 2015), we find that pretrained embeddings, e. g., GloVe, often have a detrimental effect when compared to using one wide hidden layer instead.

Depth vs width In text classification, width seems more important than depth. We recommend to use a single hidden layer, i. e., one input-to-hidden and one hidden-to-output layer, with $1,024$ hidden units and ReLU activation. While this might be overparameterized for single-label text classification tasks with few classes, we rely on recent findings that suggest that overparameterization leads to better generalization (Neyshabur et al., 2018; Nakkiran et al., 2020).

We further motivate the choice of using wide layers by our own prior work on multi-label text classification (Galke et al., 2017), in which we have shown that MLP outperforms all tested classical baselines such as SVMs, k-Nearest Neighbors, and logistic regression. In follow-up work (Mai et al., 2018), we found that also CNN and LSTM could not substantially improve over the wide MLP.

Having a fully-connected layer on-top of a bag-of-words leads to a high number of learnable parameters. Still, we can implement this first input-to-hidden layer efficiently by using an embedding layer followed by aggregation, which avoids large matrix multiplications.

For demonstration purposes, we also experiment with more hidden layers (SimpleMLP-2), as suggested by Iyyer et al. (2015), but we do not observe any improvement when the single hidden layer is sufficiently wide.

Optimization and Regularization We seek to find an optimization strategy that does not require dataset-specific hyperparameter tuning. This comprises optimizing cross-entropy with Adam (Kingma and Ba, 2015) and default learning rate 10^{-3}, a linearly decaying learning rate schedule and training for a high amount of steps (Nakkiran et al., 2020) (we use 100 epochs) with small batch sizes (we use 16) for sufficient stochasticity. For regularization during this prolonged training, we suggest to use a high dropout ratio of 0.5. Regarding initialization, we rely on framework defaults, i. e., $\mathcal{N}(0, 1)$ for the initial embedding layer and random uniform $\mathcal{U}(-\sqrt{d_{\text{input}}}, \sqrt{d_{\text{output}}})$ for subsequent layers' weight and bias parameters.

4 Experimental Apparatus

Datasets We use the same datasets and train-test split as in TextGCN (Yao et al., 2019). Those datasets are:

- Twenty Newsgroups (20ng)[1] (bydate version) with all 20 classes,
- Movie Review (mr)[2] (Pang and Lee, 2005) with the split of (Tang et al., 2015),
- R8 and R52, which are subsets of the Reuters 21578 news dataset with 8 and 52 classes, respectively,
- and Ohsumed[3], a corpus from the MEDLINE database excluding those that belong to multiple classes.

We summarize the basic characteristics of the datasets in Table 1.

Table 1: Characteristics of text classification datasets

Dataset	N	#Train	#Test	#Classes
20ng	18,846	11,314	7,532	20
R8	7,674	5,485	2,189	8
R52	9,100	6,532	2,568	52
ohsumed	7,400	3,357	4,043	23
MR	10,662	7,108	3,554	2

Procedure We have extracted accuracy scores from the literature according to our systematic selection from Section 2. *Without any hyperparameter tuning*, we run our SimpleMLP as described in Section 3. We fine-tune DistilBERT for 10 epochs with a linearly decaying learning rate of $5 \cdot 10^{-5}$ and an effective batch size of 128, while truncating inputs to 512 tokens. We repeat all experiments 5 times and report mean and standard deviation.

5 Results

Table 2 shows the accuracy scores for the text classification datasets. All graph-based models in the transductive setting show similar accuracy scores (maximum difference is 2 points). As expected, the scores decrease in the inductive setting up to a point where they are matched or even outperformed by our SimpleMLP baseline.

The strong performance of SimpleMLP rivals all techniques reported in the literature, in particular the the recently published graph-inducing methods. MLP only falls behind HyperGAT, which relies on topic models to set up the graph. Another observation is that 1 hidden layer (but wide) is sufficient for the tasks, as the scores for MLP variants with 2 hidden layers are lower. We further observe that both pure and TF-IDF weighted input representations lead to better results than approaches that exploit pretrained word embeddings such as DAN, fastText, and SWEM. With its immense pretraining, DistilBERT yields the overall highest scores.

[1]`http://qwone.com/~jason/20Newsgroups/`

[2]`https://www.cs.cornell.edu/people/pabo/movie-review-data/`

[3]`http://disi.unitn.it/moschitti/corpora.htm`

Table 2: Accuracy on text classification datasets. SD in parentheses. References indicate source of numbers.

Transductive Setting	**20ng**	**R8**	**R52**	**ohsumed**	**MR**
TextGCN (Yao et al., 2019)	86.34	97.07	93.56	68.36	76.74
Text-induced SGC (Wu et al., 2019)	88.5 (0.1)	97.2 (0.1)	94.0 (0.2)	68.5 (0.3)	75.9 (0.3)
TensorGCN (Liu et al., 2020)	87.74	98.04	95.05	70.11	77.91
HeteGCN (Ragesh et al., 2021)	87.15 (0.15)	97.24 (0.51)	94.35 (0.25)	68.11 (0.70)	76.71 (0.33)
Inductive Setting	**20ng**	**R8**	**R52**	**ohsumed**	**MR**
Log.Reg. (Ragesh et al., 2021)	83.70	93.33	90.65	61.14	76.28
fastText (Ding et al., 2020)	79.38 (0.30)	96.13 (0.21)	92.81 (0.09)	57.70 (0.49)	75.14 (0.20)
LSTM (pretrain) (Ding et al., 2020)	75.43 (1.72)	96.09 (0.19)	90.48 (0.86)	51.10 (1.50)	77.33 (0.89)
SWEM (Ding et al., 2020)	85.16 (0.29)	95.32 (0.26)	92.94 (0.24)	63.12 (0.55)	76.65 (0.63)
TextGCN (Ragesh et al., 2021)	80.88 (0.54)	94.00 (0.40)	89.39 (0.38)	56.32 (1.36)	74.60 (0.43)
HeteGCN (Ragesh et al., 2021)	84.59 (0.14)	97.17 (0.33)	93.89 (0.45)	63.79 (0.80)	75.62 (0.26)
HyperGAT (Ding et al., 2020)	86.62 (0.16)	97.07 (0.23)	94.98 (0.27)	69.90 (0.34)	78.32 (0.27)
TF-IDF + SimpleMLP (ours)	84.20 (0.16)	97.08 (0.16)	93.67 (0.23)	66.06 (0.29)	76.32 (0.17)
SimpleMLP (ours)	83.31 (0.22)	97.27 (0.12)	93.89 (0.16)	63.95 (0.13)	76.72 (0.26)
SimpleMLP-2 (ours)	81.02 (0.23)	96.61 (1.22)	93.98 (0.23)	61.71 (0.33)	75.91 (0.51)
GloVe+SimpleMLP (ours)	76.80 (0.11)	96.44 (0.08)	93.58 (0.06)	61.36 (0.22)	75.96 (0.17)
GloVe+SimpleMLP-2 (ours)	76.33 (0.18)	96.50 (0.14)	93.19 (0.11)	61.65 (0.27)	75.72 (0.45)
DistilBERT (ours)	86.24 (0.26)	97.89 (0.15)	95.34 (0.08)	69.08 (0.60)	85.10 (0.33)

DistilBERT outperforms HyperGAT by 7 points on the MR dataset while being on-par on the others.

6 Discussion

Intuitively, we explain the strong performance of MLPs that for text classification a bag-of-words model is oftentimes sufficient to grasp the topic of a text, although word order is discarded. As such, our SimpleMLP can effectively learn the function from features to class labels without any hyperaparameter tuning or early stopping. Potentially, the scores could be even increased by using a grid search over hyperparameters. However, the goal of this study is to suggest a simple but effective baseline that works well on different datasets, even with only choosing default hyperparameters. Finally, we observe that a fine-tuned DistilBERT even sets a new state-of-the-art. But still in practice, such a model has to be handled, and a from-scratch trained model like SimpleMLP may be favored in contexts where an existing model cannot be used due to legal or other reasons. Overall, the rationale of our study is to encourage a quick use and take up of a simple (MLP) baseline, which—as discussed in this work–is almost always omitted.

We expect that similar observations would be made on other text classification datasets. We acknowledging that all our current datasets are limited to English language. It is, however, notable that methods that discard word order work well for (topical) text classification.

Despite the strong results for SimpleMLP, it is certainly not a *silver bullet*. There are also tasks where the *natural* structure of the graph data provides more information than the mere text, e. g., citations networks or connections in social graphs. In such cases the performance of graph neural networks is the state of the art (Kipf and Welling, 2017; Veličković et al., 2018) and are superior to MLPs that use only the graph's vertex features and not the graph structure (Shchur et al., 2018).

In contrast, the surplus value of *synthetic* graph structures like word co-occurrence graphs computed over a text corpus is questionable. Our results in Table 2 show that SimpleMLP challenges or outperforms all recent graph-based methods when applied to unseen documents (inductive setting), except for the expensive LDA-enriched HyperGAT.

7 Conclusion

We argue that a simple multi-layer perceptron enhanced with all of today's best practices should be considered as a strong baseline for text classification tasks. In fact, the experiments show that our SimpleMLP is oftentimes on-par or even better than recently proposed models that synthesize a graph structure from the text. A future direction of this work would deeper analyze more challenging multi-label text classification tasks. The code is available on GitHuB at `https://github.com/lgalke/text-clf-baselines`.

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single \$&!#* vector: Probing sentence embeddings for linguistic properties. In *ACL (1)*, pages 2126–2136. ACL.

George Cybenko. 1989. Approximation by superpositions of a sigmoidal function. *Math. Control. Signals Syst.*, 2(4):303–314.

Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *RecSys*, pages 101–109. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. ACL.

Kaize Ding, Jianling Wang, Jundong Li, Dingcheng Li, and Huan Liu. 2020. Be more with less: Hypergraph attention networks for inductive text classification. In *EMNLP (1)*, pages 4927–4936. ACL.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*.

Lukas Galke, Florian Mai, Alan Schelten, Dennis Brunsch, and Ansgar Scherp. 2017. Using titles vs. full-text as source for automated semantic document annotation. In *K-CAP*, pages 20:1–20:4. ACM.

William L. Hamilton. 2020. *Graph Representation Learning*.

Mohit Iyyer, Varun Manjunatha, Jordan L. Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *ACL (1)*, pages 1681–1691. ACL.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling BERT for natural language understanding. In *EMNLP (Findings)*, volume EMNLP 2020 of *Findings of ACL*, pages 4163–4174. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomás Mikolov. 2017. Bag of tricks for efficient text classification. In *EACL (2)*, pages 427–431. ACL.

Ammar Ismael Kadhim. 2019. Survey on supervised machine learning techniques for automatic text classification. *Artif. Intell. Rev.*, 52(1):273–292.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *ACL (1)*, pages 655–665. ACL.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR (Poster)*. OpenReview.net.

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

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *AAAI*, pages 2267–2273. AAAI Press.

Qian Li, Hao Peng, Jianxin Li, Congying Xia, Renyu Yang, Lichao Sun, Philip S. Yu, and Lifang He. 2020. A survey on text classification: From shallow to deep learning. *CoRR*, abs/2008.00364.

Hanxiao Liu, Zihang Dai, David R. So, and Quoc V. Le. 2021. Pay attention to mlps. *CoRR*, abs/2105.08050.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020. Tensor graph convolutional networks for text classification. In *AAAI*, pages 8409–8416. AAAI Press.

Florian Mai, Lukas Galke, and Ansgar Scherp. 2018. Using deep learning for title-based semantic subject indexing to reach competitive performance to full-text. In *JCDL*, pages 169–178. ACM.

Luke Melas-Kyriazi. 2021. Do you even need attention? A stack of feed-forward layers does surprisingly well on imagenet. *CoRR*, abs/2105.02723.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. 2020. Deep double descent: Where bigger models and more data hurt. In *ICLR*. OpenReview.net.

Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. 2018. Towards understanding the role of over-parametrization in generalization of neural networks. *CoRR*, abs/1805.12076.

Yukio Ohsawa, Nels E. Benson, and Masahiko Yachida. 1998. Keygraph: Automatic indexing by co-occurrence graph based on building construction metaphor. In *ADL*, pages 12–18. IEEE Computer Society.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. ACL.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *WWW*, pages 1063–1072. ACM.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543. ACL.

Rahul Ragesh, Sundararajan Sellamanickam, Arun Iyer, Ramakrishna Bairi, and Vijay Lingam. 2021. HeteGCN: Heterogeneous graph convolutional networks for text classification. In *WSDM*, pages 860–868. ACM.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392. ACL.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. In Michael W. Berry and Jacob Kogan, editors, *Text Mining. Applications and Theory*, pages 1–20. John Wiley and Sons, Ltd.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Pitfalls of graph neural network evaluation. *CoRR*, abs/1811.05868.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. In *ACL (1)*, pages 440–450. ACL.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic BERT for resource-limited devices. In *ACL*, pages 2158–2170. ACL.

Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. PTE: predictive text embedding through large-scale heterogeneous text networks. In *KDD*, pages 1165–1174. ACM.

Ilya O. Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. 2021. Mlp-mixer: An all-mlp architecture for vision. *CoRR*, abs/2105.01601.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. *International Conference on Learning Representations*.

Ishwar Venugopal, Jessica Töllich, Michael Fairbank, and Ansgar Scherp. 2021. A comparison of deep-learning methods for analysing and predicting business processes. In *IJCNN*. IEEE.

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

Yequan Wang, Aixin Sun, Jialong Han, Ying Liu, and Xiaoyan Zhu. 2018. Sentiment analysis by capsules. In *WWW*, pages 1165–1174. ACM.

Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. 2019. Simplifying graph convolutional networks. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 6861–6871. PMLR.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Huiru Xiao, Xin Liu, and Yangqiu Song. 2019. Efficient path prediction for semi-supervised and weakly supervised hierarchical text classification. In *WWW*, pages 3370–3376. ACM.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *AAAI*, pages 7370–7377. AAAI Press.

Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. 2016. Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. In *COLING*, pages 3485–3495. ACL.

Xujuan Zhou, Raj Gururajan, Yuefeng Li, Revathi Venkataraman, Xiaohui Tao, Ghazal Bargshady, Prabal Datta Barua, and Srinivas Kondalsamy-Chennakesavan. 2020. A survey on text classification and its applications. *Web Intell.*, 18(3):205–216.