AEDA: An Easier Data Augmentation Technique for Text Classification

Akbar Karimi **Leonardo Rossi** **Andrea Prati**
IMP Lab, University of Parma, Italy
{akbar.karimi, leonardo.rossi, andrea.prati}@unipr.it

Abstract

This paper proposes **AEDA** (**A**n **E**asier **D**ata **A**ugmentation) technique to help improve the performance on text classification tasks. AEDA includes only random insertion of punctuation marks into the original text. This is an easier technique to implement for data augmentation than EDA method (Wei and Zou, 2019) with which we compare our results. In addition, it keeps the order of the words while changing their positions in the sentence leading to a better generalized performance. Furthermore, the deletion operation in EDA can cause loss of information which, in turn, misleads the network, whereas AEDA preserves all the input information. Following the baseline, we perform experiments on five different datasets for text classification. We show that using the AEDA-augmented data for training, the models show superior performance compared to using the EDA-augmented data in all five datasets. The source code is available for further study and reproduction of the results[1].

1 Introduction

Text classification is a major area of study in natural language processing (NLP) with numerous applications such as sentiment analysis, toxicity detection, and question answering, to name but a few. In order to build text classifiers that perform well, the training data need to be large enough so that the model can generalize to the unseen data. However, for many machine learning (ML) applications and domains, there do not exist sufficient labeled data for training. In this situation, data augmentation (DA) can provide a solution and help improve the performance of ML systems (Ragni et al., 2014; Fadaee et al., 2017; Ding et al., 2020). DA can be carried out in many different ways such as by modifying elements of the input sequence, namely word substitution, deletion, and insertion (Wei and Zou, 2019; Zhang et al., 2015), and back-translation (Sennrich et al., 2016). It can also be performed by noise injection in the input sequence (Xie et al., 2019) or in the embedding space utilizing a deep language model (Jiao et al., 2020; Karimi et al., 2021; Garg and Ramakrishnan, 2020).

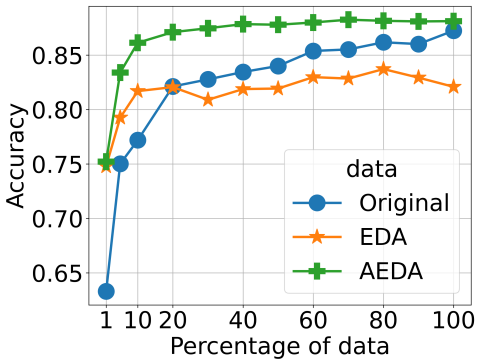


Figure 1: Average performance of the generated data using our proposed augmentation method (AEDA) compared with that of the original and EDA-generated data on five text classification tasks. Using both EDA and AEDA, we added 9 augmented sentences to the original training set to train the models. For each task, we ran the models with 5 different seed numbers and took the average score.

Using a deep language model to do DA can be complicated, while word replacement techniques with the help of a word thesaurus, even though a simple method, risks information loss due to the operations such as deletion and substitution. These operations can even result in changing the label of the input sequence (Kumar et al., 2020), thus misleading the network.

To address these problems, we propose an extremely simple yet effective approach called AEDA (An Easier Data Augmentation) which includes only the insertion of various punctuation marks into the input sequence. AEDA preserves all the input information and does not mislead the network

[1] https://github.com/akkarimi/aeda_nlp

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 2748–2754
November 7–11, 2021.

since it keeps the word order intact while changing their positions in that the words are shifted to the right. Our extensive experiments show that AEDA helps the models avoid overfitting (Figure 1).

2 Related Work

Although the textual content is always increasing, data augmentation is still a highly active area of research since for machine learning applications, especially the new ones, the initial annotated data are usually small. As a result, researchers are constantly coming up with innovative ideas to create new data from the available content.

Some have experimented at the input sequence level performing operations on words. For example, to improve machine translation quality, Fadaee et al. (2017) utilize substitution of common words with rare ones, thus providing more context for the rare words, while Sennrich et al. (2016) use back-translation where automatically translated data along with the original human-translated data are employed to train a neural machine translation system. Wang and Yang (2015) replaces words with their synonyms for classifying tweets. Similarly, Andreas (2020) replace sentence fragments from common categories with each other in order to produce new sentences.

Others have opted for using pre-trained language models such as BERT (Devlin et al., 2019). Kobayashi (2018) utilizes contextual augmentation, replacing the words with the prediction of a bidirectional language model at a desired position in the sentence. Hu et al. (2019) and Liu et al. (2020) utilize reinforcement learning with a conditional language model which is carried out by attaching the correct label to the input sequence when training (Wu et al., 2019). Working with Transformer model (Vaswani et al., 2017), Sun et al. (2020) propose Mix-Transformer where two input sentences and their corresponding labels are linearly interpolated to create new samples.

Xie et al. (2019) make use of data noising which can be considered similar to our work with the difference that they replace words choosing from the unigram frequency distribution or insert the underscore character as a placeholder, whereas we insert punctuation characters which usually occur in sentences. The related works mostly use some auxiliary data or a complicated language model to produce augmented data. Conversely, our method is extremely simple to implement and does not need any extra data. In addition, it shows superior performance to EDA in both simple models such as RNNs and CNNs and deep models such as BERT.

3 AEDA Augmentation

In order to insert the punctuation marks, we randomly choose a number between 1 and one-third of the length of the sequence which indicates how many insertions will be carried out. The reason is that we want to ensure there is at least on inserted mark and at the same time we do not want to insert too many punctuation marks as too much noise might have a negative effect on the model, although this effect can be investigated in future work. Then, positions in the sequence are also specified in random as many as the selected number in the previous step. In the end, for each chosen position, a punctuation mark is picked randomly from the six punctuation marks in {".", ";", "?", ":", "!", ","}. Table 1 shows three augmentation samples by the AEDA technique.

Original	a sad , superior human comedy played out on the back roads of life .
Aug 1	a sad , superior human comedy played out on the back roads ; of life ; .
Aug 2	a , sad . , superior human ; comedy . played . out on the back roads of life .
Aug 3	: a sad ; , superior ! human : comedy , played out ? on the back roads of life .

Table 1: Examples of the augmented data using the AEDA technique.

4 Experimental Setup

Since we compare our proposed method with Wei and Zou (2019), we used the same codebase as theirs with no changes in the implementation of the models. We executed the code using a GeForce RTX 2070 GPU with 8 GB of memory.

4.1 Datasets

We experiment with the same five datasets as our baseline. They include **SST2** (Socher et al., 2013) Standford Sentiment Treebank, **CR** (Hu and Liu, 2004; Ding et al., 2008; Liu et al., 2015) Customer Reviews dataset, **SUBJ** (Pang and Lee, 2004) Subjectivity/Objectivity dataset, **TREC** (Li and

Roth, 2002) Question Classification dataset, and **PC** (Ganapathibhotla and Liu, 2008) Pros and Cons dataset. Table 2 shows the statistics of the utilized datasets.

Dataset	N_{class}	L_{avg}	N_{train}	N_{test}	|V|
SST-2	2	19	7791	1821	15771
CR	2	19	4067	451	9048
SUBJ	2	25	9000	1000	22715
TREC	6	10	5452	500	9448
PC	2	7	40000	5806	26090

Table 2: Statistics of the utilized datasets. N_{class}: Number of classes, L_{avg}: Sentence average length, N_{train}: Number of training samples, N_{test}: Number of test samples, |V|: Number of unique words.

The train and test sets utilized for the experiments for these datasets were not made available by the baseline. Therefore, after collecting them, we shuffled and divided them into train and test sets with almost the same size as the ones reported by the baseline. For the CR dataset, we combined all the reviews from the three cited sources. The annotations included multiple target sentiments for each sentence. Therefore, to convert them into binary classes, we considered a sentence positive if there was no negative sentiment and negative if there was no positive sentiment. The datasets are available along the source code.

4.2 Models

To be consistent as well as for a fair comparison of the effects of EDA- and AEDA-augmented data, we used the same Recurrent Neural Network (RNN) (Liu et al., 2016) and Convolutional Neural Network (CNN) (Kim, 2014) as implemented in the baseline. For the initialization of the models, GloVe word vectors (Pennington et al., 2014) were utilized.

5 Results

[h] To evaluate the quality of augmented sentences, we performed experiments using the data augmented by both EDA and AEDA as well as the original data. For the results reported in Table 3, we added 16 augmentations and for the ones in Figure 2, 9 augmentations to be consistent with the baseline. All experiments were repeated with 5 different seed numbers and the average scores are reported.

	Training set size			
Model	500	2,000	5,000	full set
RNN	73.5	82.6	85.9	87.9
+EDA	76.1	81.3	85.2	86.5
+AEDA	77.8	83.9	87.2	88.6
CNN	76.5	83.8	87.0	87.9
+EDA	77.5	82.2	84.5	86.1
+AEDA	78.5	84.4	86.5	88.1
Average	75.0	83.2	86.5	87.9
+EDA	76.8	81.8	84.9	86.3
+AEDA	**78.2**	**84.2**	**86.9**	**88.4**

Table 3: Comparing average performance of EDA and AEDA across all datasets on different training set sizes. For each training sample, 16 augmented sentences were added. Scores are the average of 5 runs. Numbers are in percentages.

5.1 AEDA Outperforms EDA

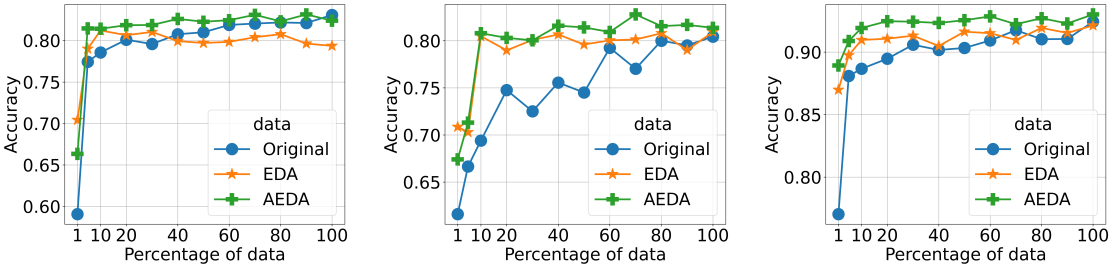
The results of the experiments with 500, 2000, 5000 and full dataset sizes for training are reported in Table 3. We can see that in some small datasets, EDA improves the results while for bigger ones it has a negative effect on the performance of the models. Conversely, AEDA gives a performance boost on all datasets, showing greater boosts for smaller ones. For instance, with 500 sentences, the average absolute improvement is 3.2% while for full dataset it is 0.5%. The reason why EDA does not perform well can be attributed to the operations such as deletion and substitution which insert more misleading information to the network as the number of augmentations grows. In contrast, AEDA keeps the original information in all augmentations.

5.2 Trend on Training Set Sizes

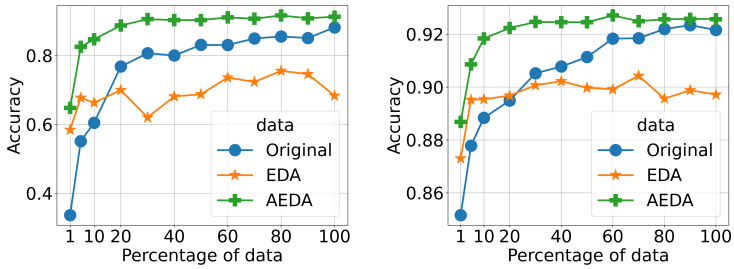
Figure 2 shows how both models perform on different fractions of the training set. These fractions include {1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100} percent. We can see that AEDA outperforms EDA in all tasks as well as showing improvements over the original data. One observation to point out is that also EDA works well on small datasets which can be because of lower number of augmentations compared to the ones reported in Table 3.

6 Ablation Study

In this section, we investigate how much gain there is for different number of augmentations, the effect of random initialization, and whether AEDA can improve deep models.



(a) SST-2 (b) CR (c) SUBJ



(d) TREC (e) PC

Figure 2: Performance of the RNN model trained on various proportions of the original, EDA-generated, and AEDA-generated training data for five text classification tasks. All the scores are the average of 5 runs.

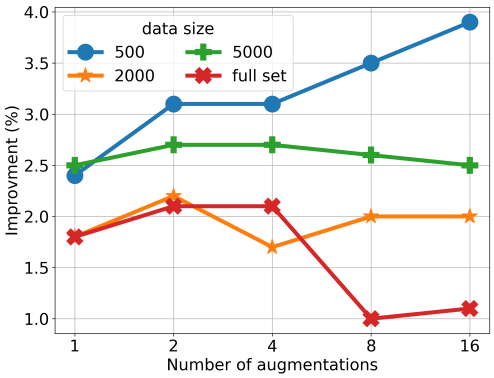
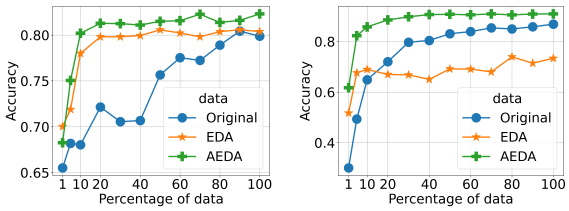


Figure 3: Impact of number of augmentations on the performance of the RNN model trained on various training sizes. Scores are the average of 5 runs over the five datasets. The y axis shows the percentage of improvement.



(a) CR (b) TREC

Figure 4: Average performance of EDA and AEDA over 21 different seed numbers. The results are in line with the experiments run over 5 seeds.

6.1 Number of Augmentations

Figure 3 presents the impact of adding various numbers of augmentations to the training set. We can see that only one augmentation can improve the performance by an absolute amount of 1.5% to 2.5% for all dataset sizes. However, as the augmentations increase, the smallest dataset greatly benefits from that by an improvement of almost 4% while the full dataset only gains 1%. The middle-sized ones have a gain in between (2% to 2.5%).

6.2 Effect of Random Initialization

When conducting the experiments, we noticed that different seed numbers produce different results. As a result, we ran the experiments for 5 times. However, in each run with the same seed number, the results can be slightly different due to the local and global generators in TensorFlow. Therefore, to ensure that 5 runs show the correct trend, we chose two of the datasets (CR and TREC) and ran the models for 21 different seeds (zero to 20). From Figure 4, we see that the trend is similar to Figure 2, which shows the average results of 5 seeds.

6.3 Using AEDA with Deep Models

The performance of AEDA on a deep model such as BERT is mixed. Table 4 shows the results of our experiments with the BERT model. We trained the model used in (Kumar et al., 2020) for 3 epochs with its default settings and observed that adding one augmentation for each training sample increased the performance by 0.15% for SST2 and 0.76% for TREC while making it deteriorate slightly for the others. However, in all cases, except for the CR dataset, it still outperforms the EDA method. The reason why AEDA does not always help a deep model can be the fact that pre-trained models have already seen a considerable amount of data with possibly similar noises to AEDA. Nevertheless, it is worth noting that, as we saw for RNN and CNN models, adding more augmentations might be more advantageous especially for small fractions of the datasets. This can be explored in the future work.

Model	SST2	CR	SUBJ	TREC	PC
BERT	91.85	**90.55**	**97.04**	96.48	**96.40**
+EDA	91.85	90.55	96.24	96.84	96.08
+AEDA	**92.00**	90.42	96.86	**97.24**	96.13

Table 4: Comparing the impact of EDA and AEDA on the BERT model. The model was trained on the combination of the original data and one augmentation for each training sample. The scores are the average of 5 runs.

7 Discussion

Comparing the results that we have gained in our experiments with the ones reported in Wei and Zou (2019), we can see some discrepancy, especially in the impact of EDA on improving the performance of the models. We speculate that the difference can be caused by the inconsistency in the training and test sets. Although we obtained the datasets from the same references they have specified, some of them are not divided into train and test datasets ready to be used. As mentioned in Section 4.1, we randomly divided them into train and test sets. In addition, some of them have different sizes which can produce different results.

With that said, to conduct a fair evaluation, we kept the same setting for all comparisons in terms of the utilized library and source code, train and test sets, number of augmentations, number of runs, batch size, and learning rate.

8 Conclusion and Future Work

We proposed an easy data augmentation technique for text classification tasks. Extensive experiments on five different datasets showed that this extremely simple method which uses punctuation marks outperforms the EDA technique which includes random deletion, insertion, and substitution of words, on all the utilized datasets. The future work will focus on exploiting the proposed method regarding which punctuation marks can have more impact, which ones to add or discard, and how many of them can be used to achieve a better performance. In addition, the question whether the punctuation marks should be inserted randomly or some positions are more effective will be investigated.

References

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566.

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.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. Daga: Data augmentation with a generation approach for low-resource tagging tasks. *arXiv preprint arXiv:2011.01549*.

Xiaowen Ding, Bing Liu, and Philip S Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 international conference on web search and data mining*, pages 231–240.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573.

Murthy Ganapathibhotla and Bing Liu. 2008. Mining opinions in comparative sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 241–248.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Zhiting Hu, Bowen Tan, Russ R Salakhutdinov, Tom M Mitchell, and Eric P Xing. 2019. Learning data manipulation for augmentation and weighting. *Advances in Neural Information Processing Systems*, 32:15764–15775.

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 *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4163–4174.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. Adversarial training for aspect-based sentiment analysis with bert. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 8797–8803. IEEE.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2873–2879.

Qian Liu, Zhiqiang Gao, Bing Liu, and Yuanlin Zhang. 2015. Automated rule selection for aspect extraction in opinion mining. In *Twenty-Fourth international joint conference on artificial intelligence*.

Ruibo Liu, Guangxuan Xu, Chenyan Jia, Weicheng Ma, Lili Wang, and Soroush Vosoughi. 2020. Data boost: Text data augmentation through reinforcement learning guided conditional generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9031–9041.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Anton Ragni, Kate M Knill, Shakti P Rath, and Mark JF Gales. 2014. Data augmentation for low resource languages. In *INTERSPEECH 2014: 15th Annual Conference of the International Speech Communication Association*, pages 810–814. International Speech Communication Association (ISCA).

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.

Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465.

Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, S Yu Philip, and Lifang He. 2020. Mixup-transformer: Dynamic data augmentation for nlp tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3436–3440.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. 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 6383–6389.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual

augmentation. In *International Conference on Computational Science*, pages 84–95. Springer.

Ziang Xie, Sida I Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y Ng. 2019. Data noising as smoothing in neural network language models. In *5th International Conference on Learning Representations, ICLR 2017*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*, pages 649–657.