Adversarial Training for Aspect-Based Sentiment Analysis with BERT

Akbar Karimi [1] **Leonardo Rossi** [1] **Andrea Prati** [1] **Katharina Full** [2]
[1] University of Parma, Italy
[2] Adidas AG, Germany
{akbar.karimi, leonardo.rossi, andrea.prati}@unipr.it, katharina.full@adidas.com

Abstract

Aspect-Based Sentiment Analysis (ABSA) deals with the extraction of sentiments and their targets. Collecting labeled data for this task in order to help neural networks generalize better can be laborious and time-consuming. As an alternative, similar data to the real-world examples can be produced artificially through an adversarial process which is carried out in the embedding space. Although these examples are not real sentences, they have been shown to act as a regularization method which can make neural networks more robust. In this work, we apply adversarial training, which was put forward by Goodfellow et al. (2014), to the post-trained BERT (BERT-PT) language model proposed by Xu et al. (2019) on the two major tasks of Aspect Extraction and Aspect Sentiment Classification in sentiment analysis. After improving the results of post-trained BERT by an ablation study, we propose a novel architecture called BERT Adversarial Training (BAT) to utilize adversarial training in ABSA. The proposed model outperforms post-trained BERT in both tasks. To the best of our knowledge, this is the first study on the application of adversarial training in ABSA.

1. Introduction

Understanding what people are talking about and how they feel about it is valuable especially for industries which need to know the customers' opinions on their products. Aspect-Based Sentiment Analysis (ABSA) is a branch of sentiment analysis which deals with extracting the opinion targets (aspects) as well as the sentiment expressed towards them. For instance, in the sentence "The spaghetti was out of this world.", a positive sentiment is mentioned towards the target which is "spaghetti". Performing these tasks requires a deep understanding of the language. Traditional machine learning methods such as SVM (Kiritchenko et al., 2014), Naive Bayes (Gamallo and Garcia, 2014), Decision Trees (Wakade et al., 2012), Maximum Entropy (Nigam et al., 1999) have long been practiced to acquire such knowledge. However, in recent years due to the abundance of available data and computational power, deep learning methods such as CNNs (LeCun et al., 1995; Kim, 2014; Zhang et al., 2015), RNNs (Liu et al., 2015; Wang et al., 2016; Ma et al., 2018), and the Transformer (Vaswani et al., 2017) have outperformed the traditional machine learning techniques in various tasks of sentiment analysis. Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) is a deep and powerful language model which uses the encoder of the Transformer in a self-supervised manner to learn the language model. It has been shown to result in state-of-the-art performances on the GLUE benchmark (Wang et al., 2018) including text classification. Xu et al. (2019) show that adding domain-specific information to this model can enhance its performance in ABSA. Using their post-trained BERT (BERT-PT), we add adversarial examples to further improve BERT's performance on Aspect Extraction (AE) and Aspect Sentiment Classification (ASC) which are two major tasks in ABSA. A brief overview of these two sub-tasks is given in Section 3.

Adversarial examples are a way of fooling a neural network to behave incorrectly (Szegedy et al., 2013). They are created by applying small perturbations to the original inputs. In the case of images, the perturbations can be invisible to human eye, but can cause neural networks to output a completely different response from the true one. Since neural nets make mistakes on these examples, introducing them to the network during the training can improve their performance. This is called "adversarial training" which acts as a regularizer to help the network generalize better (Goodfellow et al., 2014). Due to the discrete nature of text, it is not feasible to produce perturbed examples from the original inputs. As a workaround, Miyato et al. (2016) apply this technique to the word embedding space for text classification. Inspired by them and building on the work of Xu et al. (2019), we experiment with adversarial training for ABSA.

Our contributions are twofold. First, by carrying out an ablation study on the number of training epochs and the values for dropout in the classification layer, we show that there are values that outperform the specified ones for BERT-PT. Second, we introduce the application of adversarial training in ABSA by proposing a novel architecture which combines adversarial training with the BERT language model for AE and ASC tasks. Our experiments show that the proposed model outperforms the best performance of BERT-PT in both tasks.

2. Related Work

Since the early works on ABSA (Hu and Liu, 2004; Titov and McDonald, 2008; Thet et al., 2010), several methods have been put forward to address the problem. In this section, we review some of the works which have utilized deep learning techniques.

Poria et al. (2016) design a seven-layer CNN architecture and make use of both part of speech tagging and word embeddings as features. Xu et al. (2018) use convolutional neural networks and domain-specific data for AE and ASC. They show that adding the word embeddings produced from the domain-specific data to the general purpose embeddings semantically enriches them regarding the task at hand. In a recent work (Xu et al., 2019), the authors also show that using in-domain data can enhance the performance of the state-of-the-art language model (BERT). Similarly, Rietzler et al. (2019) also fine-tune BERT on domain-specific data for ASC. They perform a two-stage process, first of which is self-supervised in-domain fine-tuning, followed by supervised task-specific fine-tuning. Working on the same task, Zhaoa et al. (2019) apply graph convolutional networks taking into consideration the assumption that in sentences with multiple aspects, the sentiment about one aspect can help determine the sentiment of another aspect.

Since its introduction by Bahdanau et al. (2014), attention mechanism has become widely popular in many natural language processing tasks including sentiment analysis. Li et al. (2019) design a network to transfer aspect knowledge learned from a coarse-grained network which performs aspect category sentiment classification to a fine-grained one performing aspect term sentiment classification. This is carried out using an attention mechanism (Coarse2Fine) which contains an autoencoder that emphasizes the aspect term by learning its representation from the category embedding. Similar to the Transformer, which does away with RNNs and CNNs and use only attention for translation, Song et al. (2019) design an attention model for ASC with the difference that they use lighter (weight-wise) multi-head attentions for context and target word modeling. Using bidirectional LSTMs (Hochreiter and Schmidhuber, 1997), Li et al. (2018) propose a model that takes into account the history of aspects with an attention block called Truncated History Attention (THA). To capture the opinion summary, they also introduce Selective Transformation Network (STN) which highlights more important information with respect to a given aspect. He et al. (2017) approach the aspect extraction in an unsupervised way. Functioning the same way as an autoencoder, their model has been designed to reconstruct sentence embeddings in which aspect-related words are given higher weights through attention mechanism.

While adversarial training has been utilized for sentence classification (Miyato et al., 2016), its effects have not been studied in ABSA. Therefore, in this work, we study the impact of applying adversarial training to the powerful BERT language model.

3. Aspect-Based Sentiment Analysis Tasks

In this section, we give a brief description of two major tasks in ABSA which are called Aspect Extraction (AE) and Aspect Sentiment Classification (ASC). These tasks were sub-tasks of task 4 in SemEval 2014 contest (Pontiki et al., 2014), and since then they have been the focus of attention in many studies.

Aspect Extraction. Given a collection of review sentences, the goal is to extract all the terms, such as “waiter”, “food”, and “price” in the case of restaurants, which point to aspects of a larger entity (Pontiki et al., 2014). In order to perform this task, it is usually modeled as a sequence labeling task, where each word of the input is labeled as one of the three letters in {B, I, O}. Label ‘B’ stands for “Beginning” of the aspect terms, ‘I’ for “Inside” (aspect terms’ continuation), and ‘O’ for “Outside” or non-aspect terms. The reason for “Inside” label is that sometimes aspects can contain two or more words and the system has to return all of them as the aspect. In order for a sequence (s) of n words to be fed into the BERT architecture, they are represented as

$$[CLS], w_1, w_2, ..., w_n, [SEP]$$

where the $[CLS]$ token is an indicator of the beginning of the sequence as well as its sentiment when performing sentiment classification. The $[SEP]$ token is a token to separate a sequence from the subsequent one. Finally, w_i are the words of the sequence. After they go through the BERT model, for each item of the sequence, a vector representation of the size 768, size of BERT’s hidden layers, is computed. Then, we apply a fully connected layer to classify each word vector as one of the three labels.

Aspect Sentiment Classification. Given the aspects with the review sentence, the aim in ASC is to classify the sentiment towards each aspect as Positive, Negative, Neutral. For this task, the input format for the BERT model is the same as in AE. After the input goes through the network,

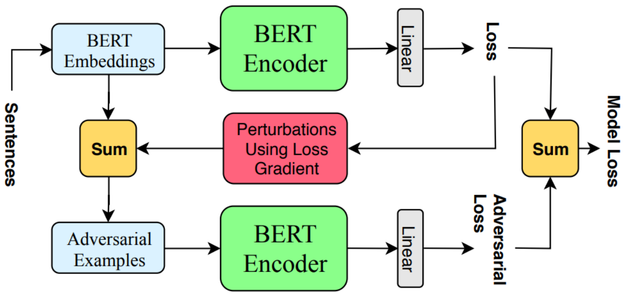


Figure 1. The proposed architecture: BERT Adversarial Training (BAT)

in the last layer the sentiment is represented by the $[CLS]$ token. Then, a fully connected layer is applied to this token representation in order to extract the sentiment.

4. Model

Our model is depicted in Figure 1. As can be seen, we create adversarial examples from BERT embeddings using the gradient of the loss. Then, we feed the perturbed examples to the BERT encoder to calculate the adversarial loss. In the end, the backpropagation algorithm is applied to the sum of both losses.

BERT Word Embedding Layer. The calculation of input embeddings in BERT is carried out using three different embeddings. As shown in Figure 2, it is computed by summing over token, segment, and position embeddings. Token embedding is the vector representation of each token in the vocabulary which is achieved using WordPiece embeddings (Wu et al., 2016). Position embeddings are used to preserve the information about the position of the words in the sentence. Segment embeddings are used in order to distinguish between sentences if there is more than one (e.g. for question answering task there are two). Words belonging to one sentence are labeled the same.

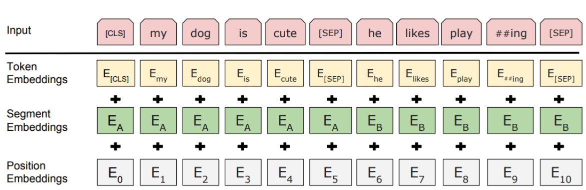


Figure 2. BERT word embedding layer (Devlin et al., 2018)

BERT Encoder. BERT encoder is constructed by making use of Transformer blocks from the Transformer model. For $\mathbf{BERT_{BASE}}$, these blocks are used in 12 layers, each of which consists of 12 multi-head attention blocks. In order to make the model aware of both previous and future contexts, BERT uses the Masked Language Model (MLM) where 15% of the input sentence is masked for prediction.

Fully Connected Layer and Loss Function. The job of the fully connected layer in the architecture is to classify the output embeddings of BERT encoder into sentiment classes. Therefore, its size is 768×3 where the first element is the hidden layers' size of BERT encoder and the second element is the number of classes. For the loss function, we use cross entropy loss implemented in Pytorch.

Adversarial Examples. Adversarial examples are created to attack a neural network to make erroneous predictions. There are two main types of adversarial attacks which are called white-box and black-box. White-box attacks (Ebrahimi et al., 2017) have access to the model parameters, while black-box attacks (Ilyas et al., 2018) work only on the input and output. In this work, we utilize a white-box method working on the embedding level. In order to create adversarial examples, we utilize the formula used by Miyato et al. (2016), where the perturbations are created using gradient of the loss function. Assuming $p(y|x;\theta)$ is the probability of label y given the input x and the model parameters θ, in order to find the adversarial examples the following minimization problem should be solved:

$$r_{adv} = \arg \min_{r, ||r|| \leq \epsilon} \log \, p(y|x + r; \hat{\theta}) \tag{1}$$

where r denotes the perturbations on the input and $\hat{\theta}$ is a constant copy of θ in order not to allow the gradients to propagate in the process of constructing the artificial examples. Solving the above minimization problem means that we are searching for the worst perturbations while trying to minimize the loss of the model. An approximate solution for Equation 1 is found by linearizing $\log p(y|x;\theta)$ around x (Goodfellow et al., 2014). Therefore, the following perturbations are added to the input embeddings to create new adversarial sentences in the embedding space.

$$r_{adv} = -\epsilon \frac{g}{||g||_2} \tag{2}$$

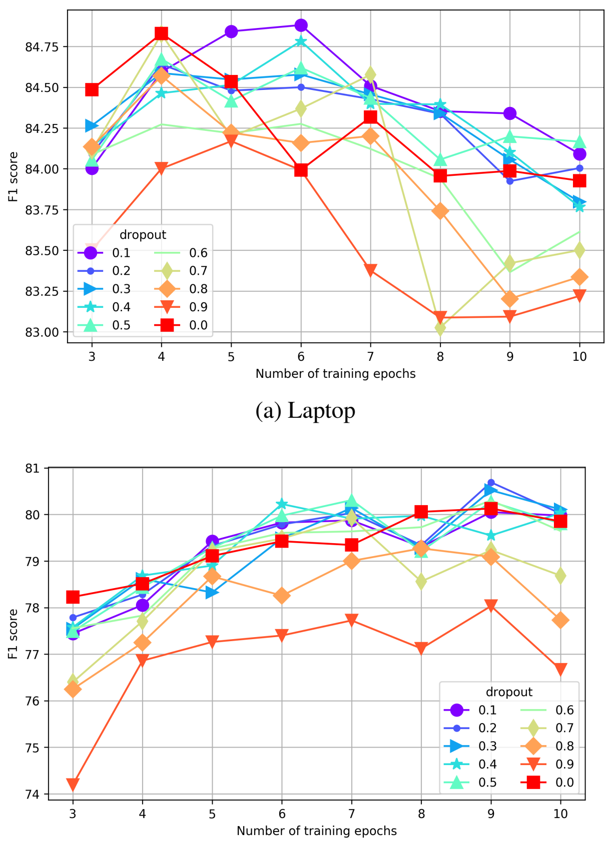
where

$$g = \nabla_x \, \log \, p(y|x; \hat{\theta}) \tag{3}$$

and ϵ is the size of the perturbations. In order to find values which outperform the original results, we carried out an ablation study on five values for epsilon whose results are presented in Figure 4 and discussed in Section 6. After the adversarial examples go through the network, their loss is calculated as follows:

$$-\log p(y|x + r_{adv}; \theta)$$

Then, this loss is added to the loss of the real examples in order to compute the model's loss.



(a) Laptop

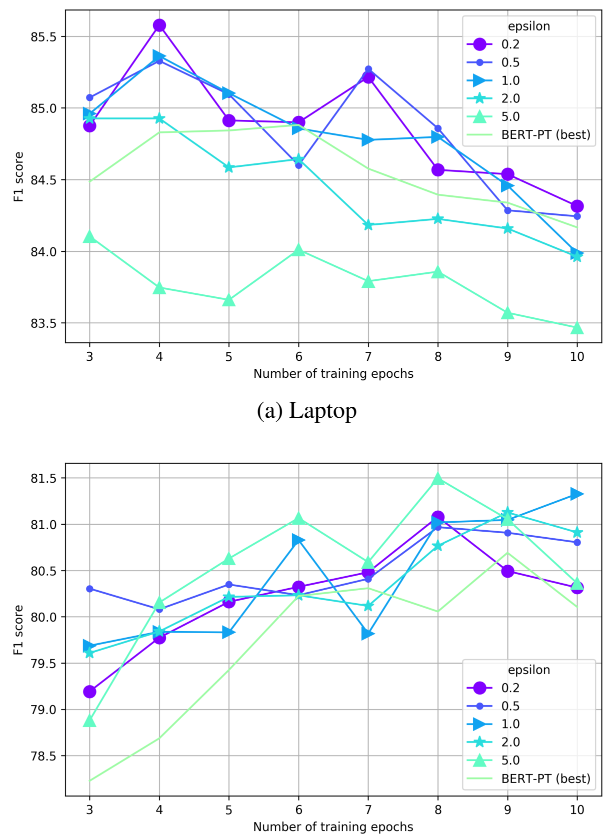
(b) Rest16

Figure 3. Ablation results on the impact of training epochs and dropout value in post-trained BERT for AE task.

5. Experimental Setup

Datasets. In order for the results to be consistent with previous works, we experimented with the benchmark datasets from SemEval 2014 task 4 (Pontiki et al., 2014) and SemEval 2016 task 5 (Pontiki et al., 2016) competitions. The laptop dataset is taken from SemEval 2014 and is used for both AE and ASC tasks. However, the restaurant dataset for AE is a SemEval 2014 dataset while for ASC is a SemEval 2016 dataset. The reason for the difference is to be consistent with the previous works. A summary of these datasets can be seen in Tables 1 and 2.

Implementation details. We performed all our experiments on a GPU (GeForce RTX 2070) with 8 GB of memory. Except for the code specific to our model, we adapted the codebase utilized by BERT-PT. To carry out the ablation study of BERT-PT model, batches of 32 were specified. However, to perform the experiments for our proposed model, we reduced the batch size to 16 in order for the GPU to be able to store our model. For optimization, the Adam optimizer with a learning rate of $3e-5$ was used. From SemEval's training data, 150 examples were chosen



(a) Laptop

(b) Rest16

Figure 4. Comparing best results of BERT-PT and BAT with different sizes of perturbations (ϵ) for AE task.

Table 1. Laptop and restaurant datasets for AE. S: Sentences; A: Aspects; Rest16: Restaurant dataset from SemEval 2016.

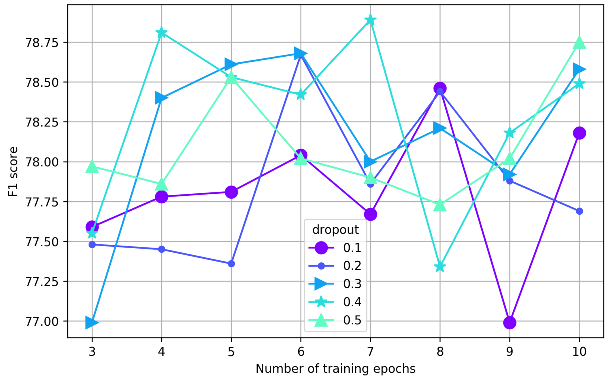
	Train		Test	
Dataset	S	A	S	A
Laptop	3045	2358	800	654
Rest16	2000	1743	676	622

Table 2. Laptop and restaurant datasets for ASC. Pos, Neg, Neu: Number of positive, negative, and neutral sentiments, respectively; Rest14: Restaurant dataset from SemEval 2014

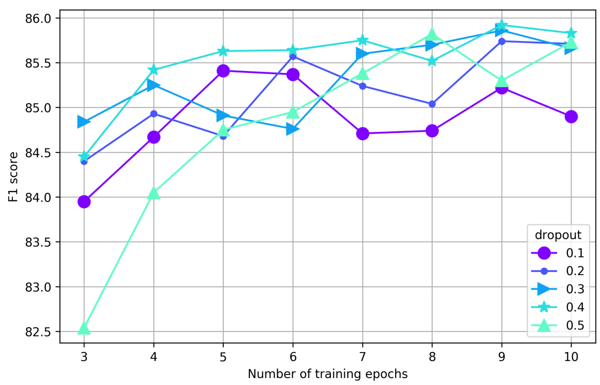
	Train			Test		
Dataset	Pos	Neg	Neu	Pos	Neg	Neu
Laptop	987	866	460	341	128	169
Rest14	2164	805	633	728	196	196

for the validation and the remaining was used for training the model.

Implementing the creation of adversarial examples for ASC task was slightly different from doing it for AE task. During our experiments, we realized that modifying all the elements

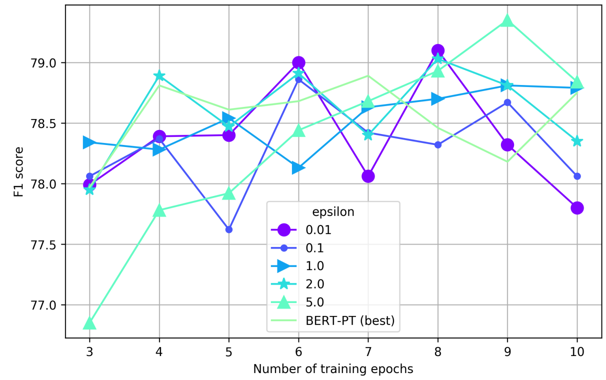


(a) Laptop

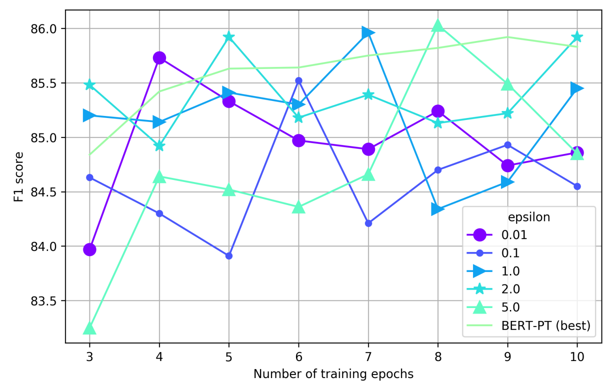


(b) Rest14

Figure 5. Ablation results on the impact of training epochs and dropout value in post-trained BERT for ASC task.



(a) Laptop



(b) Rest14

Figure 6. Comparing best results of BERT-PT and BAT with different sizes of perturbations (ϵ) for ASC task.

of input vectors does not improve the results. Therefore, we decided not to modify the vector for the $[CLS]$ token. Since the $[CLS]$ token is responsible for the class label in the output, it seems reasonable not to change it in the first place and only perform the modification on the word vectors of the input sentence. In other words, regarding the fact that the $[CLS]$ token is the class label, to create an adversarial example, we should only change the words of the sentence, not the ground-truth label.

Evaluation. To evaluate the performance of the model, we utilized the official script of the SemEval contest for AE. These results are reported as F1 scores. For ASC, to be consistent with BERT-PT, we utilized their script whose results are reported in Accuracy and Macro-F1 (MF1) measures. Macro-F1 is the average of F1 score for each class and it is used to deal with the issue of unbalanced classes.

6. Ablation Study and Results Analysis

To perform the ablation study, first we initialize our model with post-trained BERT which has been trained on uncased version of $\mathbf{BERT_{BASE}}$. We attempt to discover what number of training epochs and which dropout probability yield the best performance for BERT-PT. Since one and two training epochs result in very low scores, results of 3 to 10 training epochs have been depicted for all experiments. For AE, we experiment with 10 different dropout values in the fully connected (linear) layer. The results can be seen in Figure 3 for laptop and restaurant datasets. To be consistent with the previous work and because of the results having high variance, each point in the figure (F1 score) is the average of 9 runs. In the end, for each number of training epochs, a dropout value, which outperforms the other values, is found. In our experiments, we noticed that the validation loss increases after 2 epochs as has been mentioned in the original paper. However, the test results do not follow the same pattern. Looking at the figures, it can be seen that as the number of training epochs increases, better results are produced in the restaurant domain while in the laptop domain the scores go down. This can be attributed to the selection of validation sets as for both domains the last 150 examples of the SemEval training set were selected. Therefore, it can be said that the examples in the validation and test sets for laptop have more similar patterns than those

Table 3. Aspect extraction (AE) results

Domain	Laptop	Rest16
Methods	F1	F1
THA/STN (Li et al., 2018)	79.52	73.61
DE-CNN (Xu et al., 2018)	81.59	74.37
BERT (Devlin et al., 2018)	79.28	74.10
BERT-PT (Xu et al., 2019)	84.26	77.97
BERT-PT (best)	**84.88**	**80.69**
BAT (Ours)	**85.57**	**81.50**

Table 4. Aspect sentiment classification (ASC) results. Acc: Accuracy; MF1: Macro-F1.

Domain	Laptop		Rest14	
Methods	Acc	MF1	Acc	MF1
MGAN (Li et al., 2018)	76.21	71.42	81.49	71.48
BERT (Devlin et al., 2018)	75.29	71.91	81.54	71.94
BERT-PT (Xu et al., 2019)	78.08	75.08	84.95	76.96
BERT-PT (best)	**78.89**	**75.89**	**85.92**	**79.12**
BAT (Ours)	**79.35**	**76.5**	**86.03**	**79.24**

of restaurant dataset. To be consistent with BERT-PT, we performed the same selection.

In order to compare the effect of adversarial examples on the performance of the model, we choose the best dropout for each number of epochs and experiment with five different values for epsilon (perturbation size). The results for laptop and restaurant can be seen in Figure 4. As is noticeable, in terms of scores, they follow the same pattern as the original ones. Although most of the epsilon values improve the results, it can be seen in Figure 4 that not all of them will enhance the model's performance. In the case of $\epsilon = 5.0$ for AE, while it boosts the performance in the restaurant domain for most of the training epochs, it negatively affects the performance in the laptop domain. The reason for this could be the creation of adversarial examples which are not similar to the original ones but are labeled the same. In other words, the new examples greatly differ from the original ones but are fed to the net as being similar, leading to the network's poorer performance.

Observing, from AE task, that higher dropouts perform poorly, we experiment with the 5 lower values for ASC task in BERT-PT experiments. In addition, for BAT experiments, two different values $(0.01, 0.1)$ for epsilon are tested to make them more diverse. The results are depicted in Figures 5 and 6 for BERT-PT and BAT, respectively. While in AE, towards higher number of training epochs, there is an upward trend for restaurant and a downward trend for laptop, in ASC a clear pattern is not observed. Regarding the dropout, lower values (0.1 for laptop, 0.2 for restaurant) yield the best results for BERT-PT in AE task, but in ASC a dropout probability of 0.4 results in top performance in both domains. The top performing epsilon value for both domains in ASC, as can be seen in Figure 6, is 5.0 which is the same as the best value for restaurant domain in AE task. This is different from the top performing $\epsilon = 0.2$ for laptop in AE task which was mentioned above.

From the ablation studies, we extract the best results of BERT-PT and compare them with those of BAT. These are summarized in Tables 3 and 4 for aspect extraction and aspect sentiment classification, respectively. As can be seen in Table 3, the best parameters for BERT-PT have greatly improved its original performance on restaurant dataset (**+2.72**) compared to laptop (**+0.62**). Similar improvements can be seen in ASC results with an increase of **+2.16** in MF1 score for restaurant compared to **+0.81** for laptop which is due to the increase in the number of training epochs for restaurant domain since it exhibits better results with more training while the model reaches its peak performance for laptop domain in earlier training epochs. In addition, applying adversarial training improves the network's performance in both tasks, though at different rates. While for laptop there are similar improvements in both tasks (**+0.69** in AE, **+0.61** in ASC), for restaurant we observe different enhancements (**+0.81** in AE, **+0.12** in ASC). This could be attributed to the fact that these are two different datasets whereas the laptop dataset is the same for both tasks. Furthermore, the perturbation size plays an important role in performance of the system. By choosing the appropriate ones, as was shown, better results are achieved.

7. Conclusion

In this paper, we introduced the application of adversarial training in Aspect-Based Sentiment Analysis. The experiments with our proposed architecture show that the performance of the post-trained BERT on aspect extraction and aspect sentiment classification tasks are improved by utilizing adversarial examples during the network training. As future work, other white-box adversarial examples as well as black-box ones will be utilized for a comparison of adversarial training methods for various sentiment analysis tasks. Furthermore, the impact of adversarial training in the other tasks in ABSA namely Aspect Category Detection and Aspect Category Polarity will be investigated.

Acknowledgment

We would like to thank Adidas AG for funding this work.

References

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

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.

J. Ebrahimi, A. Rao, D. Lowd, and D. Dou. Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*, 2017.

P. Gamallo and M. Garcia. Citius: A naive-bayes strategy for sentiment analysis on english tweets. In *Proceedings of the 8th international Workshop on Semantic Evaluation (SemEval 2014)*, pages 171–175, 2014.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier. An unsupervised neural attention model for aspect extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 388–397, 2017.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

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

A. Ilyas, L. Engstrom, A. Athalye, and J. Lin. Black-box adversarial attacks with limited queries and information. *arXiv preprint arXiv:1804.08598*, 2018.

Y. Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

S. Kiritchenko, X. Zhu, C. Cherry, and S. Mohammad. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 437–442, 2014.

Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

X. Li, L. Bing, P. Li, W. Lam, and Z. Yang. Aspect term extraction with history attention and selective transformation. *arXiv preprint arXiv:1805.00760*, 2018.

Z. Li, Y. Wei, Y. Zhang, X. Zhang, and X. Li. Exploiting coarse-to-fine task transfer for aspect-level sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4253–4260, 2019.

P. Liu, S. Joty, and H. Meng. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1433–1443, 2015.

Y. Ma, H. Peng, and E. Cambria. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive lstm. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

T. Miyato, A. M. Dai, and I. Goodfellow. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*, 2016.

K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. In *IJCAI-99 workshop on machine learning for information filtering*, volume 1, pages 61–67, 1999.

M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland, Aug. 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2004. URL https://www.aclweb.org/anthology/S14-2004.

M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, A.-S. Mohammad, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq, et al. Semeval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 19–30, 2016.

S. Poria, E. Cambria, and A. Gelbukh. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems*, 108:42–49, 2016.

A. Rietzler, S. Stabinger, P. Opitz, and S. Engl. Adapt or get left behind: Domain adaptation through bert language model fine-tuning for aspect-target sentiment classification. *arXiv preprint arXiv:1908.11860*, 2019.

Y. Song, J. Wang, T. Jiang, Z. Liu, and Y. Rao. Attentional encoder network for targeted sentiment classification. *arXiv preprint arXiv:1902.09314*, 2019.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

T. T. Thet, J.-C. Na, and C. S. Khoo. Aspect-based sentiment analysis of movie reviews on discussion boards. *Journal of information science*, 36(6):823–848, 2010.

I. Titov and R. McDonald. A joint model of text and aspect ratings for sentiment summarization. In *proceedings of ACL-08: HLT*, pages 308–316, 2008.

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*, pages 5998–6008, 2017.

S. Wakade, C. Shekar, K. J. Liszka, and C.-C. Chan. Text mining for sentiment analysis of twitter data. In *Proceedings of the International Conference on Information and Knowledge Engineering (IKE)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer , 2012.

A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Y. Wang, M. Huang, L. Zhao, et al. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615, 2016.

Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

H. Xu, B. Liu, L. Shu, and P. S. Yu. Double embeddings and cnn-based sequence labeling for aspect extraction. *arXiv preprint arXiv:1805.04601*, 2018.

H. Xu, B. Liu, L. Shu, and P. S. Yu. Bert post-training for review reading comprehension and aspect-based sentiment analysis. *arXiv preprint arXiv:1904.02232*, 2019.

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

P. Zhaoa, L. Houb, and O. Wua. Modeling sentiment dependencies with graph convolutional networks for aspect-level sentiment classification. *arXiv preprint arXiv:1906.04501*, 2019.