DeepADA: An Attention-Based Deep Learning Framework for Augmenting Imbalanced Textual Datasets

Amit Kumar Sah
Department of Computer Science
South Asian University
New Delhi, India
amitcsrs@students.sau.ac.in

Muhammad Abulaish
Department of Computer Science
South Asian University
New Delhi, India
abulaish@sau.ac.in

Abstract

In this paper, we present an attention-based deep learning framework, DeepADA, which uses data augmentation to address the class imbalance problem in textual datasets. The proposed framework carries out the following functions: (i) using MPNET-based embeddings to extract keywords out of documents from the minority class, (ii) making use of a CNN-BiLSTM architecture with parallel attention to learn the important contextual words associated with the minority class documents' keywords and provide them with word-level characteristics derived from their statistical and semantic features, (iii) using MPNET, replacing the key contextual terms derived from the oversampled documents that match to a keyword with the contextual term that best fits the context, and finally (iv) oversampling the minority class dataset to produce a balanced dataset. Using a 2-layer stacked BiLSTM classifier, we assess the efficacy of the proposed framework using the original and oversampled versions of three Amazon's reviews datasets. We contrast the proposed data augmentation approach with two state-of-the-art text data augmentation methods. The experimental results reveal that our method produces an oversampled dataset that is more useful and helps the classifier perform better than the other two state-of-the-art methods. Nevertheless, we discover that the oversampled datasets outperformed their original ones by a wide margin.

1 Introduction

Textual datasets from different domains generally suffer from the issue of class imbalance, where instances from the majority class outnumber the instances from the minority class by a huge margin. In such a case, the classifier cannot perform well on the minority class dataset; as a result, the minority class instances go undetected. Most research works handle this issue by simply using a random oversampling algorithm without injecting additional knowledge into the minority class dataset. As a result, the resulting model is highly susceptible to overfitting. Nevertheless, a data augmentation-based minority class oversampling approach to handle a class imbalance in a textual dataset has not been well investigated.

Generally, the augmentation techniques generate undesirable documents that do not share the similar distribution of syntax, semantics, and pragmatics of the original data. When it comes to text data augmentation, class-indicating words (keywords) play a significant role (Abulaish and Sah, 2019). We must selectively augment the text, considering the significance of different words. In this process - (i) we should be able to identify the important class-indicating words so that the newly generated document semantically revolves around the class of the original document, (ii) we should be able to characterize the important words that aid the identification class indicating words and ultimately improves the generalization ability.

Deep learning models have emerged as effective classification models and are successful in many domains (Krizhevsky et al., 2012; Huang et al., 2021; Fazil et al., 2021), and have robust pattern learning ability and are widely successful for classification tasks. This paper presents a deep learning-based text data augmentation approach that exploits different deep learning techniques to create a balanced dataset by augmenting the newly generated documents to the minority class dataset to improve the detection efficacy of the classification algorithms on the minority class. The proposed approach first identifies keywords from the minority class data points utilizing a ranking-based weighted approach. Using an attention mechanism, it exploits the identified keywords to extract important contextual words from minority-class documents. In this process, it recognizes the word roles using statistical correlation to measure word occurrence frequencies respective to text categories

Proceedings of the 19th International Conference on Natural Language Processing (ICON), pages 318 - 327

and semantic similarity to measure word semantics respective to text categories, which helps to find the words semantically similar to text labels. Finally, it utilizes important contextual words to enrich the minority class dataset. The proposed approach oversamples the minority class dataset by generating new documents based on important contextual words and augmenting them to the minority class dataset. The proposed approach seems interpretable and improves the performance of the deep learning classifiers over the augmented datasets.

The rest of the paper is organized as follows. Section 2 presents a brief review of the existing literature on text data augmentation. Section 3 presents a detailed description of the proposed attention-based text data augmentation approach. Section 4 presents the experimental setup and evaluation results. It also presents a comparative analysis of the proposed approach with two state-of-the-art text data augmentation approaches. Finally, section 5 concludes the paper with future research directions.

2 Related Works

Researchers have come up with many approaches including words substitution-based (Wei and Zou, 2019; Kobayashi, 2018; Wu et al., 2019; Abulaish and Sah, 2019), paraphrasing-based (Sennrich et al., 2016; Edunov et al., 2018), and text generation-based (Anaby-Tavor et al., 2020; Liu et al., 2020) solutions.

In (Wei and Zou, 2019), authors introduced Easy Data Augmentation (EDA), a widely used word-replacement based data augmentation method with four basic randomization operations – (i) replacement, (ii) insertion, (iii) swap, and (iv) deletion. They have visualized that these simple operations can improve a classifier's performance on text classification tasks. In (Kobayashi, 2018), authors proposed contextual augmentation for labeled sentences to predict words from a wide range of substitute words, learned using a label-conditional bidirectional language model. In (Abulaish and Sah, 2019), authors showed that augmenting $n-$grams from a minority class document to itself that includes minority class keywords using Latent Dirichlet Allocation (LDA), if any, in the document, can improve the CNN's ability to identify the minority class instances. In (Wu et al., 2019), authors proposed to identify substitute words according to their context, which, apart from the similar meaning, also cares whether the candidates fit in the surrounding context and labels. In (Miao et al., 2020), authors exploited data augmentation to automatically create more labeled training data to fine-tune a language model to derive each aspect-opinion pair's sentiment. In (McCoy et al., 2019), authors insisted on creating new linguistic patterns for text data augmentation using large pre-trained language models.

3 Proposed Attention-Based Deep Learning Framework

In this section, we discuss the proposed attention-based deep learning framework, `DeepADA`, for data augmentation to improve the performance of classifiers on imbalanced text datasets. We evaluate the proposed approach over 3 Amazon reviews dataset, with statistics as shown in Table 1. Table 1 clearly shows that the Amazon reviews dataset suffers from class imbalance problem, as the number of positive reviews is significantly higher than the number of negative reviews. With this inference, we label the positive reviews dataset as the majority class dataset (D_{maj}) and the negative reviews dataset as the minority class dataset (D_{min}). The `DeepADA` performs the following functions:

(i) Extracts keywords out of documents from the minority class, using embeddings generated by a transformer-based language model, as discussed in section 3.1.

(ii) Makes use of a CNN-BiLSTM architecture with parallel attention to learn the important contextual words associated with the minority class documents' keywords, as discussed in section 3.2.

(iii) Using a transformer-based language model, replaces the key contextual terms derived from the oversampled documents that match to a keyword with the contextual term that best fits the context, as discussed in section 3.3.

(iv) Finally, oversamples the minority class dataset to produce a balanced dataset, as discussed in section 3.4.

3.1 Similarity-Based Weighted Keywords Extraction

This section discusses the keyword extraction mechanism that we use in our proposed approach. In order to extract keywords, we use MPNET

(Song et al., 2020), a semantically and contextually robust word embedding technique. MPNET is a transformer-based language model designed to capture the meaning of words, phrases, and documents by encoding them to vectors by inheriting the advantage of both masked language modeling (MLM), adopted in BERT, and permuted language modeling (PLM), adopted in XLNet. In order to generate the embeddings, we use SBERT (Reimers and Gurevych, 2019), which uses siamese and triplet network structures and has proven to be a successful bi-encoder model for generating semantically meaningful sentence embeddings that we can utilize for textual similarity comparisons using cosine similarity. We extract keywords from the minority class dataset (D_{min}) to employ them for their corresponding important contextual words extraction, which we utilize to generate new minority class documents.

In order to extract keywords from the minority class dataset, we first encode each j^{th} document d_j in the minority class dataset (D_{min}) to its embedding vector using SBERT, to extract semantically more meaningful sentence-level embeddings. We consider MPNET as a pre-trained model for SBERT. Then, we encode each i^{th} word w_i from the minority class vocabulary (Vb_{min}) to its embedding vector using MPNET. Thereafter, we calculate cosine similarity ($CSVal$) between each i^{th} word, $w_i \in Vb_{min}$ and j^{th} document $d_j \in D_{min}$, to give $CSVal(w_i, d_j)$ as given by equation 1; where $\overrightarrow{w_i}$ is the embedding vector corresponding to w_i, $\overrightarrow{d_j}$ is the embedding vector corresponding to d_j, and $CSVal(w_i, d_j) \in [-1, 1]$.

$$CSVal(w_i, d_j) = \frac{\overrightarrow{w_i} \cdot \overrightarrow{d_j}}{\|\overrightarrow{w_i}\| \|\overrightarrow{d_j}\|} \quad (1)$$

Once we have calculated $CSVal$ of each word from Vb_{min} with each document in D_{min} comparing their embeddings, we sort them in order such that the words in Vb_{min} which share higher $CSVal$ with most of the documents in D_{min} is ranked higher. In order to achieve this, we assign a score to each word based on its rank. The word score (WS) of word $w \in Vb_{min}$, $WS(w)$ is given by equation 2; where $r(w, d_i)$ is the rank of word w corresponding to i^{th} document $\in D_{min}$, n is the number of documents in D_{min}, and $\mathcal{P}$ is a variable vital in assigning score to a word w based on its rank values corresponding to documents $\in D_{min}$, and it basically penalizes more if the rank corresponding to a document is less; here, we choose $\mathcal{P} = 10$ over other values based on experimental fine-tuning.

$$WS(w) = \sum_{i=1}^{n} r(w, d_i) \times \mathcal{P}^{r(w,d_i)-1} \quad (2)$$

We select only the top k words from Vb_{min} based on their WS value as the minority class keywords (K_{min}).

3.2 Important Contextual Words Extraction from Minority Class Documents

In order to generate additional documents in order to augment it to the minority class D_{min}, we identify the important contextual words corresponding to each minority class document containing the keyword(s). Towards this, we first create a labeled dataset based on the presence of keyword(s) in the minority class documents as discussed in section 3.2.1. We then extract additional word-level features of each word $w \in Vb_{min}$ based on its semantic and statistical property as discussed in section 3.2.2. After that, we learn the important contextual words using a parallel attention-based CNN-BiLSTM model corresponding to the keywords from the keywords-based labeled dataset enriched with the word-level features as discussed in section 3.2. Figure 1 illustrates the important contextual words extraction process from minority class documents.

3.2.1 Keywords-Based Labeled Dataset Creation

In this section, we discuss the creation of a binary labeled dataset for important contextual words extraction D_{icwe} from the minority class dataset D_{min}. We aim to extract important words that aid the classifier most in classifying the keywords. To this end, for each keyword $kw \in K_{min}$, starting from the keywords with the highest $CSVal$ in descending order, we check if a document $r \in D_{min}$, oversample it corresponding to each word $w \in r$. We label class K to the oversampled document if $w = kw$ and assign it to D^{K}_{icwe}, and class NK to the oversampled document if $w \neq kw$ and assign it to D^{NK}_{icwe}. We continue this process until the total number of documents in the minority class dataset, and important words extraction dataset combined is equal to the number of documents in the majority class dataset, i.e., $|D_{min}| + |D^{K}_{icwe}| = |D_{maj}|$.

3.2.2 Word-Level Feature Extraction

In this section, we present the two effective word-level features that depict a word's association with different classes of the dataset based on its statistical and semantic properties. Since we aim to identify the important contextual words corresponding to each $kw \in K_{min}$, we aim to develop a robust classification framework that identifies accurate important contextual words.

(i) **Class Correlation:** This measures how frequently the word co-occurs with the different classes. If the words' frequency is higher in the K class, then its class correlation value is higher for the K class than for the NK class. The class correlation value of a word w corresponding to the class K, $CC(w, K)$ is nothing but weighted log-likelihood ratio, and is given by equation 3; where $p(\frac{w}{K})$ is the probability of observing word w in class c while $p(\frac{w}{K^{\complement}})$ is the probability of observing word w in class other than K.

$$CC(w, K) = p(\frac{w}{K}) \times log(\frac{p(\frac{w}{K})}{p(\frac{w}{K^{\complement}})}) \quad (3)$$

Here, for each word w, we calculate $CC(w, K)$ and $CC(w, NK)$ corresponding to the classes K and NK, respectively.

(ii) **Semantic Similarity:** This measures how much semantics the word w share with the label of a class. We take the semantic similarity of a word w with keyword class K as the cosine similarity value of word w with the semantic score of class K, i.e., $CSVal(w, SS(K))$, similar to the calculation in equation 1, and $SS(K)$ as the average of the word vectors of top 100 words $\in K_{min}$ in order of their WS value. We take the top 100 keywords from respective classes to calculate the classwise semantic scores since they represent the better semantic space being the centroid point of top keywords extracted using a more semantically sound approach. Here, for each word w, we calcuate $CSVal(w, SS(K))$ and $CSVal(w, SS(NK))$ corresponding to the classes K and NK respectively.

In this context, providing word-level features to the documents in D_{icwe} will help identify the important contextual words more effectively by exploiting their statistical and semantic property, which depicts their alignment to a particular category.

3.2.3 Keywords-Specific Important Contextual Words Extraction

In this section, we discuss how we extract important contextual words corresponding to the minority class keywords (K_{min}). As from section 3.2.1, we already know that each document d in D_{icwe} is created corresponding to a target word $w_t \in d$ and labeled class K or NK based on whether w_t is in K_{min} or not. After that, in section 3.2.2, we identified two word-level features based on their statistical and semantical properties. In this section, we discuss how we identify the top contextual words that help classify the target word w_t corresponding to which a document $d \in D_{icwe}$ has been created, using a parallel attention-based CNN-BiLSTM model, to ultimately identify the top contextual words corresponding to document $d \in D^{NK}_{icwe}$. We choose the CNN-BiLSTM model to exploit the benefit of both the CNN's feature extraction ability along with the BiLSTM's ability to learn long-term in textual documents (Liu and Guo, 2019; Rhanoui et al., 2019).

Let us suppose d_i is the i^{th} document, and $d_i \in D_{icwe}$ such that $d_i = \{w_1, \ldots, w_t, \ldots, w_n\}$; w_t and n being the target word and the number of words in the document respectively. `DeepADA` aims to learn the importance of each contextual word $w \in d_i$ while training the model on d_i with emphasis on w_t, where w_t is a target word corresponding to which $d_i \in D_{icwe}$ has been generated and labeled, as discussed in section 3.2. To this end, we have two parallel attention-based CNN, followed by 2 layers stacked BiLSTM, one encoding the preceding context of the document (ENC_p), and the other the following context of the target word (ENC_f) given by equations 4 and 5 respectively.

$$h^p_{w_t} = ENC_p(w_t, h^p_{w_{t-1}}) \quad (4)$$

$$h^f_{w_t} = ENC_f(w_t, h^f_{w_{t-1}}) \quad (5)$$

where ENC_p and ENC_f are two employed CNN-BiLSTM that model the preceding and following context of the target word independently.

With the help of the attention mechanism, variable weights are assigned to all words from the

beginning of the document to the target word (encoded by ENC_p), and from the target words towards the end of the document (encoded by ENC_f), depending on their contextual importance.

For encoded vector V_{d_i} of document $d_i \in D_{icwe}$; if hidden state representation of a target word $w_t \in V_{d_i}$ given by the attention-based CNN-BiLSTM classifier is h_{w_t} , then it is passed to a dense-layer to learn its hidden representation h'_{w_t} , as given by equation 6, where W and B represent the weight and bias, respectively. Thereafter, similarity is calculated between h_{w_t} and a vertex vector v_{w_t} which represents the importance of $w_t \in V_{d_i}$. We compute the normalized importance score of w_t using equation 7. The feature-level context vector v_{w_t} is randomly initialized and jointly learned during the training process. Finally, the attention-aware representation of the document d_i is learned and represented as $\mathcal{A}$. It is computed as a weighted sum of the hidden representation of each word, as given by equation 8.

$$h'_{w_t} = tanh(Wh_{w_t} + B) \tag{6}$$

$$\alpha_{w_t} = \frac{\exp(h'_{w_t} v_{w_t})}{\sum_w \exp(h'_{w_t} v_{w_t})} \tag{7}$$

$$\mathcal{A}_{d_i} = \sum_w \alpha_{w_t} h_{w_t} \tag{8}$$

Both ENC_p and ENC_f pass through the processes in equations 6, 7, and 8 simultaneously. The attention-based representation corresponding to ENC_p and ENC_f for document d_i are represented as $\mathcal{A}^p_{d_i}$ and $\mathcal{A}^f_{d_i}$. After we get the attention-based representation vector of the current word in both directions ($\mathcal{A}^p_{d_i}$ and $\mathcal{A}^f_{d_i}$), then we concatenate these two vectors to generate the final representation vector of the document d_i, pass it through a dense layer with 1024 neurons and finally through a softmax layer with 2 neurons. This is done in order to make the model learn and be able to identify the target word given the attention-based weight distribution of the contextual words.

We train the parallel attention-based CNN-BiLSTM model on D_{icwe} dataset. Once we have trained the model, we extract the attention-based vectors $\mathcal{A}^p$ and $\mathcal{A}^f$. These vectors are the attention scores corresponding to words on both sides of the target word w_t. We rank the top words on both sides of w_t based on their attention scores. In this work, we have selected the top 15% words corresponding to both the ENC_p and ENC_f.

3.3 Language Model-Based Documents' Transformation

In this section, we discuss the process of language model-based transformation of documents in D^k_{icwe}. We aim to transform a document $r \in D^k_{icwe}$ to r_t such that the transformed document r_t is a non-duplicate version of r, ensuring that words that are replaced from r to give r_t are contextually similar and gives the semantically similar meaning as r. To this end, we deploy *Fill-Mask* task supported by MPNET, where some of the words in a sentence are masked, and the MPNET model predicts which words best replaces the current word, also known as *mask language modeling*.

We replace the top k important words from each document $r \in D^k_{icwe}$, based on attention score as discussed in section 3.2. Now, for each i^{th} important word $I_{w_i} \in I_w$ where I_w is the list of important words corresponding to r, we learn its best substitute by masking and passing it through the MPNET model. We mask the words in their order of importance, i.e., attention score, and when we mask a word, the rest of the words remain the same. The MPNET model then gives the best word replacement for I_{w_i} in the form of $I^r_{w_i}$. We then replace I_{w_i} by $I^r_{w_i}$ and repeat this for every important word of the document r. In the end, we have the transformed document r_t where all the words $w \in I_w \cap r$ are replaced by their best contextual and semantically similar words given by the MPNET model.

3.4 Balanced Dataset Creation

In this section, we discuss the oversampling process of the minority class dataset D_{min} such that the number of instances in both classes of the dataset is equal. We already know from section 3.2.1 that D^k_{icwe} has been created such that augmenting it to D_{min} gives the balanced dataset. First, we transform each document $r \in D^k_{icwe}$ to give the transformed document r_t as discussed in section 3.3. Finally, we augment D^k_{icwe} with D_{min} to give oversampled minority class dataset AD_{min}, such that $|D_{maj}| = |AD_{min}|$. So, AD_{min} is the final augmented minority class dataset. We replace D_{min} with AD_{min} to give the oversampled balanced dataset.

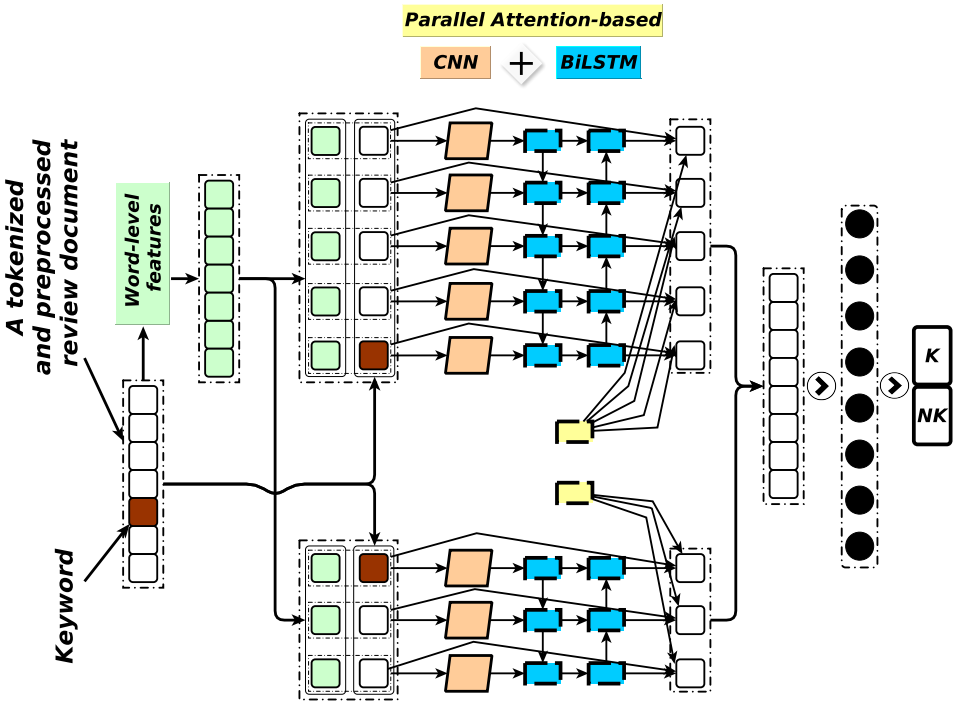

Figure 1: An overview of the important words extraction process

4 Experimental Setup and Results

In this section, we present our experimental setup and discuss the evaluation of the proposed approach. We mention that experiments were performed on a machine with a 2.10 GHz Intel(R) Silver(R) processor and 192G RAM. `DeepADA` was implemented in Keras[1]. For MPNET pre-trained models, we used Transformers[2] library.

4.1 Datasets

We use 3 publicly available Amazon reviews datasets (He and McAuley, 2016) to evaluate `DeepADA` The datasets *musical instruments* (DS_1), *patio lawn and garden* (DS_2), and *automotive* (DS_3) have overall rating on a scale of 1 to 5, 1 being the lowest and 5 being the highest rating provided by the customer, respectively. In this work, we modify this to binary by labeling ratings 1 or 2 as negative reviews and ratings 3, 4, or 5 as positive reviews. The statistics of the modified datasets are shown in Table 1.

[1]https://keras.io/

[2]https://huggingface.co/docs/transformers/index

Table 1: Statistics of the Amazon review datasets

Dataset	#Reviews	#D_{maj}	#D_{min}	IR
DS_1	10,261	9,794	467	20.97
DS_2	13,272	12,080	1,192	10.13
DS_3	20,473	19,325	1,148	16.83

Review documents differ significantly from standard grammatical structures, and the character limitations compel users to develop creative spellings. Such data needs to be preprocessed more carefully to avoid semantic loss. The preprocessing tasks taken by us are: stop-words, URLs, and hashtag symbols removal, resolving elongated words, emoticons handling, resolving contractions, stemming, and lemmatization.

4.2 Classifier Architecture and Training Details

In this section, we present the classification technique used to validate the effectiveness of `DeepADA`. As discussed, the Amazon reviews datasets are divided into majority class (D_{maj}) and minority class (D_{min}). We used a $2-$layer stacked BiLSTM architecture with 256 cells each followed by 2 neurons in the final softmax layer, as we formulated this as a binary classification problem. Other parameters include Xavier Glorot initializer to assign initial weights, adam as an

optimizer, dropout to minimize the overfitting effect, with a probability value of 0.2 at the BiLSTM layer, ReLU as an activation function throughout the model, except in the output layer, where we used the softmax function, and $L2$ regularizer with a value of λ as 0.01 over the softmax loss function.

Table 2 gives the statistics of the total number of keywords extracted corresponding to different Amazon reviews datasets to generate the keyword-based labeled dataset, based on the discussion in section 3.2.1. For MPNET-related tasks, we have used the pre-trained model proposed in (Song et al., 2020). For classification tasks throughout this work, we have used $300-$dimensional GloVe embeddings trained on the *Common Crawl* dataset with 840B tokens.

Table 2: Number of keywords extracted corresponding to different Amazon review datasets

Dataset	#Keywords
DS_1	$2,246$
DS_2	$1,172$
DS_3	$2,484$

4.3 Evaluation Metrics

When it comes to the evaluation of imbalanced data, we have very few metrics to consider (Ferri et al., 2009). In the case of a skewed dataset, the usual evaluation metrics, like accuracy, overshadow the performance of the classifier on the minority class. So, we considered reporting the performance of the classification model used in our work only for the minority class and the macro-averaged ones. As evaluation metrics, we considered precision (PR), recall (DR), F_1 score (F_1), macro precision ($MacPR$), macro recall ($MacDR$), and macro F_1 ($MacF_1$). We chose these evaluation metrics to report the classifier's performance on the minority class and observe whether there is any highly adverse impact on the majority class of the dataset.

4.4 Comparison Approaches and Baseline

In order to establish the efficacy of the proposed model on imbalanced data, we performed a comparative performance evaluation of `DeepADA` with the following two standard text data augmentation techniques:

(1) **EDA – Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks** (Wei and Zou, 2019): In this work, authors have applied 4 simple text data augmentation operations namely – (i) synonym replacement, (ii) random insertion, (iii) random swap, and (iv) random deletion either randomly or regulated by variables in a document. They observed that the classifier's performance was improved on the augmented version of the datasets using these simple data augmentation mechanisms.

(2) **Contextual Augmentation – Data Augmentation by Words with Paradigmatic Relations** (Kobayashi, 2018): In this work, authors have presented a novel text data augmentation technique using different words given by a bi-directional language model and further introduced a label-conditional architecture into the language model. The proposed method produced various words compatibly with the labels of original texts and improved neural classifiers more than synonym-based augmentation. We refer to this work as CDA in the coming sections.

In order to study the effectiveness of incorporating the word-level features in the proposed approach, we created a baseline DeepADA$_b$ by removing word-level features from `DeepADA`.

4.5 Evaluation Results and Comparative Analysis

In order to evaluate `DeepADA`, we balanced the original datasets by oversampling their minority class with new documents generated using `DeepADA`, as discussed in section 3.4. For comparison, we have considered two state-of-the-art text data augmentation techniques, namely EDA (Wei and Zou, 2019) and CDA (Kobayashi, 2018) as well as a baseline `DeepADA`$_b$, which is similar to ablation study that simply excludes the word-level features from `DeepADA`, as discussed in section 4.4. We balanced the original datasets using all EDA, CDA, and `DeepADA`$_b$ for comparison purposes. We performed experimentation on the BiLSTM model discussed in section 4.2 for evaluation purpose. We trained the BilSTM model on 56%, validated it on 14%, and finally tested the model to observe its effectiveness after getting trained on 30% of the datasets. We performed this on both the original and balanced versions of the datasets. While training the BiLSTM, we set the maximum

Table 3: Comparative performance evaluation results of DeepADA on minority class

Approach	DS_1			DS_2			DS_3		
	PR	DR	F_1	PR	DR	F_1	PR	DR	F_1
Original Dataset	45.45	10.42	16.95	37.21	13.48	19.80	35.86	15.03	21.18
EDA (Wei and Zou, 2019)	95.37	98.92	97.11	90.86	97.90	94.25	90.35	98.69	94.33
CDA (Kobayashi, 2018)	95.65	97.15	96.39	**95.98**	94.03	95.00	94.95	97.31	96.12
DeepADA	**97.11**	**99.83**	**98.45**	92.98	98.43	**95.63**	**97.53**	99.41	**98.47**
DeepADA_b	96.63	99.66	98.12	92.59	**98.79**	95.59	97.34	**99.57**	98.44

Table 4: Macro comparative performance evaluation results of DeepADA

Approach	DS_1			DS_2			DS_3		
	$MacPR$	$MacDR$	$MacF_1$	$MacPR$	$MacDR$	$MacF_1$	$MacPR$	$MacDR$	$MacF_1$
Original Dataset	70.61	54.90	57.25	64.61	55.62	57.30	65.48	56.71	58.95
EDA (Wei and Zou, 2019)	97.12	97.03	97.04	94.17	93.56	93.71	94.72	95.87	95.16
CDA (Kobayashi, 2018)	96.44	96.46	96.45	95.13	95.09	95.10	96.15	96.15	96.16
DeepADA	**98.47**	**98.40**	**98.42**	**95.65**	**95.48**	**95.48**	**98.47**	**98.45**	**98.45**
DeepADA_b	98.14	98.06	98.08	95.64	95.43	95.43	98.45	98.42	98.42

epochs as 100 with the "early stopping" regularization technique to combat overfitting. We have reported the results obtained on test data.

Tables 3 and 4 give the detailed experimental results over minority class datasets and macro-averaged results over all the classes of the datasets, highlighting the performance of DeepADA in comparison to EDA and CDA as well as the baseline approach.

4.5.1 Minority Class Performance

Table 3 shows that the classifier's performance on the original imbalanced datasets was the worst. On the original imbalanced datasets, in terms of F_1 score, we can state that the classifier performance increased with the size of the original dataset. However, when we observed the PR and DR values on the original imbalanced datasets, the same assumption did not stand. It shows that the performance of the deep learning classifier on the original imbalanced datasets was not just abysmal but also noisy. The original imbalanced datasets reported highest PR value of 45.45 on DS_1, and the highest DR and F_1 values of 15.03 and 21.18, both on DS_3.

We were amazed by the classifier's performance on the balanced datasets obtained using any text data augmentation mechanisms. Among the result obtained on the balanced versions of the datasets, the lowest PR, DR and F_1 values were 90.35, 94.03 and 94.25 and were reported on DS_3 using EDA, DS_2 using CDA and DS_3 using EDA respectively; while the highest PR, DR and F_1 values were 97.53, 99.83 and 98.47 and were reported on DS_3, DS_1 and DS_3 all using DeepADA. Overall, DeepADA performed the best in terms of F_1 value. We observed that the performance of DeepADA_b was at par with DeepADA on all evaluation measures over all the datasets. In some cases, like on DS_2 and DS_3, it even outperformed DeepADA in terms of DR value. Out of EDA and CDA, EDA outperformed CDA in terms of DR value on all the datasets, while at the same time, CDA surpasses EDA in terms of PR value on all the datasets. Overall, among EDA and CDA, CDA performed comparatively better than EDA in terms of F_1 value on all datasets except on DS_1. After studying Table 3 in detail, we can conclude that performance over the balanced version of all the datasets created using DeepADA was the best, while the baseline DeepADA_b performed second.

4.5.2 Macro-Averaged Performance

Table 4 shows that the classifier's macro averaged performance on both the classes of the original imbalanced datasets was worst, similar to that observed on the minority class. The original imbalanced datasets reported highest PR value of 70.61 on DS_1, and the highest DR and F_1 values of 56.71 and 58.95, both on DS_3. Compared to the performance on the minority class, the higher value of these evaluation measures signifies that the model can perform well on the majority class

but fails miserably to identify the minority class instances correctly.

Table 4 shows that similar to the performance reported on the minority class, the classifier's performance on the balanced datasets obtained using any text data augmentation mechanisms outperforms its performance on the original imbalanced datasets. The detailed study in Table 4 reveals that over all the datasets, the performance of DeepADA and DeepADA$_b$ over all the macro-averaged evaluation metrics were reported to be first and second best, respectively. Out of EDA and CDA, CDA outperformed EDA in terms of all the evaluation metrics over all the datasets except DS_1. It suggests that both the DeepADA and DeepADA$_b$ generate high-quality minority class documents, which, when augmented to the minority class dataset, gives a balanced dataset capable of making the classifier perform better on the minority class dataset without degrading its performance on the majority class.

5 Conclusion and Future Work

This paper presents a deep learning-based text data augmentation approach, DeepADA, to address the class imbalance issue of classifying textual datasets. The oversampled dataset generated using DeepADA can be helpful for deep learning models that extract patterns from the data. Experiments on different datasets show that DeepADA significantly outperforms the state-of-the-art methods. The ablation study in the form of the baseline DeepADA$_b$ reveals that statistical correlation and semantic similarity are essential for effective word selection. The performance observed on the minority class dataset, and the macro-averaged performance over the 3 datasets indicates that the classifier acquires stronger generalization ability when trained on oversampled datasets generated using DeepADA. Exploring more word-level features and extensive study on various transformer-based language models to generate more qualitative oversampled datasets seems a promising future direction of research.

References

Muhammad Abulaish and Amit Kumar Sah. 2019. A text data augmentation approach for improving the performance of cnn. In *11th International Conference on Communication Systems Networks (COMSNETS), Bangalore, India*, pages 625–630. IEEE.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7383–7390.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500.

Mohd Fazil, Amit Kumar Sah, and Muhammad Abulaish. 2021. DeepSBD: A deep neural network model with attention mechanism for socialbot detection. *IEEE Transactions on Information Forensics and Security*, 16:4211–4223.

C. Ferri, J. Hernández-Orallo, and R. Modroiu. 2009. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1):27–38.

Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, WWW, page 507–517, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Faliang Huang, Xuelong Li, Changan Yuan, Shichao Zhang, Jilian Zhang, and Shaojie Qiao. 2021. Attention-emotion-enhanced convolutional lstm for sentiment analysis. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–14.

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), Louisiana, USA*, pages 452–457.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems, Nevada, USA*, volume 25. Curran Associates, Inc.

Gang Liu and Jiabao Guo. 2019. Bidirectional lstm with attention mechanism and convolutional layer for text classification. *Neurocomputing*, 337:325–338.

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.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings*

of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, pages 3428–3448.

Zhengjie Miao, Yuliang Li, Xiaolan Wang, and Wang-Chiew Tan. 2020. Snippext: Semi-supervised opinion mining with augmented data. In *Proceedings of The Web Conference 2020*, pages 617–628.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese bert-networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China*, pages 3982–3992.

Maryem Rhanoui, Mounia Mikram, Siham Yousfi, and Soukaina Barzali. 2019. A cnn-bilstm model for document-level sentiment analysis. *Machine Learning and Knowledge Extraction*, 1(3):832–847.

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

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and permuted pre-training for language understanding. In *Advances in Neural Information Processing Systems*, volume 33, pages 16857–16867. Curran Associates, Inc.

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), Hong Kong, China*, pages 6382–6388.

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.