IndiText Boost: Text Augmentation for Low Resource India Languages

Shreyas Labhsetwar
slabhsetwar@ucsd.edu

Onkar Litake
olitake@ucsd.edu

Niraj Yagnik
nyagnik@ucsd.edu

1 Instructions

Data Augmentation is a process in NLP that enables us to artificially increase the training data size by generating various versions of the real data. Different downstreaming task require different data augmentation methods. We focus on data augmentation for text classification. To perform better on the classification task, the data must be changed such that it maintains the class categories.

The size of the training data is expanded through data augmentation, which enhances model performance. The model performs better the more data we have. The distribution of the generated augmented data should neither be too similar nor too different from the original.

Data augmentation strategy is used to deal with the problem of data scarcity. Through the years, large amount of work has been done on data augmentation for English language. In contrast very less work has been done on Indian languages, especially Hindi & Marathi. This is contrary to the fact that data augmentation is used to deal with data scarcity.

In this project, we are going to work on data augmentation for Indian language: Hindi & Marathi. This project stems from the motivation that no such work is carried out in the past. We are going to implement pre-existing approaches for data augmentation like Easy Data Augmentation, Back Translation, Paraphrasing, Document Generation etc. We plan on making some changes to these existing methods to tailor them specific for Indian languages.

This work will be helpful in lot of NLP tasks like News Classification, Hate Detection, Emotion Analysis, Sentiment Analysis, Spam Classification. All these tasks currently face the problem of data scarcity, which will be solved by our project. We aim to provide different methods for Data Augmentation. The user can implement different techniques and select what works best for the given use case. We have implemented various augmentation tasks for different languages, users can select any of these approach for their work.

2 What you proposed vs. what you accomplished

Proposed	**Status**
Data Collection for 5+ Indian Languages	Done (Collected 6)
Get the baseline scores for each language	Done
Implement EDA	Done
Implement Back-Translation	Done
Implement Text-Generation Augmentation method	Done
Implement Text-Rephrasing	Done
Implement Text-Expansion	Done
Test all the augmentation methods	Done

3 Related work

Data augmentation has been extensively researched in the field of Natural Language Processing (NLP) as a valuable approach to address the lack of training data [Feng et al., 2021]. One prominent technique, introduced by [Sennrich et al., 2015], is back translation, which enhances the performance of machine translation (MT) measured by BLEU scores [Papineni et al., 2002]. This method involves translating sentences into a different language and then translating them back into the original language, thereby augmenting the original text.

[Fadaee et al., 2017] present a data augmentation method specifically designed for low-frequency words. This method generates new sentence pairs that include these infrequently occurring words. [Kafle et al., 2017] contribute two data augmentation techniques for enhancing visual question answering. The first approach employs semantic annotations to augment the questions, while the second technique utilizes an LSTM network [Hochreiter and Schmidhuber, 1997] to generate new questions from images. [Wang and Yang, 2015] propose an augmentation technique that involves replacing query words with their synonyms. The synonyms are retrieved based on cosine similarities calculated using word embeddings. Additionally, [Kolomiyets et al., 2011] propose a data augmentation strategy that involves replacing temporal expression words with their corresponding synonyms. This approach relies on the vocabulary provided by the Latent Words Language Model (LWLM) and the WordNet.

[Şahin and Steedman, 2019] suggest two text augmentation methods that rely on dependency trees. The initial technique involves cropping sentences by removing specific dependency links. The second technique involves rotating sentences using tree fragments that pivot around the root. [Chen et al., 2020] propose a text augmentation approach that involves interpolating input texts within a hidden space. [Wang et al., 2018] propose a method for augmenting sentences by randomly substituting words in both input and target sentences with vocabulary words. SeqMix [Guo et al., 2020] proposes a technique for generating augmentations by smoothly merging input and target sequences.

EDA [Wei and Zou, 2019] employs four operations to create data augmentation: synonym replacement, random insertion, random swap, and random deletion. In a different approach, [Kobayashi, 2018] suggests stochastically replacing words with predictions generated by a bidirectional language model. [Andreas, 2019] proposes a compositional data augmentation technique that constructs synthetic training examples by substituting text fragments in a real example with other fragments appearing in similar contexts. [Kumar et al., 2020] utilize pretrained Transformer models, such as GPT-2, BERT, and BART, for conditional data augmentation. They feed the concatenation of class labels and input texts into these models to generate augmented texts. Additionally, [Kumar et al., 2020] propose a language model-based data augmentation method. This method involves fine-tuning a language model on a limited training dataset and then using class labels as input to generate augmented sentences. [Min et al., 2020] explore various syntactically informative augmentation techniques by applying syntactic transformations to original sentences and demonstrate that subject/object inversion can enhance robustness to inference heuristics.

4 Dataset

The proposed work obtains data augmentation for a total of following 6 languages:

- Sindhi [Raza, year]: This dataset is a valuable resource for researchers in the field of Sindhi Natural Language Processing (NLP), as it represents one of the limited publicly accessible collections of Sindhi articles. It comprises a total of 3364 articles that span three distinct categories: sports, entertainment, and technology. The dataset was sourced from awamiawaz.pk, and its availability greatly facilitates investigations and advancements in Sindhi NLP research.
- Hostility Detection Dataset in Hindi [Bhardwaj et al., 2020]: The dataset contains 8200 hostile and non-hostile texts from social media platforms like Twitter, Facebook, and WhatsApp. The nostile class is further subdivided into Fake, Offensive, Hate, Defamation.
- L3Cube-MahaHate - Marathi [Velankar et al., 2022]: Dataset consists of over 25000 distinct tweets labeled into four major classes i.e hate, offensive, profane, and not-hate.
- Gujarati News Dataset - Gujarati [Arora, 2020]: his data set contains 6500 news article headlines which are collected from Gujarati news websites. It contains 3 labels namely entertainment, business, technology.
- Multi domain corpus for sentimental analysis - Telugu 2 Class [Gangula and Mamidi, 2018]: It contains 339 different Telugu song lyrics written in Telugu script. Out of them 230 are positive and 109 are negative. It contains a total of 13997 sentences and 81798

words.

- Telugu NLP - Telugu 5 Class [Rajkumar, year]: The data is extracted from Telugu book. It contains the following labels: business, editorial, entertainment, nation, sport.

For each of the six languages, the proposed work performs data augmentations for two tasks: i) Binary Classification and ii) Multi-Class Text Classification. Thus a total of 12 datasets are collected for the research and experimentations. One hundred examples are sampled from each dataset as the train set. The reason behind selecting the lower number of sentences is to replicate the scenario of implementing the augmentation techniques on low-resource Indian languages, which won't have sufficient labeled data.

For languages that did not have a dataset for binary text classification, the work filters the multi-class dataset to include only two labels while maintaining class balance. The statistics for the binary and multi-class text classification tasks are given in Table 1 and 2.

Binary Task		
Language	**Type of Task**	**Average number of words**
Sindhi	News Classification	157.02
Marathi	Hate Detection	48.89
Hindi	Hate Detection	52.61
Gujarati	News Classification	22.08
Telugu	Sentiment Analysis	71.98
Sanskrit	Document Classification	32.65

Table 1: Basic Statistics for Binary Text Classification Task

Multi-Class Task			
Language	**Type of Task**	**Number of labels**	**Avg no. of words**
Sindhi	News Classification	3	186.17
Marathi	Hate Detection	4	45.432
Hindi	Hate Detection	4	50.12
Gujarati	News Classification	3	21.41
Telugu	News Classification	3	354
Sanskrit	Doc Classification	3	24.636

Table 2: Basic Statistics for Multi-Class Text Classification Task

5 Baselines

For the baseline, for each language, we fine-tune the pre-trained BERT model (more specifically, the bert-base-multilingual-cased model) for the un-augmented data. The proposed work adopts a standard set of hyperparameters for each experiment. The list of hyperparameters for each model with the number of epochs being 3 and the max_length of 512.

The accuracy scores obtained by the fine-tuned BERT model for the augmented data of each language are recorded as the benchmark for all the augmentation techniques the work intends to utilize.

6 Methodology

The proposed work experiments with a series of data augmentation techniques to understand what is the most efficient method. The general pipeline to test the augmentation technique remains standard across all the methods and is illustrated in 1. Each augmentation task is provided with an initial 100 sets of text samples for each classification task and each language. We generate 1 augmentation for each sentence, hence a total of 100 augmented examples are generated. Post-generation of the new text samples, concatenation is performed to get the augmented text database. This augmented text is now used to fine-tune the BERT model as

used by the baseline models. The pre and post-augmentation accuracy scores are compared to deduce the efficacy of the augmentation technique. A deep-dive into each augmentation technique is provided:

6.1 Easy Data Augmentation(EDA)

[Wei and Zou, 2019] : The work first proposes to leverage EDA to provide a simple yet effective solution for generating augmented data. The technique aims to improve the generalization and efficacy of Machine Learning models by allowing the model to be trained on a diverse variety of the original day. The EDA method consists of four basic operations that can be applied to individual data samples and the proposed work intends to apply these techniques on Indian Languages:

- **Synonym Replacement:** For this technique, we will randomly select words from the input text of an Indian language and replace them with their corresponding synonyms. This technique will thus introduce semantic variations in the new augmented text while maintaining the meaning of the data. N words are randomly selected from the input sentence (that are not stop words); these words and then replaced by their synonyms.
- **Random Insertion:** This technique will find a random synonym of a random word in the sentence written in the Indian language (that is not a stop word). This synonym is inserted at an arbitrary position in the sentence.
- **Random Swap (RS)**: Randomly choose two words in the sentence and swap their positions. Do this n times.
- **Random Deletion (RD):** This technique randomly removes words in the text with a certain probability (p). The model is forced to rely on the remaining words after certain terms are eliminated, possibly leading to the learning of more robust representations.

6.2 Back-Translation:

[Sennrich et al., 2015] In this technique, the input text is translated from one language to another and is retranslated back to the same language. Upon back-translation, the back-translate sentences are mixed with the source sentences to train the model on augmented data. The illustation for back-translation is provided in 2.

The work utilizes the googletranslate api [goo, 2021] for the translation and back-translation tasks.

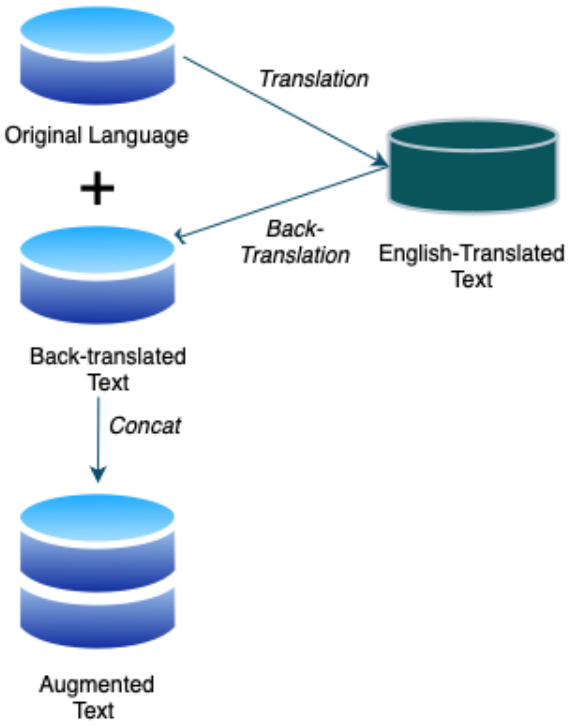

Figure 2: Backtranslation Technique

6.3 Paraphrasing:

[Andreas, 2019] With this technique, the proposed work employs Generative Large Language Models to rephrase the original input text while preserving the underlying context and meaning. More specifically, the GPT3.5 API [OpenAI, 2021] with the davinci by OpenAI is utilized for generating the paraphrased sentences. Each of the hundred text samples, along with their label, is used as the input to the prompt as indicated in 3. The langdetect library [Nakatani Shuyo, 2010] ensures the rephrased text complies with the language and length requirements.

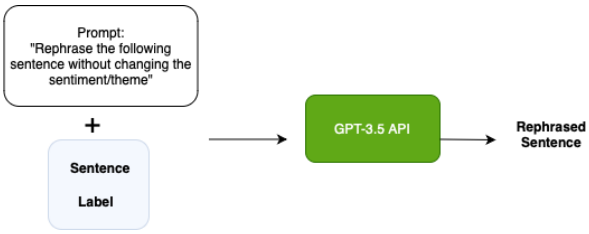

Figure 3: Paraphrasing Module

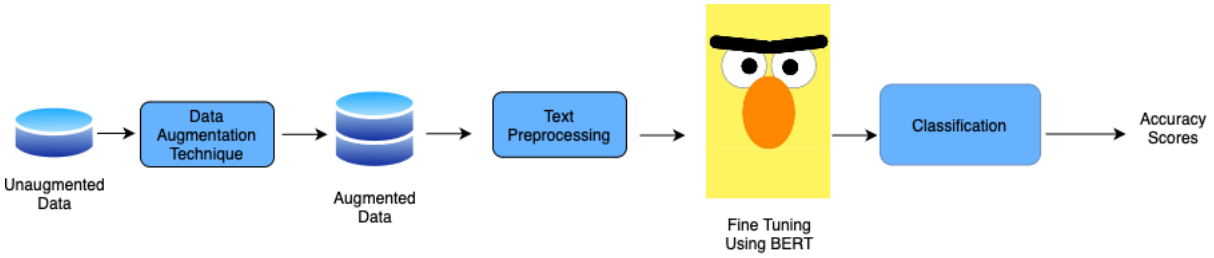

Figure 1: Finetuning BERT for augmented data

6.4 Text Extender: [Kumar et al., 2020]

In this technique, the input size is extended by using Language Models (GPT2 specifically) while maintaining the overall context and sentiment of the input. By adding more context and possibly offering fresh viewpoints or facts, this strategy helps lengthen the input.

6.5 Text Generation:

[Dai et al., 2023] The text Generation module, similar to the paraphraser, utilizes the OpenAI GPT3.5 API [OpenAI, 2021] and relies heavily on the prompt provided to the API. For the task of generating text similar to the input text and the associated label, a static prompt is provided to the API. The prompt offers an example text sentence associated with the label and asks the model to generate a text similar to the input text sentence. This process is repeated for each text sample on the original text dataset for each language.

All the proposed methods were successfully implemented and tested. The work utilizes Google Colab to conduct the experiments and make optimal use of the GPU. All the data augmentation techniques are applied for each classification task and language. The BERT model is fine-tuned for every augmented dataset obtained to make the comparisons.

7 Results

For Binary text classification, As seen in Table 3, all the data augmentation tasks for Sanskrit perform better than the baseline model, with Synonym replacement and random deletion being the best performers. Similarly, for Marathi, all the models exceed the performance of the baseline model, with Translate and Generate modules being the best performer. Similar trends of the augmentations models outperforming the baseline models are observed in every language except Hindi, where the baseline model itself generates a very high score for the very little data it was fine-tuned with. This increase in performance can be attributed to the availability of a large amount of Hindi data on the internet. The abundance of Hindi text data allows the model to learn from a vast range of unannotated text, which helps build strong language representations and capture relevant linguistic patterns. Apart from very few exceptions in Sindhi and Telugu text classification tasks, the augmented models perform better than the baseline model.

Similarly, for the multiclass text classification task, the augmentation methods outperform the baseline methods for every language with very few exceptions. Many augmentation techniques perform better than the baseline model and the binary classification task. Detailed results are given in Table 4

We can see from both the result tables that basic augmentation using Easy Data Augmentation outperforms advanced augmentation techniques like augmentation, Paraphrasing, Summarization, and generation.

8 Error analysis

We did not really encounter any failure cases in our analysis. The datasets which we used were highly cited/well curated ones, and fine-tuning HuggingFace's multi-lingual BERT was error-free as well.

Random Insertion technique for EDA works on the phenomemon that we randomly insert a synonym of a word(selected word should not be a stop word). For some langauges like Gujarati, Sindhi, Sanskrit there are no available list of stop words which we can exclude from the sentence. Hence, at times it is possible that a stopword might be replaced with snonym which in turn can reduce the accuracy. This can be seen for binary classification in Sindhi language. Here the F1 score drops

Language	Methodology	Accuracy	Precision	Recall	F1 Macro	F1 Mirco
Sanskrit	Baseline	0.53	0.265	0.5	0.346	0.53
	Synonym Replacement	0.73	0.73	0.73	0.73	0.729
	Random Insertion	0.62	0.618	0.616	0.617	0.62
	Random Swap	0.67	0.682	0.659	0.654	0.67
	Random Delete	0.73	0.731	0.7322	**0.729**	0.73
	LLM Expand	0.67	0.676	0.674	0.669	0.67
	Back-Translate	0.71	0.718	0.7023	0.7104	0.71
	Paraphrase	0.7	0.698	0.697	0.698	0.7
	Generate	0.617	0.624	0.617	0.621	0.617
Marathi	Baseline	0.656	0.664	0.656	0.652	0.656
	Synonym Replacement	0.666	0.693	0.666	0.678	0.667
	Random Insertion	0.686	0.701	0.686	0.693	0.687
	Random Swap	0.6862	0.686	0.686	0.686	0.686
	Random Delete	0.6568	0.657	0.656	0.656	0.656
	LLM Expand	0.696	0.696	0.696	0.696	0.696
	Back-Translate	0.725	0.738	0.725	**0.732**	0.725
	Paraphrase	0.705	0.717	0.7058	0.711	0.705
	Generate	0.715	0.719	0.715	0.717	0.715
Hindi	Baseline	0.91	0.910	0.91	0.909	0.91
	Synonym Replacement	0.94	0.940	0.94	**0.940**	0.94
	Random Insertion	0.9	0.9	0.9	0.9	0.9
	Random Swap	0.91	0.9101	0.91	0.909	0.91
	Random Delete	0.88	0.880	0.88	0.879	0.88
	LLM Expand	0.91	0.914	0.91	0.909	0.91
	Back-Translate	0.9	0.9	0.9	0.9	0.9
	Paraphrase	0.91	0.911	0.909	0.910	0.91
	Text Generate	0.92	0.9206	0.919	0.920	0.92
Gujarati	Baseline	0.528	0.542	0.53	0.535	0.528
	Synonym Replacement	0.85	0.853	0.85	**0.851**	0.85
	Random Insertion	0.84	0.854	0.84	0.847	0.839
	Random Swap	0.5	0.25	0.5	0.333	0.5
	Random Delete	0.85	0.853	0.85	0.849	0.85
	LLM Expand	0.83	0.831	0.83	0.829	0.83
	Back-Translate	0.71	0.766	0.71	0.737	0.71
	Paraphrase	0.66	0.69	0.66	0.678	0.66
	Text Generate	0.7	0.705	0.7	0.7	0.7
Telugu	Baseline	0.54	0.212	0.5	0.352	0.51
	Synonym Replacement	0.6	0.601	0.6	0.6	0.6
	Random Insertion	0.5	0.25	0.5	0.333	0.5
	Random Swap	0.68	0.673	0.666	0.637	0.67
	Random Delete	0.72	0.793	0.728	**0.772**	0.73
	LLM Expand	0.66	0.678	0.663	0.681	0.67
	Back-Translate	0.57	0.571	0.57	0.570	0.57
	Paraphrase	0.55	0.588	0.55	0.568	0.55
	Text Generate	0.52	0.63	0.52	0.569	0.52
Sindhi	Baseline	0.93	0.931	0.93	0.929	0.93
	Synonym Replacement	0.88	0.885	0.879	0.882	0.88
	Random Insertion	0.79	0.827	0.79	0.808	0.79
	Random Swap	0.96	0.962	0.96	0.959	0.96
	Random Delete	0.99	0.990	0.99	**0.989**	0.99
	LLM Expand	0.98	0.985	0.98	0.985	0.98
	Back-Translate	0.96	0.960	0.96	0.96	0.96
	Paraphrase	0.94	0.94	0.94	0.94	0.94
	Text Generate	0.95	0.951	0.95	0.950	0.95

Table 3: Results for the task of Binary Classification

Language	Methodology	Accuracy	Precision	Recall	F1 Macro	F1 Mirco
	Baseline	0.71	0.749	0.68	0.616	0.71
Sanskrit	Synonym Replacement	0.79	0.8072	0.808	**0.807**	0.79
	Random Insertion	0.78	0.787	0.777	0.782	0.78
	Random Swap	0.7	0.523	0.666	0.575	0.7
	Random Delete	0.77	0.775	0.772	0.773	0.77
	LLM Expand	0.77	0.778	0.780	0.778	0.77
	Back-Translate	0.71	0.712	0.705	0.708	0.71
	Paraphrase	0.74	0.748	0.75	0.749	0.74
	Text Generate	0.75	0.759	0.741	0.750	0.75
	Baseline	0.298	0.247	0.298	0.261	0.298
Marathi	Synonym Replacement	0.432	0.422	0.432	**0.427**	0.432
	Random Insertion	0.394	0.396	0.394	0.395	0.394
	Random Swap	0.278	0.187	0.278	0.191	0.278
	Random Delete	0.365	0.376	0.365	0.360	0.365
	LLM Expand	0.361	0.366	0.360	0.367	0.361
	Back-Translate	0.480	0.373	0.480	0.420	0.480
	Paraphrase	0.326	0.376	0.326	0.349	0.326
	Text Generate	0.326	0.319	0.326	0.323	0.326
	Baseline	0.53	0.407	0.53	0.455	0.53
Hindi	Synonym Replacement	0.55	0.516	0.55	0.532	0.55
	Random Insertion	0.58	0.618	0.58	**0.598**	0.58
	Random Swap	0.52	0.464	0.52	0.484	0.52
	Random Delete	0.58	0.568	0.58	0.549	0.58
	LLM Expand	0.59	0.592	0.59	0.581	0.59
	Back-Translate	0.56	0.537	0.56	0.548	0.56
	Paraphrase	0.53	0.5132	0.51	0.521	0.53
	Text Generate	0.51	0.5362	0.51	0.522	0.51
	Baseline	0.47	0.480	0.469	0.475	0.47
Gujarati	Synonym Replacement	0.51	0.382	0.506	0.435	0.51
	Random Insertion	0.68	0.684	0.679	0.681	0.68
	Random Swap	0.68	0.713	0.679	0.666	0.68
	Random Delete	0.78	0.786	0.780	**0.779**	0.78
	LLM Expand	0.57	0.573	0.58	0.572	0.57
	Back-Translate	0.58	0.585	0.581	0.583	0.58
	Paraphrase	0.54	0.544	0.54	0.54	0.54
	Text Generate	0.62	0.646	0.62	0.633	0.59
	Baseline	0.45	0.476	0.473	0.488	0.46
Telugu	Synonym Replacement	0.72	0.754	0.719	0.736	0.72
	Random Insertion	0.72	0.718	0.718	0.718	0.72
	Random Swap	0.67	0.728	0.674	0.663	0.68
	Random Delete	0.78	0.792	0.792	**0.782**	0.78
	LLM Expand	0.57	0.578	0.58	0.581	0.57
	Back-Translate	0.878	0.885	0.877	0.881	0.878
	Paraphrase	0.646	0.699	0.647	0.672	0.646
	Text Generate	0.747	0.773	0.747	0.760	0.747
	Baseline	0.49	0.492	0.489	0.489	0.49
Sindhi	Synonym Replacement	0.48	0.671	0.475	0.556	0.48
	Random Insertion	0.9	0.907	0.9	**0.904**	0.9
	Random Swap	0.57	0.574	0.568	0.558	0.57
	Random Delete	0.89	0.896	0.8903	0.890	0.89
	LLM Expand	0.76	0.783	0.7624	0.758	0.787
	Back-Translate	0.72	0.755	0.722	0.738	0.72
	Paraphrase	0.9	0.906	0.9f	0.903	0.9
	Text Generate	0.63	0.63	0.626	0.516	0.63

Table 4: Results for the task of Multiclass Classification

from 0.929 to 0.008.

Due to the generation limit of GPT, augmentation techniques which used Large Language Models like GPT, were incomplete when big sentences were passed as input. This is especially seen in the performance of the Text Generation module for the Telugu Binary Text Classification Task where no substantial model scores is obtained. The average length of the the original and augmented dataset is given in table 5 which reasons the lack of improvement.

	Original Train Set	**Augmented Train Set**	**Test Set**
Text Length	71.98	46.11	111.39

Table 5: Average Word length of binary Telugu text classification

9 Contributions of group members

List what each member of the group contributed to this project here. For example:

- Shreyas Labhsetwar: Collected Telugu 2-class and Telugu 5-class datasets, application of baselines, random swap, random delete, and LLM expand using GPT-2. Also did some report writing and inclusion of the results tables.
- Onkar Litake: Collected Marathi 2-Class, Marathi 4-Class, Hindi 2-Class, Hindi 4-Class, Gujarati 2-Class, Gujarati 3-Class. Worked on Data Augmentation techniques like Synonym Replacement, Random Insertion. Worked on report writing.
- Niraj Yagnik: Collected Sindhi 2-Class, Sindhi 4-Class, Telugu 2-Class, Telugu 3-Class datasets. Worked on the Back-Translation, Paraphrasing and Text Generation Modules of Data Augmentation using GPT3.5 API. Worked on report writing

10 Conclusion

From the results, we can see that the proposed eight methodologies for data augmentation consistently outperform the baseline on both binary and multiclass classification tasks across all the languages. Amongst these methods, EDA is a clear winner showing consistently great performance across all languages. For the task of binary classification, random delete, LLM Expand, and text generate methodologies show good performance overall as compared to other methods. A similar trend can also be observed in the case of multiclass classification, however, even Back-Translate demonstrates good performance here. We were pleasently surprised to see the almost consistent good performance demonstrated by random delete methodology, because at the face of it, random delete seems to be eliminating information from each sentence instead of supplementing it. In the future, we would love to carry these experiments out on larger chunks of the datasets, and would also love to back our experimental results with the underlying mathematics. Moreover, we would also like to experiment with ensemble models, which we believe would result in better performance.

References

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*, 2021.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*, 2015.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. Data augmentation for low-resource neural machine translation. *arXiv preprint arXiv:1705.00440*, 2017.

Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan. Data augmentation for visual question answering. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 198–202, 2017.

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

William Yang Wang and Diyi Yang. 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, 2015.

Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. Model-portability experiments for textual temporal analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, volume 2, pages 271–276. ACL; East Stroudsburg, PA, 2011.

Gözde Gül Şahin and Mark Steedman. Data augmentation via dependency tree morphing for low-resource languages. *arXiv preprint arXiv:1903.09460*, 2019.

Jiaao Chen, Zichao Yang, and Diyi Yang. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. *arXiv preprint arXiv:2004.12239*, 2020.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. Switchout: an efficient data augmentation algorithm for neural machine translation. *arXiv preprint arXiv:1808.07512*, 2018.

Demi Guo, Yoon Kim, and Alexander M Rush. Sequence-level mixed sample data augmentation. *arXiv preprint arXiv:2011.09039*, 2020.

Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

Sosuke Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*, 2018.

Jacob Andreas. Good-enough compositional data augmentation. *arXiv preprint arXiv:1904.09545*, 2019.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*, 2020.

Junghyun Min, R Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. Syntactic data augmentation increases robustness to inference heuristics. *arXiv preprint arXiv:2004.11999*, 2020.

Owais Raza. Awamiawaz sindhi articles classification dataset. https://www.kaggle.com/datasets/owaisraza009/awamiawaz-sindhi-articles-classification-dataset, year. Accessed: May 22, 2023.

Mohit Bhardwaj, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. Hostility detection dataset in hindi. *arXiv preprint arXiv:2011.03588*, 2020.

Abhishek Velankar, Hrushikesh Patil, Amol Gore, Shubham Salunke, and Raviraj Joshi. L3cube-mahahate: A tweet-based marathi hate speech detection dataset and bert models, 2022.

Gaurav Arora. iNLTK: Natural language toolkit for indic languages. In *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 66–71, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.nlposs-1.10. URL https://www.aclweb.org/anthology/2020.nlposs-1.10.

Rama Rohit Reddy Gangula and Radhika Mamidi. Resource creation towards automated sentiment analysis in Telugu (a low resource language) and integrating multiple domain sources to enhance sentiment prediction. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://aclanthology.org/L18-1100.

Sudalai Rajkumar. Telugu nlp dataset. https://www.kaggle.com/datasets/sudalairajkumar/telugu-nlp, year. Accessed: Month Day, Year.

Google Translate API. Google Cloud Documentation, 2021. URL https://cloud.google.com/translate. Accessed: June 12, 2023.

OpenAI. OpenAI GPT-3.5 API. OpenAI Documentation, 2021. URL https://platform.openai.com/docs/guides/gpt-3.5.

Nakatani Shuyo. langdetect. GitHub Repository, 2010. URL https://github.com/shuyo/language-detection.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, Hongmin Cai, Lichao Sun, Quanzheng Li, Dinggang Shen, Tianming Liu, and Xiang Li. Auggpt: Leveraging chatgpt for text data augmentation, 2023.