A Study of Various Text Augmentation Techniques for Relation Classification in Free Text

Praveen Kumar Badimala Giridhara[1,2,*], Chinmaya Mishra[1,2,*],
Reddy Kumar Modam Venkataramana[1,*], Syed Saqib Bukhari[2] and Andreas Dengel[1,2]
[1]*Department of Computer Science, TU Kaiserslautern, Gottlieb-Daimler-Straße 47, Kaiserslautern, Germany*
[2]*German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany*

Keywords: Relation Classification, Text Data Augmentation, Natural Language Processing, Investigative Study.

Abstract: Data augmentation techniques have been widely used in visual recognition tasks as it is easy to generate new data by simple and straight forward image transformations. However, when it comes to text data augmentations, it is difficult to find appropriate transformation techniques which also preserve the contextual and grammatical structure of language texts. In this paper, we explore various text data augmentation techniques in text space and word embedding space. We study the effect of various augmented datasets on the efficiency of different deep learning models for relation classification in text.

1 INTRODUCTION

Relation classification is an important task in processing free text using Natural Language Processing (NLP). The basic idea behind relation classification is to classify a sentence into a predefined relation class, given the two entities in the sentence. For instance, in the sentence; $< e1 >$ *Bill Gates* $< /e1 >$ *is the founder of* $< e2 >$ *Microsoft* $< /e2 >$,"Bill Gates" and "Microsoft" are the two entities denoted by "$< e1 >< /e1 >$" and "$< e2 >< /e2 >$" respectively. The relation classification process should be able to classify this sentence as belonging to the relation class *"founderOf"* (which is a predefined class). One of the major limitations in the field of NLP is the unavailability of labelled data. It takes a great deal of time and effort to manually annotate relevant datasets. Hence, it has become increasingly necessary to come up with automated annotation methods. This has led to the development of many semi-supervised annotation methods for generating annotated datasets. But, they fall short when compared to the quality and efficiency of a manually annotated dataset.

One workaround would be to perform data augmentations on manually annotated datasets. Data augmentation techniques are very popular in the field of image processing because of the ease in generating augmentations using simple image transformations. However, it is very challenging to find an appropriate method for text data augmentation as it is difficult to preserve grammar and semantics.

As per our knowledge, there have been no work that studies different text data augmentation techniques on relation classification for free text over different Deep Learning Models as yet. In this paper, we investigate various text data augmentation techniques while retaining the grammatical and contextual structure of the sentences when applying them. We also observe the behavior of using augmented datasets with the help of two different deep learning models namely, CNN (Zeng et al., 2014) and Attention based BLSTM (Zhou et al., 2016).

2 RELATED WORKS

Data Augmentation in the field of Image Processing and Computer Vision is a well-known methodology to increase the dataset by introducing varied distributions and increase the performance of the model for a number of tasks. In general, it is believed that the more the data a neural network gets trained on, the more effective it becomes. Augmentations are performed by using simple image transformations such as rotation, cropping, flipping, translation and addition of Gaussian noise to the image. Krizhevsky et al., (Krizhevsky et al., 2012) used data augmentation

*Contributed equally.

Giridhara, P., Mishra, C., Venkataramana, R., Bukhari, S. and Dengel, A.
A Study of Various Text Augmentation Techniques for Relation Classification in Free Text.
DOI: 10.5220/0007311003600367
In *Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2019)*, pages 360-367
ISBN: 978-989-758-351-3
Copyright © 2019 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

methods to increase the training data size for training a deep neural network on ImageNet dataset (Deng et al., 2009). The increase in training data samples showed reduced overfitting of the model (Krizhevsky et al., 2012) and increased the model performance. These techniques enable the model to learn additional patterns in the image and identify new positional aspects of objects in it.

On similar lines, data augmentation methods are explored in the field of text processing for improving the efficacy of models. Mueller and Thyagarajan (Mueller and Thyagarajan, 2016) replaced random words in a sentence with their respective synonyms, to generate augmented data and train a siamese recurrent network for sentence similarity task. Wang and Yang (Wang and Yang, 2015) used word embeddings of sentences to generate augmented data for the purpose of increasing data size and trained a multi-class classifier on tweet data. They found the nearest neighbour of a word vector by using cosine similarity and used that as a replacement for the original word. The word selection was done stochastically.

For information extraction, Papadaki (Papadaki, 2017) applied data augmentation techniques on legal dataset (Chalkidis et al., 2017). A class specific probability classifier was trained to identify a particular contract element type for each token in a sentence. They classified a token in a sentence based on the window of words/tokens surrounding them. They used word embeddings obtained by pre-training a word2vec model (Mikolov et al., 2013) on unlabeled contract data. Their work examined three data augmentation methods namely; interpolation, extrapolation and random noise. The augmentation methods manipulated the word embeddings to obtain a new set of sentence representations. The work by Papadaki (Papadaki, 2017) also highlighted that interpolation method performed comparatively better than the other methods like extrapolation and random noise. The work by Zhang and Yang (Zhang and Yang, 2018) explored various perturbation methods where they introduced random perturbations like Gaussian noise or Bernouli noise into the word embeddings in text related classification tasks such as sentence classification, sentiment classification and relation classification.

One of the recent works by Kobayashi (Kobayashi, 2018) trained a bi-directional language model conditioned on class labels where it predicted the probability of a word based on the surrounding context of the words. The words with best probability values were taken into consideration while generating the augmented sentences wherein the words in the sentences were replaced in a paradigmatic way.

However, to the best of our knowledge, there have been no work that specifically focuses on studying the different text data augmentation techniques on Relation Classification task in free text so far.

3 AUGMENTATION METHODS

In this Section, we describe various types of augmentation techniques that have been used in our experiments. As discussed briefly in Section 1, it is very challenging to create manually annotated datasets due to which we only have a few publicly available datasets with acceptable number of training and test data. Due to the constraints of cost and effort, most of the manually annotated datasets have less number of samples and it becomes difficult to optimally train deep learning models with the limited amount of data.

The use of distant supervision methods Mintz et al., (Mintz et al., 2009) to annotate data is able to compensate for the lack of quantity but is often susceptible to inclusion of noise when generating the dataset. This in turn constrains the performance of training models while performing relation classification. We consider applying augmentations on a manually augmented dataset as one of the ways to workaround the problem.

In our experiments, the training data was augmented at two levels; the text level and the word embedding level. Word Similarity and Synonym methods were used to generate new texts whereas interpolation and extrapolation methods (Papadaki, 2017) were used to generate embedding level augmentations. In order to apply the augmentation techniques, we tagged the sentences using NLTK (Bird et al., 2009) POS tagger. We restricted the augmentations only to nouns, adjectives and adverbs in each sentence. It was observed that by applying the restriction, we were in a better position to preserve the grammatical and semantic structures of the original sentences as compared to randomly replacing the words. GloVe word vectors (Pennington et al., 2014), which are a collection of pre-trained word vectors were used as word embeddings for words in our experiments.

We describe each of the augmentation methods in the following subsections.

3.1 Similar Words Method

This method by Wang and Yang (Wang and Yang, 2015) of text data augmentation works by exploiting the availability of similar words in the word embedding space. We replaced words with their respec-

tive top scored similar words to generate new sentences. An example input sentence and the resulting augmented sentence are given below:

- *Input Sentence* : The winner was $< e1 >$ Marilyn Churley $< /e1 >$ of the $< e2 >$ Ontario $< /e2 >$ New Democratic Party .
- *Augmented Sentence* : The winners was $< e1 >$ Monroe Churley $< /e1 >$ of the $< e2 >$ Manitoba $< /e2 >$ York Democrat parties

Here "Ontario" is replaced with "Manitoba", "New Democratic Party" with "York Democrat parties" among others.

3.2 Synonym Method

We followed the work by Mueller and Thyagarajan (Mueller and Thyagarajan, 2016) where they randomly replaced words with their synonyms obtained from WordNet Synonym Dictionary (Miller, 1995). We restricted the words to nouns, adjectives and adverbs and replaced them with their respective top scored synonym words to generate new sentences. A sample input sentence and the resulting augmented sentence are shown below:

- *Input Sentence* : Dominguez was not hotheaded said $< e1 >$ Woods $< /e1 >$ a former $< e2 >$ Arizona $< /e2 >$ attorney general.
- *Augmented Sentence* : Dominguez was not hotheaded said $< e1 >$ Wood $< /e1 >$ a former $< /e2 >$ Arizona $< /e2 >$ lawyer general.

Here "Woods" is replaced with "Wood", "attorney" with "lawyer" among others.

3.3 Interpolation Method

As described in work by Papadaki (Papadaki, 2017), we first obtained the top three nearest neighbours of a word in the embedding space. Then, we found the centroid from the embedding vectors of the nearest neighbours.

We calculated the new word embedding vector from the centroid and the original word embedding using the formula below:

$$w_j^{'} = (w_k - w_j)\lambda + w_j \quad (1)$$

In equation 1, $w_j^{'}$ denotes the new word embedding, w_k denotes the centroid, w_j denotes the original word embedding and λ is a parameter within the range of [0-1]. It controls the degree of interpolation. The new word embedding vector tends to move away from the original word embedding in the direction towards the centroid based on the λ value. The λ value that was used in our experiments is 0.25. Figure 1 depicts a graphical representation of the procedure.

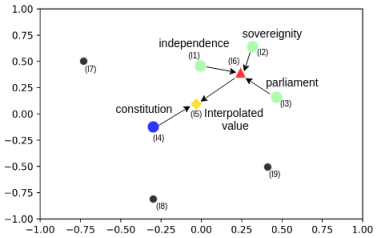

Figure 1: Interpolation between the centroid and word embedding to create a new embedding vector. The green points (l1,l2 and l3) represent the selected nearest neighbors, red indicator mark (l6) is the centroid obtained from the nearest neighbors, blue point (l4) depicts the original embedding vector and the yellow indicator mark (l5) depicts the resulting new word embedding vector. The black points represent other word embeddings present in the embedding space.

After obtaining the new word embedding vectors for the words in a sentence, we replaced them in place of their original embeddings and generated a new word embedding list for the sentence.

3.4 Extrapolation Method

Similar to the interpolation method (subsection 3.3), we calculated the new word embedding vector from the centroid and the original word embedding for a word using the formula given below (Papadaki, 2017):

$$w_j^{'} = (w_j - w_k)\lambda + w_j \quad (2)$$

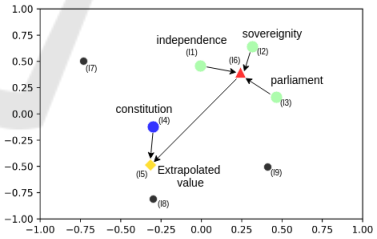

Figure 2: Extrapolation between the centroid and word embedding to create a new embedding vector. The labels are the same as figure 1.

In equation 2, $w_j^{'}$ denotes the new word embedding, w_k denotes the centroid, w_j denotes the old word embedding and λ is a parameter with value in the range [0-∞] that controls the degree of extrapolation. The new word embedding vector tends to move away from the original word embedding in the direction opposite to the centroid constrained by the λ value. The λ value that was used in our experiments is 0.25 for SemEval2010 and 0.5 for KBP37. Figure 2 depicts a graphical representation of the procedure.

3.5 Random Noise Method

As described in the work by Papadaki (Papadaki, 2017), in this method, instead of generating new augmented sentences we inserted perturbations to the word embeddings in the embedding layer of our model. For each word embeddings we added a Gaussian noise value to them as per the equation below:

$$w_j^{'} = w_j + x_j \tag{3}$$

where, w_j is the original word embedding, $w_j^{'}$ is the resulting word embedding and x_j is the random noise value. The values for each vector element of noise was sampled from the truncated normal distribution with $\mu = 0$, $\sigma = 1$ and with range between 0 and 0.3. Later we randomly selected the vector elements with a probability of 0.3 and rest of the vector elements were set to zero. The resulting vector is considered as noise vector. The values were kept in a small range so as to keep the resulting word embedding from moving too far away from the contextual word embedding space. One exception in this method to that of the other augmentation methods is that, we inserted perturbations in all the words regardless of its POS tag.

We have tried to explore the strengths and weaknesses of these methods in the experiments.

4 DEEP LEARNING METHODS

We have used two deep learning models namely; CNN and Attention-Based Bidirectional LSTM, for our experiments. Both the models have been briefly introduced in the subsections below.

4.1 CNN

Convolutional Neural Networks (CNN) were first implemented by (LeCun et al., 1998) on MNIST dataset. The basic principle behind CNN is to apply convolutions using multi-dimensional filters over an image as a sliding window to extract feature maps. CNNs have proved to be very efficient in the field of image processing as they are able to exploit the spatial structure of pixel intensities in an image. Recently, it has been observed that CNNs have also been able to extract lexical and sentence level features. We decided to use the text CNN model proposed by Zeng et al., (Zeng et al., 2014) which is widely used for relation classification tasks.

4.2 Attention-based Bidirectional LSTM

Recurrent Neural Networks (RNNs) are frequently used in the NLP domain. RNNs capture the sequential word patterns since they are inherently able to remember the previous states. RNNs are good at modelling sequential data (Boden, 2002). RNNs suffer from vanishing gradient problem. Long Short Term Memory (LSTM), which is a subset of RNN, is used to overcome the vanishing gradient problem by making use of a gated approach. One of the recent variants of LSTM for language specific tasks is Attention based LSTMs. They are able to capture the key part of the sentence by assigning higher weights to the relevant parts of an input sentence. Attention mechanism in neural networks have shown success in solving different tasks, like question answering (Hermann et al., 2015) and machine translation (Bahdanau et al., 2014). The attention mechanism in a model allows us to capture the most important semantic information in a sentence. In our experiments we are using the Attention-Based Bidirectional LSTM model proposed by Peng Zhou and Xu (Peng Zhou and Xu, 2016).

5 DATASETS

We used two datasets to run the augmentation experiments. The first dataset is **SemEval2010** (Hendrickx et al., 2009) which has a total of 10 unique relation classes (ignoring the direction aspect). It is a manually annotated public dataset which is generally accepted as a benchmark for relation classification tasks. The second dataset is the **KBP37** dataset which was created by Dongxu and Dong (Dongxu and Dong, 2016) and has a total of 19 unique relation types (ignoring direction aspect). We decided to use this (KPB37) dataset because it has more relation types and is more realistic than the SemEval2010 dataset (Dongxu and Dong, 2016). We wanted to experiment how different augmentation techniques worked on these datasets.

Each relation type (except *'other'* and *'no_relation'*) is further split into two relation classes for both the datasets, which considers the two directions w.r.t the positioning of the entities in a sentence. As an example, the relation *'org:subsidiaries'* is further split into *'org:subsidiaries(e1,e2)'* and *'org:subsidiaries(e2,e1)'* based on the location of the two entities e1 and e2 in a sentence. A few statistical details about both the datasets are provided in table 1.

Table 1: Details about SemEval2010 and KBP37 datasets.

Dataset: SemEval2010	No. of sentences in train: 8000 No. of sentences in test: 2717 No. of relation types: 19 No. of relation classes: 10	
	Classes: Cause-Effect	Instrument-Agency
	Product-Producer	Content-Container
	Entity-Origin	Entity-Destination
	Component-Whole	Member-Collection
	Communication-Topic	Other
Dataset: KBP37	No. of sentences in train: 15917 No. of sentences in test: 3405 No. of relation types: 37 No. of relation classes: 19	
	Classes: per:alternate_names	org:alternate_names
	per:origin	org:subsidiaries
	per:spouse	org:top_members/employees
	per:title	org:founded
	per:employee_of	org:founded_by
	per:countries_of_residence	org:country_of_headquarters
	per:stateorprovinces_of_residence	org:stateorprovince_of_headquarters
	per:cities_of_residence	org:city_of_headquarters
	per:country_of_birth	org:members
	no_relation	

6 EXPERIMENTS

As mentioned in Section 4, we implemented the text CNN model proposed by (Zeng et al., 2014) and attention-based BLSTM model proposed by Zeng et al., (Zhou et al., 2016). These two models were considered as the benchmark models for our experiments. We used PF (Position Features) method as introduced by (Zeng et al., 2014), where we specify the distance of each word in a sentence to the two entities (nominals) and feed them to the CNN and the attention-based BLSTM models along with the input sentences. For instance, in the sentence *"The winner was* $<e1>$ *Marilyn Churley* $</e1>$ *of the* $<e2>$ *Ontario* $</e2>$ *New Democratic Party."*, words between $<e1></e1>, <e2></e2>$ denote the two entities w_1, w_2 respectively. We find the distance of the current word to w_1 and w_2 and use the resulting distance vectors d_1 and d_2 in the models.

For each dataset, we randomly extracted 75% samples from the train data and used them as a reduced dataset (which we refer to as 75% dataset in the rest of the paper) for the experiments. While re-sizing the datasets, we took care to maintain the "data to class ratio" of the original datasets by selecting a proportional number of samples from each of the classes. Augmentation methods were applied on each of the datasets, namely the 100% and the 75% datasets. While generating augmented data, the same PF and class labels were used as that of the source sentences. The resulting augmented data was appended to the original data in order to generate new train data. For example, on the 75% KBP37 train data, after running the synonym method we obtain a new list of sentences. This list was then appended to the original 75% train data and the resulting list was used as a training file for our models.

As mentioned in Section 3, we only considered nouns, adjectives and adverbs for data augmentations. It came to our notice that in many cases, replacing verbs/prepositions with similar words/synonyms resulted in making the sentences ungrammatical and also in some cases changed the contextual meaning of the sentence. The resulting sentences were no longer a logical fit for their respective relation classes. However, replacing nouns, adjectives and adverbs retained the meaning even though at times it was not 100% grammatical. We were able to retain the context and generate augmented data by restricting the replacements to the above mentioned POS tags for each sentence in the datasets.

When training the models, we first found out the optimal hyper-parameters for each of the datasets (original, and 75%) for both KBP37 and SemEval2010 and used them as baseline parameters. Since not every hyper-parameter detail and implementation was available for the considered models (section 4), we implemented and optimized the models. Once we obtained an accuracy closer to the benchmark after several runs, we froze the hyper-parameters. Then, for each dataset we applied the

Table 2: F1 scores obtained for KBP37 dataset.

Deep Learning Methods	Experiments	F1 scores and Train Data Size			
		100% Training Samples		75% Training Samples	
		F1 score	No. of Training Samples	F1 score	No. of Training Samples
CNN	Baseline	50.74	15917(-)	49.22	11922(-)
	Similar Words	50.86 ↑	31834(+)	49.45 ↑	23844(+)
	Synonym	51.68 ↑	31834(+)	49.28 ↑	23844(+)
	Interpolation	50.89 ↑	31834(+)	50.25 ↑	23844(+)
	Extrapolation	50.36 ↓	31834(+)	48.75 ↓	23844(+)
	Random Noise	50.45 ↓	31834(+)	49.90 ↑	23844(+)
att-BLSTM	Baseline	51.84	15917(-)	49.05	11922(-)
	Similar Words	51.84 =	31834(+)	50.40 ↑	23844(+)
	Synonym	51.60 ↓	31834(+)	49.84 ↑	23844(+)
	Interpolation	50.01 ↓	31834(+)	49.49 ↑	23844(+)
	Extrapolation	51.51 ↓	31834(+)	49.87 ↑	23844(+)
	Random Noise	51.19 ↓	31834(+)	49.63 ↑	23844(+)

(-) denotes the original number of training samples. (+) denotes that augmented data has been added to the original training samples. (=) denotes same F1 score with respect to the corresponding baseline score in the same column. (↑) denotes an increase in F1 score with respect to the corresponding baseline score in the same column. (↓) denotes a decrease in F1 score with respect to the corresponding baseline score in the same column.

augmentation methods and generated the corresponding new augmented data. In order to verify the effect of augmentations on the results, we used the same hyper-parameters and ran the models with the new augmented train data. We did not do any cross validations during training. We used the pre-trained, 300-dimensional GloVe word vectors for all the experiments. The embedding layer was kept as *non-trainable* in all the models while training. Test data was kept separate and no augmentations or data resizing was done on it. For all the trained models we used the same test data, i.e, for all experiments run on KBP37 dataset, we used the test set provided for KBP37 and the same was done for SemEval2010.

7 EXPERIMENTAL RESULTS

Table 2 shows the F1 scores for the experiments conducted on KBP37 dataset. The results are split for 100% training samples and 75% training samples. For testing, we used the same test set for all the trained models which had 3405 sample sentences as provided in the KBP37 dataset (1).

Table 3 shows the F1 scores for the experiments conducted on SemEval2010 dataset. As described above for KBP37 dataset, the results are split for 100% training samples and 75% training samples and we also mention the number of training samples used in each of the experiments. For testing, we used the same test set for all the trained models which had 2717 sample sentences as provided in SemEval2010 dataset (1).

For KBP37 dataset, we take the results obtained by Dongxu and Dong (Dongxu and Dong, 2016) for PF(Position Feature) experiments as the state of the art, where they obtained an F1-score of 51.3% for CNN model and 54.3% for RNN model. Similarly, for SemEval2010 dataset we consider F1 score of 78.9% obtained by Zeng et al., (Zeng et al., 2014) for CNN and F1 score of 84.0% obtained by Zhou et al., (Zhou et al., 2016) for att-BLSTM as state of the art results. The best results obtained after multiple runs have been mentioned under baseline results. Since the embedding layers were kept as non-trainable in all the models, a slight dip is observed in the F1 scores with respect to the state of the art results considered.

A decrease of up to 3% is observed between the baseline F1 scores for 75% data samples and the F1 scores for 100% data samples. The F1 scores for baseline and the F1 scores obtained from models trained on augmented data vary by approximately $\pm 2\%$. We observe an increase in the F1 scores for all deep learning models across both the datasets, when training on augmented data generated with synonym method over 75% training samples.

One key observation is that for augmentation experiments performed over 75% dataset, there is at least one augmentation method where we are able to achieve an F1 score similar to that of F1 score for baseline (100% data). This holds true for both the SemEval2010 and KBP37 datasets. For instance, we can consider the F1 score for baseline (100% data) for KBP37 dataset. For CNN model we get an F1 score of 50.74%. For interpolation experiment performed on 75% dataset, we observe an F1 score of

Table 3: F1 scores obtained for SemEval2010 dataset.

Deep Learning Methods	Experiments	F1 scores and Train Data Size			
		100% Training Samples		75% Training Samples	
		F1 score	No. of Training Samples	F1 score	No. of Training Samples
CNN	Baseline	79.51	8000(-)	77.31	5994(-)
	Similar Words	78.35 ↓	16000(+)	77.72 ↑	11988(+)
	Synonym	77.97 ↓	16000(+)	77.64 ↑	11988(+)
	Interpolation	79.84 ↑	16000(+)	76.70 ↓	11988(+)
	Extrapolation	79.52 ↑	16000(+)	78.08 ↑	11988(+)
	Random Noise	76.54 ↓	16000(+)	76.56 ↓	11988(+)
att-BLSTM	Baseline	79.89	8000(-)	78.00	5994(-)
	Similar Words	79.41 ↓	16000(+)	78.03 ↑	11988(+)
	Synonym	79.04 ↓	16000(+)	78.42 ↑	11988(+)
	Interpolation	78.40 ↓	16000(+)	75.78 ↓	11988(+)
	Extrapolation	78.08 ↓	16000(+)	76.59 ↓	11988(+)
	Random Noise	78.79 ↓	16000(+)	78.69 ↑	11988(+)

(-) denotes the original number of training samples. (+) denotes that augmented data has been added to the original training samples. (↑) denotes an increase in F1 score with respect to the corresponding baseline score in the same column. (↓) denotes a decrease in F1 score with respect to the corresponding baseline score in the same column.

50.25% which is very close to the 100% baseline result of 50.74%. The same can also be observed for att-BLSTM model on KPB37 dataset, where we get a 100% baseline F1 score of 51.84% and 50.40% for Similar Words method performed on 75% train data. All such results have been highlighted in blue in both the results tables (3, 2).

With these observations, we extrapolate that adding meaningful augmentations which preserve the grammatical and the semantic structures of the original sentences enable us to obtain results similar to that of the original dataset even with less data. However, one limitation in the augmentation techniques used is the lack of ability to introduce new variations with respect to the positions of entities in the original sentences. For example, for an input Sentence: *"Dominguez was not hotheaded said* $< e1 >$ *Woods* $< /e1 >$ *a former* $< e2 >$ *Arizona* $< /e2 >$ *attorney general."*, on applying synonym method we obtain, *"Dominguez was not hotheaded said* $< e1 >$ *Wood* $< /e1 >$ *a former* $< /e2 >$ *Arizona* $< /e2 >$ *lawyer general."*. It can be clearly seen that even though we are able to generate a new sentence which preserves the grammatical and semantic structure of the original sentence, we do not change the positions of the entities denoted by $< e1 >, < /e1 >$ and $< e2 >< /e2 >$ respectively. The position features contribute significantly to the learning while training the models. A lack of new entity ordering results in an almost saturated F1 score for augmentation experiments run on 100% data. Additionally, the interpolation and extrapolation methods may perform better if we used word vectors that are more relevant to these datasets thereby capturing their data distributions, instead of using pre trained GloVe word vectors. We have not explored all the variations of hyper-parameter values for each augmentation method. Further tuning of these hyper-parameters may yield better results.

8 CONCLUSION

In this paper, we investigated the available techniques for text data augmentations with respect to Relation Classification in free text. As per our knowledge, there have been no work that studies different text data augmentation techniques on relation classification for free text over different Deep Learning Models as yet. We implemented five text data augmentation techniques and explored the ways in which we could preserve the grammatical and the contextual structures of the sentences while generating new sentences automatically using data augmentation techniques. We discussed the importance of manually annotated data and why it is crucial to build new methods to augment data on smaller manually annotated datasets. The experiments are carried on one subsampled dataset (75%) and it will be interesting to observe the effects of augmentations in even more detail on further subsampled datasets.

From our experiments, we observed that we are able to mimic the performance of an original dataset by adding augmented data to a small dataset. We also highlighted the inability of the existing text data augmentation techniques to introduce new features for learning. This is a limitation which bars automatic data generation from being as diverse and contextual as a manually annotated dataset with real

texts. Hence, it is crucial to develop new augmentation methods which can introduce diversity and at the same time retain the grammar and context in a sentence. As a future work, one can experiment different combinations of these methods as an hybrid approach and see the possible improvement of accuracy for different NLP related tasks and also generate synthetically similar sentences using Generative Adversarial Networks (GANs) which are widely used in image domain to obtain synthetic data.

REFERENCES

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

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python*. OReilly Media Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, 1st edition.

Boden, M. (2002). A guide to recurrent neural networks and backpropagation. In *The Dallas Project, SICS Technical Report T2002:03*.

Chalkidis, I., Androutsopoulos, I., and Michos, A. (2017). Extracting contract elements. In *Proceedings of the 16th edition of the International Conference on Artificial Intelligence and Law*, pages 19–28. ACM.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Dongxu, Z. and Dong, W. (2016). Relation classification: Cnn or rnn? In *Natural Language Understanding and Intelligent Applications*, pages 665–675, Cham. Springer International Publishing.

Hendrickx, I., Kim, S. N., Kozareva, Z., Nakov, P., Ó Séaghdha, D., Padó, S., Pennacchiotti, M., Romano, L., and Szpakowicz, S. (2009). Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 94–99. Association for Computational Linguistics.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1684–1692.

Kobayashi, S. (2018). Contextual augmentation: Data augmentation by words with paradigmatic relations. In *NAACL-HLT*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. In *Gradient-based learning applied to document recognition.Proceedings of the IEEE, 86(11)*, pages 2278–2324. IEEE.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, pages 3111–3119. Curran Associates Inc.

Miller, G. A. (1995). Wordnet: A lexical database for english. In *Communications of the ACM Vol. 38. No. 11*, pages 39–41. ACM.

Mintz, M., Bills, S., Snow, R., and Jurafsky, D. (2009). Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011. Association for Computational Linguistics.

Mueller, J. and Thyagarajan, A. (2016). Siamese recurrent architectures for learning sentence similarity. In *AAAI*, pages 2786–2792.

Papadaki, M. (2017). *Data Augmentation Techniques for Legal Text Analytics*. Department of Computer Science, Athens University of Economics and Business, Athens.

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

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

Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J. (2014). Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*, pages 2335–2344.

Zhang, D. and Yang, Z. (2018). Word embedding perturbation for sentence classification. *arXiv preprint arXiv:1804.08166*.

Zhou, P., Shi, W., Tian, J., Qi, Z., Li, B., Hao, H., and Xu, B. (2016). Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 207–212.