Improving short text classification through global augmentation methods

Vukosi Marivate[1,2][0000−0002−6731−6267] and Tshephisho Sefara[2][0000−0002−5197−7802]

[1] University of Pretoria, South Africa
[2] Council for Scientific and Industrial Research, South Africa
vukosi.marivate@cs.up.ac.za
tsefara@csir.co.za

Abstract. We study the effect of different approaches to text augmentation. To do this we use 3 datasets that include social media and formal text in the form of news articles. Our goal is to provide insights for practitioners and researchers on making choices for augmentation for classification use cases. We observe that Word2vec-based augmentation is a viable option when one does not have access to a formal synonym model (like WordNet-based augmentation). The use of *mixup* further improves performance of all text based augmentations and reduces the effects of overfitting on a tested deep learning model. Round-trip translation with a translation service proves to be harder to use due to cost and as such is less accessible for both normal and low resource use-cases.

Keywords: natural language processing · data augmentation · deep neural networks · text classification

1 Introduction

In this paper, we look at data augmentation for Natural Language Processing (NLP) applications. Encouraged by the ever-present use of data augmentation in computer vision applications [4], we want to be able to provide researchers and practitioners with a better understanding of augementation for NLP tasks. Augmentation has benefited many image classification tasks [4], the structure of images are changed in order to increase the number of samples available to the machine learning algorithm while introducing resilience in the final model. Augmentation of text data to be able to create robust models has had different factors of success. Some approaches require more direct information about the language at hand than others [19], while others are more agnostic given a learned language model that can be used [16].

Over the last few years, the development of distributed word representations (word embeddings) [20] has improved the modelling of semantic relations between words in a text. This has created many new approaches to understanding text, in the case of this work, text classification tasks. In order to improve the classification accuracy, as well as make models more robust, one can look

at data augmentation as a way to improve performance and create robustness. More recently, the development of unsupervised language models [24,13], makes it possible for us to use more data-driven language models that can be combined with data augmentation methods to improve performance and robustness of machine learning models for NLP.

We are motivated by a number of factors. We would like to be able to train classification models that do not necessarily have a large amount of labelled data. Labelling of data comes at a cost. Whether it is for identifying fake news [17], understanding political phenomena [27], feedback on government services [29], or better coordination during emergencies [14], getting labels is always challenging. As such, knowing that building machine learning models requires a large amount of data, we need to still be able to build models with smaller data. A large organisation might have access to large data sets as well as resources to label a large chunk of it. A smaller organisation tends to not have large data and fewer resources to label. To extend the use of the classifiers further than the distribution of information fed into it, we need to be able to change the input data in a way that makes the final learned model more robust to slight changes in the input distribution. This could be caused by the evolution of language or even geographical changes. Another use is in semi-supervised learning, where we use the few labels we have to create a classifier (that is likely noisy) to label more unlabelled data and then feed this back to train another classifier.

Our contributions, in this paper, is a short survey of a number of data augmentation methods, specifically looking at methods that augment data with more of a global view. That is, the schemes replace words that are used similarly from a global view instead of a contextually local view. So how are similar words used across texts, instead of what might be the best word to replace in this specific sentence in this specific document. We discuss methods that use linguistic features, a model that uses a translation service and then augmentation methods that act on embeddings/language models. To better understand the behaviour of the augmentation methods, we evaluate the approaches on a number of classification datasets under a number of conditions and provide insights into the different approaches. We also show the effect of the *mixup* [34] method on NLP tasks as an augmentation approach. This paper is organised as follows; We cover different text augmentation methods first. The approach in our comparative study is described in Section 3. Section 4 discusses the experimental results and then we conclude in Section 5.

2 Augmentation methods for text

For many machine learning tasks, data augmentation has been employed as a regularisation method while training supervised machine learning models. The more diverse examples fed to the model during training, the better the model generalises, and consequently, the better they predict when presented with new examples. Data augmentation is famously used in images, audio and more recently in text. In this section, we describe prior approaches to text augmentation.

For our work, we categorise text augmentation techniques into two categories. Namely, *augmentation on text source* and *augmentation on text representation*. In the rest of this section, we discuss methods that fall in either of these categories. A summary of the methods we discuss in this section is shown in Table 1 and Fig. 1 illustrate how these methods augment data.

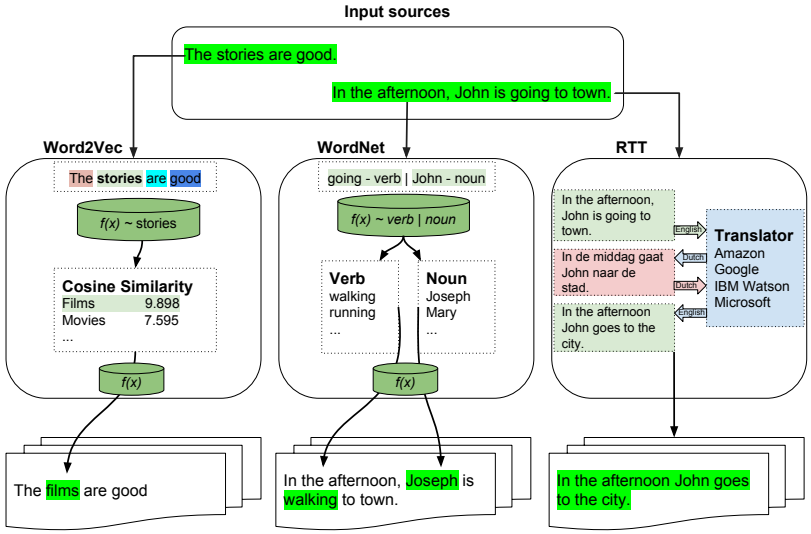


Fig. 1. Overview of augmentation structures

Table 1. Augmentation techniques

Method	Augmentation on	Labels	Linguistics	Semantic
Synonyms from WordNet [35]	*text source*	No	Yes	Yes
Words from Word2vec [16]	*text source*	No	Yes	Yes
Round-trip Translation [18,30]	*text source*	No	Yes	Yes
mixup [34]	*text representation*	Yes	No	No

2.1 Augmentation on source text

Augmenting textual data requires language modelling rules, written by linguistic experts, which are limited for low-resource languages. The alternative way is to use a dictionary/thesaurus or database of synonyms to make word replacements. In a situation where there is no dictionary, the other way is to use distributed

word representation where semantically similar words are identified together. In this subsection, we present how word replacements and sentence paraphrasing can be used as augmentation techniques.

Synonym Augmentation We first begin with methods that use linguistic features. The best way to augment data is by using human rephrasing of sentences, but this is expensive. Therefore, the most natural option in data augmentation for most authors is to replace words or phrases with their synonyms. Verbs and nouns are the best name classes to have synonyms in various context. The popular open-source lexical database for the English language is WordNet [21]. It groups words like nouns, verbs, adjectives and adverbs into sets of cognitive synonyms called *synsets* each expressing different concept, provides short definitions and usage examples, and records a number of relations among these synonym sets. WordNet thus superficially resembles a thesaurus in that it groups words together based on their meanings. However, with the distinction that WordNet labels the semantic relations among words and it interlinks word forms and strings of letters specific to senses of words resulting with words found in close proximity to one another in the network semantically disambiguated. A good illustration of synonym augmentation is shown in Fig. 1 as WordNet.

The thesaurus-based augmentation method has been applied to training a Siamese adaptation of the Long Short-Term Memory (LSTM) network to assess the semantic similarity between sentences [22]. Zhang *et al.* [35] used a WordNet-based augmentation method to augment their training data by using a Geometric function to help select words from a given dataset, using the selected words to find their synonyms to train a temporal convolutional network that learns text understanding from character level input up to abstract text concepts.

Semantic Similarity Augmentation Using distributed word representation (word embeddings) [20], one can identify semantically similar words [16]. This approach requires either pre-trained word embedding models for the language at hand or enough data from the target application to be able to build the embedding model. This approach thus does not require access to a dictionary or thesaurus for a language in order to find synonyms. This can benefit languages where such resources might be harder to obtain but there might be enough unsupervised text data to be able to build the embedding models. With recent advances in building complete language models [25,7,24,10], further advances can be made to identify syntactic and semantic similarity for word augmentation.

For this paper, we are not exploring the use of language models. One can view language models as enabling a more localised replacement of words given the local context in the sentence. On the other hand, word embeddings give a global context. Language Models such as that by [7] allow the filling in of blanks in any part of a sentence. While other language models can be more used to fill in the *next* missing word in a sentence[25], reading a sentence left to right then filling in the missing word at the end. This exploration remains an area of future work.

Round-trip Translation Round-trip translation (RTT) is also known as recursive, back-and-forth, and bi-directional translation. It is the process of translating a word, phrase or text into another language (forward translation), then translating the results back into the original language (back translation) [1]. RTT can be used as an augmentation technique to increase the training data. An example is shown in Fig. 1. Fadaee *et al.* [9] used a method called translation data augmentation that uses rare word substitution to augment data for low-resourced neural machine translation (NMT). Their method augments both the source and target sentence to create a new pair preserving the meaning of the sentence. They proposed a weaker notion of label preservation that allows alteration of sentence pairs as long as the sentences remain translations of each other.

Sennrich *et al.* [30] used back translation as a data augmentation method to leverage the target monolingual data. In their method, both the NMT model and training algorithm are kept unaltered, and they employed back translation to construct training data. That is, target monolingual sentences are translated with an existing machine translation system into source language, that is used as additional parallel data to retrain the source-to-target NMT model. Aroye hun and Gelbukh [2] used English RTT on four intermediate languages (French, Spanish, German, and Hindi) that increased the size of their training set by a factor of 5. This method improved the performance of their model for identification of aggression in social media posts. Although back-translation has been proven to be effective and robust [2], one major problem for further improvement is the quality of automatically generated training data from monolingual sentences. Some of the incorrect translations are very likely to decrease the performance of a source-to-target model due to the deficiency of a machine translation system.

2.2 Augmentation on representation

Augmentation on source text requires that we are able to have access to a method to replace words in the source text to create more examples while at the same time keeping the meaning of the full sentence the same. In this subsection, we present a method that was introduced as a regularisation technique on Deep Neural Networks but can be viewed as augmentation that instead acts on the text representation. We discuss *mixup* which acts both on the input and the outputs.

***mixup* Augmentation** *mixup*, introduced in [34] can be seen as an augmentation method that can be classified as *augmentation on representation*. *mixup* is data agnostic and is applied to images, speech and tabular data in the original paper [34]. *mixup* creates new training examples by drawing samples (sets of two or more) from the original data and combining them convexly. It combines the data both in terms of the input and output. The simplest implementation, which the authors suggest, creates a new augmented dataset by taking pairs of samples from the initial dataset and convexly summing both the input and output. That

is, given two examples $(X_1, y_1), (X_2, y_2)$, where X is the input vector and y is the one-hot encoded labels, we construct a new example:

$$\hat{X} = \lambda X_1 + (1 - \lambda) X_2$$

$$\hat{y} = \lambda y_1 + (1 - \lambda) y_2$$

where $\lambda \in [\, 0, 1]$ and is drawn from Beta distribution, $\lambda = \beta(\alpha, \alpha)$, α is set [34]. It is somewhat akin to soft-labelling but is best suited for learning algorithms that use the cross-entropy loss and also changes the input. The standard method is used for classification problems (it remains further work to explore the uses for regression). For text problems, augmentation by *mixup* can be done on the text representation. As such we can use *mixup* with bag-of-words models, TFIDF [26], word embeddings and language models.

3 Method and approach

Our goal is to measure the impact of augmentation methods on the performance of classification algorithms. We would like to compare methods across different datasets and on as level a playing field as possible. It is important to do this so that insights can be used by other researchers as they build tools for different classification problems in text. We first describe the data we will use for classification, then the learning algorithms we will be using, and finally the last subsection details the experiments we will perform to test the augmentation approaches to better understand their strengths and limitations.

3.1 Data

The data we will use is from three different datasets that represent different challenges with text. For short text, we use the Sentiment 140 data set [11] as well as the Hate Speech data set [6]. For longer text, we use the AG News data set used in [35]. The Sentiment 140 uses noisy labelling [11] in the form of weak supervision to label the training data. The Hate Speech dataset uses human annotated labels. Even though the Sentiment 140 has a separate test set that is made up of human annotated labels, we will first focus on splitting the original data. The data is summarised in Table 2.

Table 2. Summary of data

Data Source	Words	Approx. Sentences	Avg Number of Words	Labels
Sentiment 140	18M	2M	13	Binary
AG News	3M	154K	31	Categorical
Hate Speech	243K	24K	14	Categorical

3.2 Augmentation algorithms

We augment each data set using four types of augmentation methods: WordNet-based augmentation, Word2vec-based augmentation, Round-trip translation as well as *mixup*. The first three methods augment data on text and can be combined with *mixup* which augments data on the fly. As part of this paper, we also release a library that allows the use of all text-based augmentation methods[3]

WordNet-based synonym augmentation WordNet-based augmentation is a type of augmentation that randomly selects r words from a sentence (if they exist) using any distribution. To decide on which words to replace, we selected replaceable words like verbs, nouns, and the combination of them using a part-of-speech tagger implemented in the natural language toolkit (NLTK)[3] from a given text and randomly selects r of them. The probability of r is calculated by a Geometric distribution $P[r] \sim p^r$ where parameter p is the probability of success. The synonym s chosen given a word is also determined by another Geometric distribution in which $P[s] \sim p^s$ using the same probability of success $p = 0.5$[35]. The new sentence is constructed by replacing the selected verb or noun with their synonyms. The algorithm has options to choose to augment using either verbs or nouns or even a combination. We report the results choosing the outcome of p on the first trial.

Word2vec-based (learned semantic similarity) augmentation Word2vec is another robust augmentation method that uses a word embedding model [20] trained on the public dataset to find the most similar words for a given input word. We use both a pretrained Wikipedia Word2Vec model for formal text. For social media data, we convert a Glove model, pretrained on Twitter data, to Word2vec format using Gensim [28]. We load the converted models in our algorithm to augment data by randomly selecting a word in a sentence to determine its similar words using cosine similarity. To select a similar word, we use the cosine similarity as a relative weight to select a similar word that replaces the input word. Our algorithm is illustrated in Algorithm 1, it receives a string and an integer where the string is an input data and the integer represents a number of repetitions to augment a given input data. The advantage of Word2vec is that it tends to produce vectors that are more topically related, in other words it allows words with similar meaning to have similar representation.

Round Trip Translation (RTT) augmentation We implement RTT augmentation using Google translation services[4] as well as Amazon translate[5]. We translate text in English to a target language then back to English. To measure the effect of RTT on a different number of augmentations, we translated text to

[3] https://github.com/dsfsi/textaugment
[4] http://translate.google.com
[5] https://aws.amazon.com/translate/

Algorithm 1: Word2vec-based augmentation algorithm [32].

Input: s: a sentence, run: a number
Output: $\hat{s}$ a sentence with words replaced

```
def Augment(s,run):
    Let V⃗ be a vocabulary;
    for i in range(run) :
        w_i ← randomly select a word from s;
        w⃗ ← find similar words of w_i;
        s_0 ← randomly select a word from w⃗ given weights as distance;
        ŝ ←replace w_i with similar word s_0;
    return(ŝ);
```

French then back to English. We ensured that the paraphrased back-translated texts carry the same meaning as the source text. Due to the cost of doing the augmentation, we are unable to show results on our larger datasets. This is a challenge in using RTT.

***Mixup* augmentation** We implement this method with the α variable set to 0.2. We run *mixup* on its own as well as in combination with the other methods described.

Evaluation metrics For all of the experiments, we use error, and with some experiments loss, as metrics to give insight to the behaviour of the augmentation algorithms and their impact on model performance. We use the *error* for the rest of the paper defined as.

$$error = 1 - accuracy \tag{1}$$

We also present cross-entropy loss when investigating overfitting.

3.3 Learning Algorithms

We test two types of learning algorithms for the classification of text. Specifically, we use a deep learning approach as well as a logistic regression (LR). The inputs of the algorithms will be slightly different given the different types of augmentation. We would like to be able to show the impact of augmentation on the state of the art approaches as well as the impact on traditional algorithms and NLP representation.

We implement our models using Keras[6] (a deep learning toolkit). For logistic regression, the model contains a dense layer given few arguments; a number of labels, and activation function as *softmax*. We compile the model using an adaptive learning rate method for gradient descent called ADADELTA [33] as

[6] http://keras.io

our optimiser, and categorical cross entropy as our loss function. For the DNN, we use a Bidirectional LSTM network [12] coupled with 1 dimensional convolutional neural network and 3 fully connected layers activated by a *rectified linear unit* and *softmax* illustrated in Table. 3.

Table 3. Table of a DNN architecture. Dropout layers use probability of 0.4. Tanh represents a rectified linear unit. The number of filters for the last layer corresponds to the number of classes of the given dataset.

Layer	Type	Filters/Neurons	Kernel
1	Bidirectional LSTM+tanh	256	-
2	Dropout	-	-
3	Conv1D+sigmoid	512	3
4	GlobalMaxPool1D	512	-
5	Fully connected+tanh	512	-
6	Dropout	-	-
7	Fully connected+tanh	200	-
8	Dropout	-	-
9	Fully connected+softmax	4	-

4 Experiments and Results

This section explains detailed experiments on the effectiveness of augmentation in several settings. We conduct three types of experiments on the datasets. The first experiment tests the importance of augmentation on limited data. The second experiment tests the effectiveness of augmentation under overfitting settings. The last experiment tests how the number of augmentations affects performance.

4.1 Effect of augmentation on less data

Limited data is a challenge both in getting well-labelled data for supervised learning problems [8] and also continues to be a bigger challenge for low-resource languages [5]. Here we present what the effect of the different augmentation schemes are when labelled data is reduced.

Here we show the effect of augmentation on learning for logistic regression and Deep Neural Network (DNN) models. For logistic regression, we use a TFIDF vector scheme [26]. For TFIDF, we fit the vector model using only the training data. For the DNN, we use pre-trained word vector representations from Glove [23]. For Glove we use the pre-trained Wikipedia model for the AG News dataset and pre-trained Twitter model for the social media datasets. For all of the datasets in this experiment, we augment the data 5 times. We also include the original dataset, resulting in an augmented dataset that is 6 times the size of the original. To make results comparable, when we use no augmentation we just repeat the same original dataset 6 times. Due to limitations of RTT we were

only able to run experiments on logistic regression on AG News and on the Hate Speech Dataset. As we cannot compare RTT across all experiments, we discuss it at the end.

First looking at the AG News results (Figures 2a and 2b), we start noticing a few trends. We use 10000 articles for training and a further 10000 for validation. We repeat the experiment 5 times. When looking at the validation error for logistic regression on the AG News dataset, Fig. 2a, we note that augmented data leads to better results. For the logistic regression results (Fig 2a), the WordNet-based Synonym and W2V augmentation lead to lower validation error. At the same time, across all augmentation schemes, *mixup* improved performance. With the DNN, WordNet-based Synonym augmentation with *mixup* leads to the lowest validation error. We believe this is due to the fact that the AG News dataset is made up of news articles that have longer length and are written formally, as such a synonym database is likely to make good substitutions.

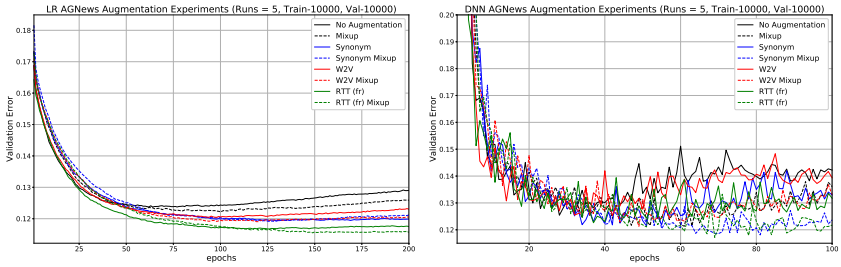
Turning our attention to the two social media based datasets, we now have to deal with more informal language. In the Sentiment 140 dataset experiments (Figures 3a and 3b), we use 100000 posts for training and 10000 for validation. Again augmentation outperforms no augmentation. We are not able to get results for *mixup* on Logistic regression with TFIDF due to computational constraints. *mixup* still provides better performance, when combined with another augmentation method for the DNN, and we expect that this is the same for LR. *mixup* also affects the performance of the learning even when it is starting to overfit. We discuss more details of this later.

Finally for the Hate Speech dataset, we use the smallest amount of training data. We have a training set of size 2000 and a validation set of the same size. For the logistic regression (Fig 4a), the *mixup* augmentation provides good performance. The same result is seen for the DNN (Fig 4b).

RTT Augmentation proved to have differing effects. On the logistic regression, its performance on AG News was the best while on the Hate Speech dataset it was less successful. On the DNN, we observe similar results for both AG News and the Hate Speech data sets. We do think that this does likely indicate that on social media data, RTT will have a hard time doing translations and as such will increase error. At the same time, the more formal AG News dataset benefits from the augmentation. More still needs to be done to explore this area. In the results we present for RTT, for AG News we used Amazon Translate with *English to French and back to English* as well as *English to German and back to English*. For the Hate Speech dataset we used Google translate for *English to French and back to English*.

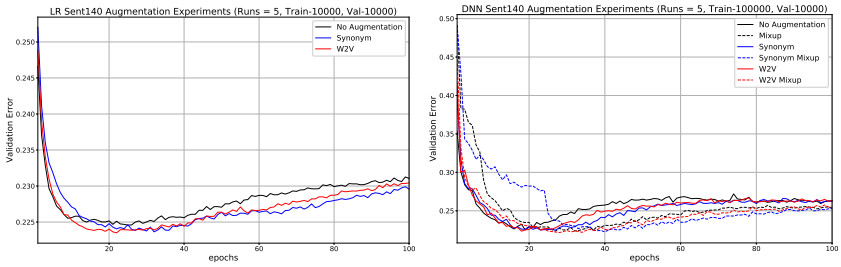
4.2 Effect of different number of augmentations

Multiple augmentations may have an impact on the error. The next subsections discuss how multiple augmentations impact the error. We focus on using LR to conduct the experiments. We augment the data in portions of different sizes. For AG News, we augment 1000 and 10000 tokens for 5 and 10 times and report on the error using WordNet-based augmentation. We obtain the error of 0.1824



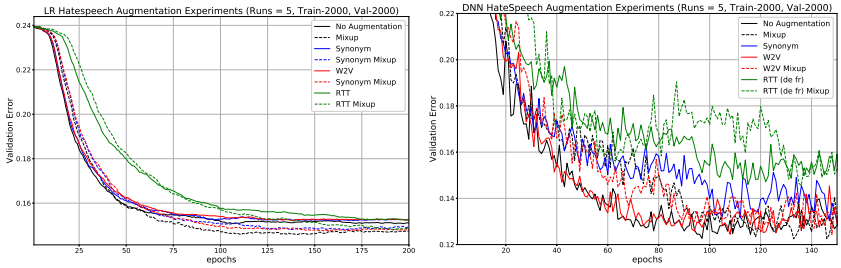
(a) AG News - LR error (b) AG News - DNN Error

Fig. 2. Effect of augmentation on different training set sizes for AG News



(a) Sentiment 140 - LR error (b) Sentiment 140 - DNN Error

Fig. 3. Effect of augmentation on different training set sizes for Sentiment 140



(a) Hate Speech - LR error (b) Hate Speech - DNN Error

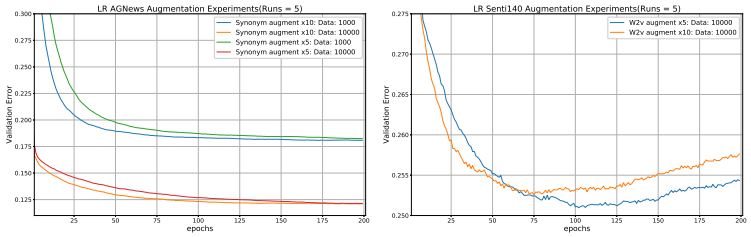
Fig. 4. Effect of augmentation on different training set sizes for Hate Speech (Synonym Mixup ommited for readability)

when augmenting 1000 tokens for 5 times. The error reduced to 0.1809 when we increase the number of augmentations to 10. This shows the effect of increasing number of augmentation on error is very low. As such, we increase the data size and augment 10k of the data. We observe an error of 0.12136 when augmenting 5 times. Then, we increase the number of augmentations to 10 and the error

slightly reduce to 0.121. With these results for longer text, we observe that number of augmentations has an impact on the error and when augmenting larger data the error is slightly reduced by a difference of 0.12136-0.1824.

For Sentiment 140, we augment 10000 tokens using the same settings. We observe an error of 0.25434 when augmenting for 5 times, we increased the number of augmentations to 10 then the error increased to 0.25764, this shows Word2vec introduced more noise on the data.

What we observe is that moving from 5 to 10 augmentations had only slight effects on lowest error for the cases we studied.



(a) AG News - LR error (b) Sentiment 140 - LR error

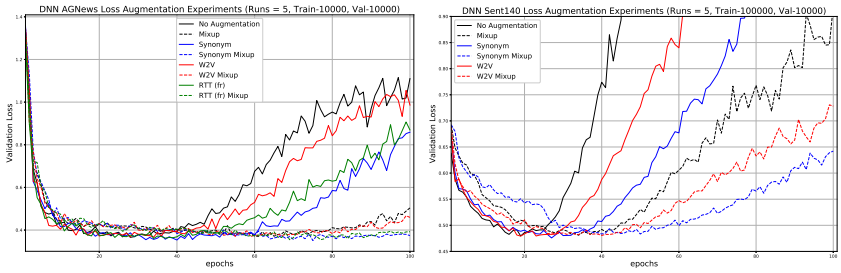
Fig. 5. Effects of different sizes of augmentation given different input data on error. (a) WordNet-based augmentation slightly reduces the error with more number of augmentations. (b) Word2vec increase the error by introducing more noise.

4.3 Effect of augmentation on overfitting

Augmentation is viewed as a form of regularisation [31]. We can look at the effect of the different augmentation methods on how they reduce the effects of overfitting. Figure 6 shows the effect of the different methods we tested on overfitting. On both the AG News dataset (Fig 6a) and the Sentiment 140 dataset (Fig 6b) augmentation reduces the overfitting. *mixup* has the largest impact. On AG News, overfitting effect is reduced by Synonym, W2V and RTT augmentations, even without *mixup*.

5 Conclusion and Future Work

The use of WordNet-based Synonym augmentation on AG News or long text does result in most of the words found from the database. Hence, the probability of those words to be augmented is higher and the resulting augmented data does not change the meaning of the message. And this, results in WordNet-based augmentation approach being a rich augmentation compared to RTT and Word2vec-based approach. The WordNet-based augmentation approach highly



(a) AG News - DNN Loss (b) Sentiment 140 - DNN Loss

Fig. 6. Effect of overfitting on DNN model shown through cross-entropy loss for the AG News and Sentiment 140 datasets

depends on an existing database. Hence, the disadvantage is when such database in not available for low-resourced languages, it first needs to be created at a great cost. The alternative way of augmenting low-resource language, is to use unsupervised word embedding models that can be trained using Glove, Word2vec, and fastText [15] on pre-collected available corpuses such as Wikipedia, Newspapers or literature. The vector representations in the models can be used to identify the nearest neighbours via the cosine similarity (such as used in the the Word2vec based augmentation). Such an approach becomes more feasible for augmenting data from lower resourced languages. As shown in the paper experiments Word2vec-based augmentation resulted with good comparable results compared to synonym-based approach on Sentiment 140, this shows that augmentation can be done with only Word2vec. Even on AG News, Word2vec-based augmentation is competitive.

RTT-based augmentation is expensive in many ways. If using an online service, the commercial services available require financial resources. The free tiers made available will only be able to translate a few thousand words for free. If one wants to reduce the cost, then one can train or use a pre-trained neural machine translation model. Commercial strength grade translation models though are hard to come by and will require a lot of data to train (which is another cost). As such they are even less feasible for lower resourced languages. We were only able to use RTT augmentation on the smaller datasets of AG News and Social Media Hate Speech. Even with this, we had to use two different services (Google and Amazon) to keep costs low. In our academic setting it is not feasible to use RTT on very large datasets and remains future work.

There are a number of avenues of future work. We have provided experiments showing efficacy of different augmentation schemes on a level playing field. More local context augmentation using language models is an avenue of extending this work. Another avenue is investigating how to improve semi-supervised learning of low resourced languages using augmentation. Given the success of *mixup*, one can explore other methods that augment the data as a way of regularisation.

References

1. Aiken, M., Park, M.: The efficacy of round-trip translation for mt evaluation. Translation Journal **14**(1) (2010)
2. Aroyehun, S.T., Gelbukh, A.: Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling. In: Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018). pp. 90–97 (2018)
3. Bird, S., Klein, E., Loper, E.: Natural language processing with Python: analyzing text with the natural language toolkit. " O'Reilly Media, Inc." (2009)
4. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation policies from data. arXiv preprint arXiv:1805.09501 (2018)
5. Das, A., Ganguly, D., Garain, U.: Named entity recognition with word embeddings and wikipedia categories for a low-resource language. ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP) **16**(3), 18 (2017)
6. Davidson, T., Warmsley, D., Macy, M., Weber, I.: Automated hate speech detection and the problem of offensive language. In: Proceedings of the 11th International AAAI Conference on Web and Social Media. pp. 512–515 (2017)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
8. Dundar, M., Kou, Q., Zhang, B., He, Y., Rajwa, B.: Simplicity of kmeans versus deepness of deep learning: A case of unsupervised feature learning with limited data. In: Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on. pp. 883–888. IEEE (2015)
9. Fadaee, M., Bisazza, A., Monz, C.: Data augmentation for low-resource neural machine translation. arXiv preprint arXiv:1705.00440 (2017)
10. Fedus, W., Goodfellow, I., Dai, A.M.: Maskgan: Better text generation via filling in the _. arXiv preprint arXiv:1801.07736 (2018)
11. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford **1**(12) (2009)
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
13. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. pp. 328–339 (2018)
14. Imran, M., Castillo, C., Diaz, F., Vieweg, S.: Processing social media messages in mass emergency: A survey. ACM Computing Surveys (CSUR) **47**(4), 67 (2015)
15. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics. vol. 2, pp. 427–431 (2017)
16. Kobayashi, S.: 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). vol. 2, pp. 452–457 (2018)
17. Krishnan, S., Chen, M.: Identifying tweets with fake news. In: 2018 IEEE International Conference on Information Reuse and Integration (IRI). pp. 460–464. IEEE (2018)
18. Lau, J.H., Clark, A., Lappin, S.: Unsupervised prediction of acceptability judgements. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language

Processing (Volume 1: Long Papers). pp. 1618–1628. Association for Computational Linguistics, Beijing, China (July 2015)
19. Li, Y., Cohn, T., Baldwin, T.: Robust training under linguistic adversity. EACL 2017 p. 21 (2017)
20. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
21. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)
22. Mueller, J., Thyagarajan, A.: Siamese recurrent architectures for learning sentence similarity. In: AAAI. vol. 16, pp. 2786–2792 (2016)
23. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
24. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)
25. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding with unsupervised learning. Tech. rep., Technical report, OpenAI (2018)
26. Ramos, J., et al.: Using tf-idf to determine word relevance in document queries. In: Proceedings of the first instructional conference on machine learning. vol. 242, pp. 133–142 (2003)
27. Ratkiewicz, J., Conover, M., Meiss, M.R., Gonçalves, B., Flammini, A., Menczer, F.: Detecting and tracking political abuse in social media. ICWSM **11**, 297–304 (2011)
28. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. pp. 45–50. ELRA, Valletta, Malta (May 2010)
29. Sano, Y., Yamaguchi, K., Mine, T.: Automatic classification of complaint reports about city park. Information Engineering Express **1**(4), 119–130 (2015)
30. Sennrich, R., Haddow, B., Birch, A.: Improving neural machine translation models with monolingual data. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 86–96 (2016)
31. Smirnov, E.A., Timoshenko, D.M., Andrianov, S.N.: Comparison of regularization methods for imagenet classification with deep convolutional neural networks. Aasri Procedia **6**, 89–94 (2014)
32. Wang, W.Y., Yang, D.: 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. pp. 2557–2563 (2015)
33. Zeiler, M.D.: Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)
34. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 (2017)
35. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in neural information processing systems. pp. 649–657 (2015)