Data Augmentation and Semi-supervised Learning for Deep Neural Networks-based Text Classifier

Heereen Shim*†
Philips Research
Eindhoven, the Netherlands
heereen.shim@philips.com

Stijn Luca
Department of Data Analysis and Mathematical Modelling
Ghent University
Ghent, Belgium
stijn.luca@ugent.be

Dietwig Lowet
Philips Research
Eindhoven, the Netherlands
dietwig.lowet@philips.com

Bart Vanrumste‡
Campus Groep T, e-Media Research Lab
KU Leuven
Leuven, Belgium
bart.vanrumste@kuleuven.be

ABSTRACT

User feedback is essential for understanding user needs. In this paper, we use free-text obtained from a survey on sleep-related issues to build a deep neural networks-based text classifier. However, to train the deep neural networks model, a lot of labelled data is needed. To reduce manual data labelling, we propose a method which is a combination of data augmentation and pseudo-labelling: data augmentation is applied to labelled data to increase the size of the initial train set and then the trained model is used to annotate unlabelled data with pseudo-labels. The result shows that the model with the data augmentation achieves macro-averaged f1 score of 65.2% while using 4,300 training data, whereas the model without data augmentation achieves macro-averaged f1 score of 68.2% with around 14,000 training data. Furthermore, with the combination of pseudo-labelling, the model achieves macro-averaged f1 score of 62.7% with only using 1,400 training data with labels. In other words, with the proposed method we can reduce the amount of labelled data for training while achieving relatively good performance.

KEYWORDS

Text classification, data augmentation, semi-supervised learning, deep neural networks

ACM Reference Format:
Heereen Shim, Stijn Luca, Dietwig Lowet, and Bart Vanrumste. 2020. Data Augmentation and Semi-supervised Learning for, Deep Neural Networks-based Text Classifier. In *The 35th ACM/SIGAPP Symposium on Applied Computing (SAC '20), March 30-April 3, 2020, Brno, Czech Republic.* ACM, Brno, Czech Republic, Article 4, 8 pages. https://doi.org/10.1145/3341105.3373992

*Also with Campus Groep T, e-Media Research Lab, KU Leuven.
†Also with Department of Electrical Engineering (ESAT), STADIUS, KU Leuven.
‡Also with Department of Electrical Engineering (ESAT), STADIUS, KU Leuven.

SAC '20, March 30-April 3, 2020, Brno, Czech Republic

ACM ISBN 978-1-4503-6866-7/20/03.
https://doi.org/10.1145/3341105.3373992

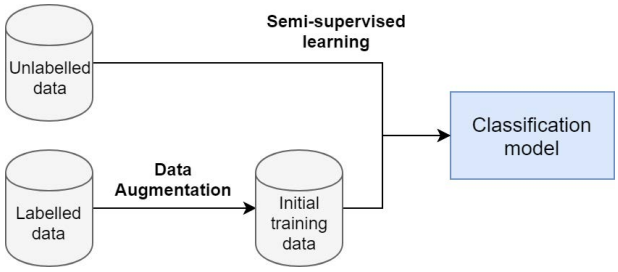

Figure 1: Overview of the proposed approach.

1 INTRODUCTION

User feedback contains rich information about the users and is essential for user-driven development. Many products are providing in-app survey and collecting feedbacks from the users to identify their needs for the better quality of service and support. Especially, open-ended questions are good for understanding user-specific problems. Unlike a closed-ended question that provides pre-defined options limiting the users' answer, the open-ended questions allow the users to answer it in free-text format such that they can answer based on their situation and feeling [5]. The answers to these open-ended questions can be used to obtain detailed information on the users.

One of the biggest challenges with analysing free-text is how to automate the process. Manually analysing free-text is labour-intensive and not suitable as the amount of user feedback is increasing. In this case, developing a free-text analysis tool with the help of recent advances of deep neural networks could be a solution. However, there are technical challenges in applying the deep neural networks to the real-world application: one is labelling data for training the model. Data labelling is a time-consuming and tedious task and it requires a lot of human and financial resources [3]. Moreover, since the labels are prone to be added or deleted as a new batch of user feedback is obtained, the data labelling process is expected to be repeated frequently throughout product development. Minimised manual labelling could mitigate these issues.

In this paper, we focus on the topic of sleep. The goal is to understand user-specific situations and problems via free-text to provide personalised coaching service to users who want to optimise their nights of sleep. As the first step, we collected experimental data containing pairs of a free-text sentence and a set of sleep issues via a web-based survey (Section 3). To automate analysing these free-text data, we aim to build a neural networks-based text classifier with the limited number of labelled data. In this paper, we propose a method which is a combination of data augmentation and semi-supervised learning as shown in Figure 1 (Section 4). We evaluate our method and show the proposed method achieves similar performance while reducing the amount of labelled data (Section 5). Also, we analyse the error of the model and investigate the effects of the proposed method (Section 6).

2 RELATED WORK

Neural language model for NLP tasks

One of the breakthroughs in neural networks based natural language processing (NLP) is attention mechanism [1, 12]. The attention mechanism is firstly proposed to solve long term dependency problems of sequential models [2, 6] that use a single context vector compressing every input from previous time steps. Attention mechanism allows the models to take hidden states from several time steps as inputs and calculate the degree of importance regarding the current time step's input. After Ashish Vaswani et al.[17] proposed Transformer architecture with sorely attention mechanisms, Transformer architecture has been widely used for language models[4, 13] to capture complex linguistic patterns. These pre-trained language models can be easily fine-tuned on various downstream NLP tasks [11, 20, 21]. However, it is widely accepted that the performance of the neural networks-based model is highly dependent on the size and the quality of training data.

Data augmentation for language data

Data augmentation is a technique that can increase the size of a data. Many researchers have been working on data augmentation in various fields, including vision [9] and speech [14]. Compared to these fields, data augmentation for language is less studied and there is no standard method yet. Some researchers proposed data augmentation methods for language data, including synonym replacement by using a thesaurus[22], similar word replacement by using a pre-trained word embedding [18], contextual word replacement by using a pre-trained language model [8], and sentence rephrase by back-translation [15]. However, these techniques are computationally expensive compared to their performance gain. Because of this reason, simple text editing operations are commonly used in practice. Recent research empirically shows that simple text editing operations could contribute substantially to improvements in various text classification tasks [19]. We will explain this simple text editing method in Section 4.2

Semi-supervised learning

Semi-supervised learning focuses on leveraging both labelled and unlabelled data to build a better classifier. Pseudo-labelling, also known as self-training, is a type of semi-supervised learning methods which is used to add more labels with iterative training. In spite of its simplicity, using these pseudo-labels can improve classifier's performance, especially when there are little labelled training data [10]. However, since the classifier uses its predictions to teach itself, pseudo-labelling might reinforce the initial model's error [23].

Table 1: Example of question and answer. Red coloured text shows a spelling error.

Question	What is going on with your sleep?
Answer	I mainly have tow problems. The first is that it's hard for me to stay asleep for more than about an hour without waking up. The other problem is I sometimes have trouble either going to sleep or getting back to sleep once I wake up.

3 DATA

For experiments, free-text data was collected via a web-based survey. Participants was asked to fill in questionnaires in free-text sentences and select sentences representing to their answers. The free-text responses to the open-ended questions will be used as input for the classification model, while the selected sentences will be used as ground truth labels to train and validate the model. In the following subsections, we explain the data collection protocol and provide initial data analysis result.

3.1 Participants

We recruited American adults for the survey by using Amazon's Mechanical Turk (MTurk) platform. Before participating in the survey, participants were informed of the background, purpose, and legal basis of the survey and their rights. When participants signed up for participating in the study, they received the link to the web page hosting the survey. Additional inclusion criteria were applicable:

Inclusion criteria for subject selection.

- They are 18 years or older
- They have an MTurk-approval rate of 97% or higher (this means that at least 97% of the prior tasks they completed on MTurk were of acceptable quality)
- They are proficient in English
- They are willing and able to provide informed consent

3.2 Survey Questions

Participants described issues related to their sleep with at least one complete sentence. Beforehand, the participants were provided with a guide to imagine that they are sitting at the doctor's office because they are having some sleep issues. Table 1 illustrates an example of the question and a user's answer. As we can see, the answer contains spelling errors. After describing sleep-related issues, participants were asked to select at most 3 sentences that capture the meaning of their answers from Table 2. Labels of the selected sentences are used as ground truth labels in experiments.

Table 2: Options for selecting matched sentences.

Label	Sentence
troubleFallingAsleep	I lie in bed awake have trouble falling asleep
troubleStyaingAsleep	I have been waking up frequently
wakeUpTooEarly	I am waking up too early (before I want/have to)
staysUpLate	I am staying up (too) late
sleepsInLater	I am sleeping in (too) late
problemWakingUp	I have trouble waking up
SnoringBothersMe	I am bothered by my snoring
SnoringBothersOthers	Others are bothered by my snoring
SnoringStoppedBreathing	I stop breathing during the night
otherIssue	I have another concern
goodSleep	I have no sleep concern

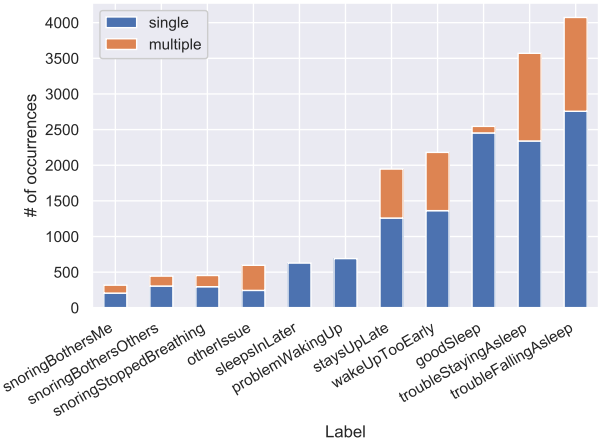

Figure 2: Label distribution of the train set. Blue graphs represent when the sample is single-labelled and orange graphs represents when the sample is multi-labelled.

3.3 Data Analysis

In total 16,096 sentences were collected. We split data into train and test set: the train set consists of 14,363 samples and the test set consists of 1,733 samples. Figure 2 illustrates the label distribution of train set. It is observed that the data distribution is highly skewed: it has imbalances between labels and between single- and multiple-labelled data.

4 METHOD

4.1 Classification model

Bidirectional Encoder Representations from Transformers (BERT) [4] is used as a baseline text classifier. We use a pre-trained BERT model and fine-tune on our data to detect multiple sleep issues from the given free-text input. As it is illustrated in Figure 3, we add a dense layer on the top of the pre-trained BERT model and the final hidden vector of the classification token [CLS] is fed into this dense

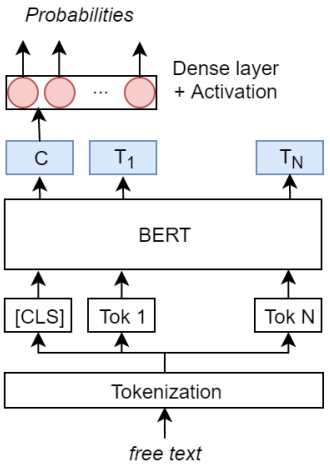

Figure 3: Overview of the BERT model for multi-label classification.

layer. To perform multi-label classification, the sigmoid function is used for an activation function and binary cross-entropy is used as a loss function. For more details on tokenization and BERT model, please refer the original paper [4].

4.2 Data augmentation

We use Easy Data Augmentation (EDA) technique [19] which consists of four different text editing operations:

- **Synonym replacement**: N words are randomly selected from the sentence and replaced with one of its synonyms chosen at random.
- **Random noise injection**: N words are randomly selected from the sentence and a single character of each word is replaced with a random alphabetical character.
- **Random swap**: we randomly choose two words in the sentence and swap their positions and repeat N times.
- **Random deletion**: N words from the sentence are randomly chosen and removed from the sentence.

The value of N is decided based on the length of each sentence. We set a percentage $p = 0.1$ and calculated $p \times len(sentence)$, where the number words in the sentence is used as a length of the sentence. Rounded up value of $p \times len(sentence)$ is used as the value of N.

During data augmentation, we select sample sentences consisting of more than 5 words to avoid too short sentences. Four operations are applied separately to each sentence. For synonym replacement, we only select a word that contains more than two characters to avoid selecting too short words. Also, unlike the original paper [19], we insert random noise rather than a synonym. This can be seen as introducing misspelling to make the model robust to a spelling error, which is common in user-generated free-text. Table 3 illustrates examples of text editing operations.

4.3 Pseudo-labelling

We use pseudo-labelling [23] as a semi-supervised learning method. During pseudo-labelling procedure, a classifier is trained on the

Table 3: Examples of text editing operations.

Operation	Text
Original	I snore a lot.
Synonym replacement	I snore a ~~lot~~ entirely.
Random noise injection	I snore~~t~~ a lot
Random swap	snore I a lot
Random deletion	I snore a ~~lot~~

initial labelled data and then the trained model is used to do classify unlabelled data. The predicted labels with a high confidence score, which are called pseudo-labels, are then added to the new training set. For selecting pseudo-labels, we set a threshold value of 0.6. Then the classifier is re-trained with the new training set consisting of the initial labelled and the pseudo-labelled data. This process is repeated until it reaches a certain termination condition. In this paper, we set the termination condition based on the number of iterations and the size of the pseudo-labelled data. Until the number of iterations reaches the limit, which is set as 5, we check the size of the pseudo-labelled data. If the size of the pseudo-labelled data is not bigger than the pseudo-labelled data from the previous step, the pseudo-labelling process is terminated. Algorithm 1 illustrates the pseudo-labelling procedure.

Algorithm 1: Pseudo-labelling procedure

Data: Training set D_t, labelled set D_l, unlabelled set D_u
Result: New training set $\hat{D}_t$, trained model M_i

$D_t \leftarrow D_l$
$i = 0$
termination condition $\leftarrow$ False
while *termination condition == False* **do**
  $M_i \leftarrow \text{train}(D_t)$
  predictions $\leftarrow \text{inference}(M_i, D_u)$
  $D_p \leftarrow \text{thresholding}(\text{predictions})$
  $\hat{D}_t \leftarrow D_t + D_p$
  termination condition $\leftarrow$ check condition(D_p, i)
  if *termination condition == False* **then**
    $D_t \leftarrow \hat{D}_t$
    $i+ = 1$
  end
end

5 EXPERIMENTS AND RESULTS

To validate our method, 4 experiments were conducted: Firstly, we check the baseline model's performance without data augmentation and pseudo-labelling. Secondly, we apply data augmentation and train the model with augmented data. Thirdly, we apply pseudo-labelling and iteratively train the model. Lastly, we train the initial model with augmented data and iteratively train the model with pseudo-labels. The purpose of these experiments is to evaluate how the proposed method can contribute to performance improvement.

Table 4: Detailed implementation specification.

Item	Specification
CPU	Intel® Xeon® W-2123 CPU @ 3.60 GHz
GPU	NVIDIA GeForce GTX 1080 ti, 11 GB memory
Graphic driver	NVIDIA graphic driver version 416.34
CUDA	Version 10.0
OS	Windows 10, 64-bit
Python	Version 3.6.6
Pytorch	Version 1.0.1

Evaluation Metric

In our experiment, we use f1 score for each label as an evaluation metric, which is defined as follows:

$$Precision = \frac{tp}{tp + fp}$$
$$Recall = \frac{tp}{tp + fn}$$
$$F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

where tp, fp, and fn represent true positive, false positive, and false negative of each label, respectively.

Additionally, we use marco-, micro-averaged f1 scores [16]. Macro-averaged f1 is per label averaged and does not take label imbalance into account. Micro-averaged f1 is calculated by counting the total true positives, false negatives, and false positives so that it would be more affected by the performance of the classes which has more examples.

Settings

All experiments were performed on the Windows 10 operating system. The detailed specification of hardware and software is summarized in Table 4. We used PyTorch version of BERT [7]. The smallest model, whose size of the final hidden vector of classification token is 512, was used for the experiments with pre-trained model weights. Softmax function in the final output layer was changed to sigmoid function to perform multi-label classification. We did not change other hyperparameter settings except the number of training epochs: we trained the model on training datasets for 10 epochs.

5.1 Baseline model

We trained the BERT with the entire training data. Table 5 shows the result. It is observed that the trained model achieved varying performances over labels: it achieves relatively high performance on some labels (e.g., *troubleFallingAsleep*, *troubleFallingAsleep*, and *goodSleep*) that occurred more in train set than other labels (e.g.,*otherIssue* and *snoringBothersMe*) where the model achieves low performances. This result implies that the trained model tends to performs well on some labels with more training data compared to other labels with less training data. We call these labels with relatively few training samples as minority labels. This results in a large difference between macro- and micro-averaged performances. In our case,

Table 5: Classification results when using entire training data.

Label	Precision	Recall	F1
troubleFallingAsleep	0.79	0.84	0.82
troubleStayingAsleep	0.78	0.79	0.79
wakeUpTooEarly	0.74	0.68	0.71
staysUpLate	0.74	0.65	0.69
problemWakingUp	0.75	0.74	0.75
sleepsInLater	0.64	0.57	0.60
snoringBothersOthers	0.82	0.65	0.73
snoringBothersMe	0.67	0.38	0.49
snoringStoppedBreathing	0.78	0.56	0.65
goodSleep	0.97	0.94	0.96
otherIssue	0.42	0.25	0.31
Macro-averaged	0.74	0.64	0.68
Micro-averaged	0.79	0.75	0.77

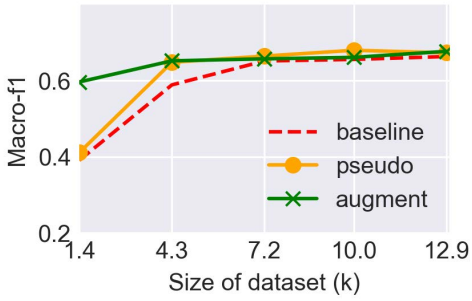

Figure 4: Performances based on various dataset sizes. X-axis represents the size data used for training. Y-axis represents macro-averaged f1 score.

the macro-averaged f1 score is suitable for evaluating the model's performance. Because micro-averaged performance might mislead interpretation that the trained model works well even though it misclassifies minority labels.

We further investigate how the size of the training set affects the model's performance. We trained model with the following training set fractions (%): {10, 30, 50, 70, 90, 100} whose sizes (k) are: {1.4, 4.3, 7.2, 10.0, 12.9, 14.3}. In Figure 4, the red dashed line shows the performance of the model based on the size of the training set. Interestingly, the model achieves nearly 90% of performance upper limit - that can be achieved when around 14,300 samples are used - with only using around 4,300 samples. Also, we can see that the size of the training set does not have a significant impact on the model's performance after around 7,200. This shows that after some number of data, the performance tends to be saturated. Details of the training set size and its performance are described in the following Section 6.2.

5.2 Data augmentation

We investigated the effect of data augmentation while varying the size of the training data. In Figure 4, the green line shows the performance of the model with data augmentation. From Figure 4, it can be seen that the data augmentation provides the largest performance increase when the training data is the smallest: it is observed that macro-f1 score is increased from 39.13% to 59.7% when only using 1,400 samples. However, the amount of performance improvement decreases as the training size increases. This suggests that the best scenario to apply data augmentation is when the only small size of data is available for training. Details of additional training data made by data augmentation are given in the following Section 6.2.

5.3 Pseudo-labelling

We iteratively trained a model with a subset of training data as a labelled data and the remaining as an unlabelled data by applying pseudo-labelling. We investigate how the size of the labelled training data could affect the final model's performance. In Figure 4, the orange line shows the performance of the model trained with pseudo-labelling. Similar to the data augmentation result, the performance improvement tends to decrease as the size of labelled data increases. The largest improvement is observed when around 4,300 samples are given as labelled data: the model use 10,000 of unlablled data with pseudo-labels achieves a macro-f1 score of 64.8% while the model trained without pseudo-labels achieves 58.9%. One noticeable thing is that when only around 1,400 samples are given as a labelled data, there is almost no performance increase even after iterative training with 12,900 unlabelled samples with pseuo-labels: it only increases from 39.1% to 41.1%. We will discuss this in the following Section 6.3.

5.4 Data augmentation + Pseudo-labelling

We apply data augmentation to the initially given labelled data and iteratively train the model by using unlabelled data with pseudo-labels. As it can be seen from the Table 6, the baseline model, without data augmentation and pseudo-labelling, achieves macro-averaged f1 of 39.1% when around 1,400 of labelled data are given for training. If we train the model with 1,400 of labelled data and 11,900 of unlabelled data with pseudo-labels, it slightly improves the performance to 41.2%. Compared to this, if data augmentation is applied to the 1,400 of labelled data and the model is iteratively add more training data with pseudo-labels, it achieves macro-averaged f1 score of 62.7%. However, it can be interpreted that this improvement is mainly derived from data augmentation, because the model trained with 1,400 of labelled data and additional augmented data without pseudo-labels already achieves macro-averaged f1 score of 59.7%. In other words, data augmentation can contribute to performance improvement significantly when there is a little amount of labelled data. On top of that, pseudo-labelling can provide additional performance increase by using unlabelled data.

Table 6: Details of models' training data sizes and compared performances.

Model	Labelled	Augmented	Unlabelled	Macro-f1
Baseline	1,436	0	0	0.39
Pseudo-labelling	1,436	0	12,927	0.41
Data augmentation	1,436	5,607	0	0.60
Data augmentation +pseudo-labelling	1,436	5,607	12,927	**0.63**

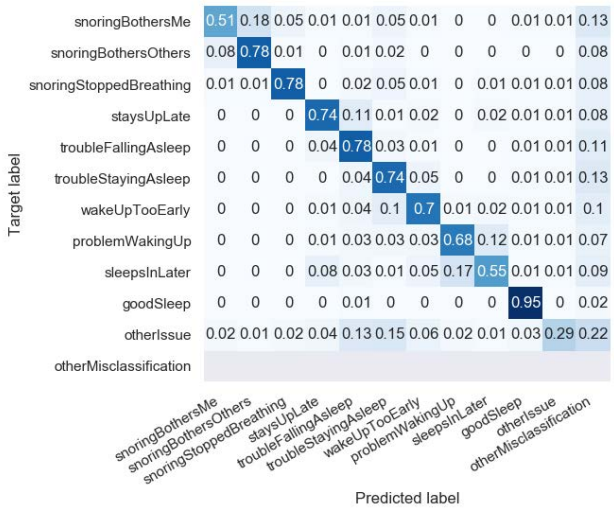

Figure 5: Normalised confusion matrix of the trained model's predictions. A row represents target label, whereas a column represents predicted label. The values of the diagonal elements represent the degree of correctly predicted classes.

6 DISCUSSION

6.1 Misclassification analysis

To analyse the misclassification, firstly we plot a confusion matrix based on predictions made by the baseline model from Section 5.1. Since our case is multi-label classification, we select samples whose ground truth label set contains only single labels. We added *otherMisclassification* label, which means that the trained model predicted more than one labels. As it is shown in Figure 5, misclassification happens more often in between similar labels: the trained model often misclassifies *snoringBothersMe* as *snoringBothersOthers* and sometimes fails to distinguish *problemWakingUp* and *sleepsInLater*. We speculate that this is because samples with these labels are too close to be distinguished from each other. This suggests avoiding pre-defining too similar labels for classification.

6.2 Data augmentation result analysis

We investigate the effects of the size of the training set and data augmentation on the model's performance. Table 7 shows the size of each fraction with and without data augmentation and performance with each dataset. To analyse the effect of data augmentation, two comparisons are given: model trained with augmented data when (1) data augmentation is only applied to training samples of minority labels or (2) data augmentation is applied to entire data. Minority labels mean the labels with relatively few training samples, including *snoringBothersMe, snoringBothersOthers, snoringStoppedBreathing, otherIssue, sleepsInLater, and problemWakingUp*. In Table 7, min and max represent a minimum and a maximum number of training samples per label, respectively. For example, in 10% of data, the smallest label set (*snoringBothersMe*) consists of 29 samples and the largest label set (*troubleFallingAsleep*) consists of 399 samples. From the Table 7, we can see that even when data augmentation is applied to only data of minority labels, the model's performance is similar to when data augmentation is applied to entire data which means that there are more than two times many training samples. This implies that what plays a key role in data augmentation is the number of training samples of minority labels, not the total size of the training set.

Table 7: Size of training set and data augmentation result and trained model's performance.

Percent of dataset	Min	Max	Total	Macro-f1
10%	29	399	1436	0.39
30%	98	1,221	4309	0.59
50%	167	2,044	7,182	0.65
70%	219	2,829	10,054	0.66
90%	297	3,623	12,927	0.66
100%	317	4,073	14,363	0.68
Data augmentation on data of minority labels				
10% + data augmentation	133	486	2,555	0.56
30% + data augmentation	460	1,376	7,170	0.60
50% + data augmentation	774	2,126	11,385	0.64
70% + data augmentation	1,005	2,750	15,326	0.66
90% + data augmentation	1,331	3,371	18,970	0.65
100% + data augmentation	1,551	5,260	26,206	0.68
Data augmentation on entire data				
10% + data augmentation	133	1,946	7,043	0.60
30% + data augmentation	458	5,913	20,809	0.65
50% + data augmentation	776	9,708	34,059	0.66
70% + data augmentation	1,006	13248	47,024	0.66
90% + data augmentation	1,335	16749	59,595	0.68
100% + data augmentation	1,557	20,283	71,377	0.69

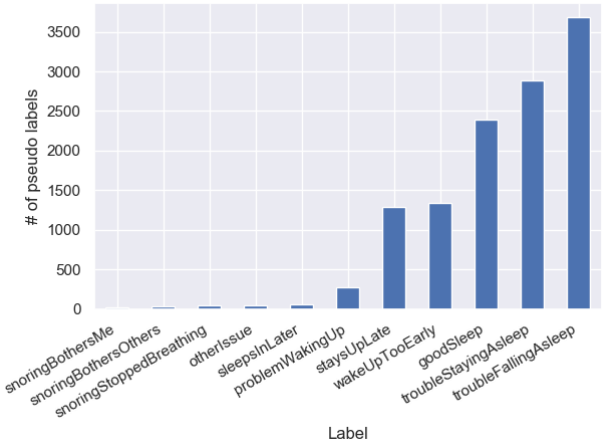

Figure 6: The number of pseudo-labelled data obtained by using the initial model trained with 1,400 labelled data.

Table 8: The number of pseudo-labels and increase over iterative training.

Iteration	Pseudo-labels	Increase
1	10,059	10,059
2	11,705	1,646
3	12,139	434
4	12,344	205
5	12,451	107

6.3 Pseudo-labelling result analysis

In previous Section 5.3, we showed that the model trained with 1,400 labelled data and 12,900 unlabelled data with pseudo-labels achieves the almost same performance of the model trained without pseudo-labels. We hypothesise that this is because the initial model trained with 1,400 samples with ground truth is not robust enough to get sufficient pseudo-labels. To validate this, we investigate the number of pseudo-labels obtained by using the initial model. As it is shown in Figure 6, there is almost no pseudo-labelled data of minority labels. This means that no additional samples of minority labels will be added to the new training set for the next iteration. This could enhance the data imbalance and result in poor performance at the end of iterative training. This suggests that the initial model's performance, especially for minority labels, is critical in the pseudo-labelling method.

Another observation from analysing pseudo-labelling result is that the termination condition of the size of pseudo-labelled data is not strict enough: as shown in Table 8, during the iterative training the size of pseudo-labelled data is increasing, but with negligible margin after the first iteration. Therefore, the iteration was repeated until it met the termination condition of the iteration number, which is set as 5 times in this paper. In future work, adding a margin for the termination condition of the size of pseudo-labelled data is foreseen to avoid unnecessary iterations.

6.4 Data augmentation and pseudo-labelling efficiency analysis

To evaluate the efficiency of the proposed method, we investigate the computation power required to train each model. Table 9 summarises the required training time for each model and its training data. For pseudo-labelling, the values of training set and training time are for a single iteration and the value of the performance is the final model's performance. It is observed that data augmentation on 100% of data does not contribute to performance increase significantly when considering its increase of training time. Unlikely, data augmentation on 10% of data provides a relatively high performance boost with only around 15 minutes of training time increase. For pseudo-labelling, it seems not efficient compared to data augmentation, because it requires multiple training sessions. As it is described in Section 6.3, considering the number of newly added pseudo-labels sharply decreases after the first iteration, the best scenario is to train the model with augmented data and conduct pseudo-labelling only 1-2 times.

7 CONCLUSION

In this paper, we propose a method which is a combination of data augmentation and semi-supervised learning to reduce manual data labelling process for developing a deep neural networks-based text classification model. To validate our method, experiments on how each method could contribute to the performance improvement with various settings were conducted. We experimentally showed that applying data augmentation can improve the model's performance, especially when there is little training data. Also, the result shows that the size of minority labels is critical to the model's performance when the training data is imbalanced. Furthermore, using unlabelled data with pseudo-labels can provide additional performance improvement. However, for the pseudo-labelling, the training time increases as the training sessions are iterated. These results suggest two possible scenarios: Firstly, develop an initial model with augmented data when there is little training data. Secondly, apply pseudo-labelling when there is additional data which is not labelled yet and iterate the process only 1-2 times. We expect this method can boost the development process by reducing manual data labelling.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 766139. This article reflects only the author's view and the REA is not responsible for any use that may be made of the information it contains.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).
[2] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).
[3] Michael Chui, James Manyika, and Mehdi Miremadi. 2018. What AI can and can't do (yet) for your business. *McKinsey Quarterly* (2018).

Table 9: Details of each model's training set, approximated training time, and performance.

Model	Train set	Time	Macro-f1
Baseline with 100% of data	14,363	45m	0.68
Data augmentation on 100% of data	71,377	3h30m	0.69
Baseline with 10% of data	1,436	5m	0.39
Data augmentation on 10% of data	7,043	20m	0.60
Pseudo-labelling with 10% of data	$\leq$ 14,363	$\leq$ 45m	0.41
Data augmentation on 10% of data + pseudo-labelling	$\leq$ 19,970	$\leq$ 20m + 45m	0.63

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[5] Barbara Snell Dohrenwend. 1965. Some effects of open and closed questions on respondents' answers. *Human Organization* 24, 2 (1965), 175–184.

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

[7] Huggingface [n. d.]. PyTorch - BERT. https://github.com/huggingface/pytorch-transformers.

[8] Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201* (2018).

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[10] Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, Vol. 3. 2.

[11] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[12] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[13] Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. 2018. *Improving language understanding with unsupervised learning*. Technical Report. Technical report, OpenAI.

[14] Anton Ragni, Katherine Mary Knill, Shakti P Rath, and Mark John Gales. 2014. Data augmentation for low resource languages. (2014).

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

[16] Mohammad S Sorower. 2010. A literature survey on algorithms for multi-label learning. *Oregon State University, Corvallis* 18 (2010), 1–25.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[18] William Yang Wang and Diyi Yang. 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*. 2557–2563.

[19] Jason W Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196* (2019).

[20] Yuxiang Wu and Baotian Hu. 2018. Learning to extract coherent summary via deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[21] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv preprint arXiv:1906.08237* (2019).

[22] Ye Zhang and Byron Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820* (2015).

[23] Xiaojin Jerry Zhu. 2005. *Semi-supervised learning literature survey*. Technical Report. University of Wisconsin-Madison Department of Computer Sciences.