Quantifying the Evaluation of Heuristic Methods for Textual Data Augmentation

Omid Kashefi and **Rebecca Hwa**
School of Computing and information
University of Pittsburgh
{kashefi, hwa}@cs.pitt.edu

Abstract

Data augmentation has been shown to be effective in providing more training data for machine learning and resulting in more robust classifiers. However, for some problems, there may be multiple augmentation heuristics, and the choices of which one to use may significantly impact the success of the training. In this work, we propose a metric for evaluating augmentation heuristics; specifically, we quantify the extent to which an example is "hard to distinguish" by considering the difference between the distribution of the augmented samples of different classes. Experimenting with multiple heuristics in two prediction tasks (positive/negative sentiment and verbosity/conciseness) validates our claims by revealing the connection between the distribution difference of different classes and the classification accuracy.

1 Introduction

Machine learning approaches have been shown to be capable of making accurate predictions in many well-known problem domains with an abundance of training data. This heavy reliance on the availability of the data, however, may hamper the application of machine learning approaches to resource-limited problem domains, where a sizable training data are not always available.

There is a growing body of research on training under resource scarcity, and data augmentation is one of such techniques. It aims to reconcile the data requirement of the machine learning approaches by applying a general (e.g., randomly remove a word) or domain-inspired heuristic (e.g., replace an adjective with an antonym) to the (limited) existing data in order to generate more training samples.

One challenge for data augmentation is in choosing the most appropriate heuristic for the application in question. There may be many domain-independent augmentation heuristics, and a domain expert may come up with many different domain-inspired heuristics; but the choices of which examples from these heuristics to use may have a significant impact on the success of the trained model.

A straightforward approach to choose an augmentation heuristic is to actually perform the classification experiment on all possible augmented datasets and then chose the best performing one(s) based on the evaluative results. However, this approach may not be computationally practical when there are too many augmentation heuristic options.

In this paper, we propose an alternative heuristic evaluation approach based on the idea that a good heuristic should aim to generate *"hard to distinguish"* samples for different classes. We further argue that the generation quality of "hard to distinguish" examples could be quantified as the difference between the distribution of the augmented samples that a heuristic generates for different classes.

To calculate the distribution difference, we proposed to use pre-trained off-the-shelf embeddings to convert sentences into class distributions, then calculate the KL-divergence between them and used that as a metric to evaluate the "hard to distinguish" examples that a heuristic produces.

We validate our proposed heuristic evaluation approach by experimenting with multiple heuristics and augmented datasets for two classification tasks: predicting whether a sentence expresses positive or negative sentiment and predicting whether a sentence is verbose or concise. Results suggest that quantifying the "hard to distinguish" example generation quality of the heuristics as the difference between class distribution of the augmented examples, could be served as an effective metric for choosing a suitable augmented dataset for a classification task.

2 Data Augmentation

Data augmentation is a technique for generating additional training data by applying a heuristic transformation to the existing training examples. For example, an existing image could by rescaled or flipped to get more images with the same label to expand the size and diversity of the training dataset and thus train a more reliable and accurate model (Frénay and Verleysen, 2014; Hendrycks et al., 2018; Shorten and Khoshgoftaar, 2019).

In general, data augmentation could be formulated as Equation 1, where h is a heuristic function that transforms the datapoint and label pair of (x, y) to a new augmented sample $(\hat{x}, \hat{y})$.

$$(\hat{x}, \hat{y}) = h(x, y) \tag{1}$$

The majority of existing data augmentation approaches are *label-preserving*, which relaxes the Equation 1 as $(\hat{x}, y) = h(x, y) = (h(x), y)$; this means, if x belong to some class A, augmented $\hat{x}$ also belong to class A. For example, using a synonym replacement heuristic, a sentence with positive sentiment could be augmented into a new example, while preserving the overall positive sentiment. Label-preserving data augmentation requires existing labeled samples for every class that is needed to be augmented.

Data augmentation can be *non-label-preserving* as well, where the label itself might also transform using function h_y that expands Equation 1 as:

$$(\hat{x}, \hat{y}) = h(x, y) = (h_x(x), h_y(y))$$

This means, while x belongs to class A, $\hat{x}$ might not necessarily belong to class A. For example, by replacing the most positive word(s) of a sentence with positive sentiment with an antonym, the sentence's sentiment may become negative. Non-label-preserving data augmentation is not bound to the assumption of having labeled samples for instances of all classes and samples from one class may be enough to generate instances of other classes.

Given a classification task, there may be multiple heuristics and data augmentation approaches that allow us to transform existing samples to new ones, but the choice of heuristic may significantly impact the success of the task. In this paper, we aim to answer the key question: *"which heuristic and data augmentation approach is more appropriate for a classification task?"* In Section 3, we propose a low-cost approach to quantify the evaluation of different heuristics and the resulting augmented datasets for classification tasks.

We believe our proposed approach could be a contribution to the NLP community because data augmentation has been shown to be useful for many NLP applications, with researchers proposing many different approaches for text data augmentation; for example, (Zhang et al., 2015; Wei and Zou, 2019) used thesaurus-based and (Wang and Yang, 2015; Kobayashi, 2018; Jiao et al., 2019) used embedding-based lexical substitution approach, (Wei and Zou, 2019; Xie et al., 2019) used random noise injection, including random word insertion, deletion, or sentence shuffling, (Luque, 2019) used instance crossover by combining halves of tweets, (Guo et al., 2019) adapt the mixup approach (Zhang et al., 2018) to text by interpolating the distributed representation of different sentences, (Sennrich et al., 2016; Fadaee et al., 2017; Xie et al., 2019) used back-translation, and (Hu et al., 2017; Iyyer et al., 2018; Anaby-Tavor et al., 2020; Kumar et al., 2020) used (deep) generative models to augment more training examples.

However, with all these textual augmentation options, trying all of them for a (classifier) training task might be impractical, and to our best knowledge, there is not a guideline for how to choose between them for a task.

3 Quantification of Heuristics Suitability

A straightforward approach to assess which heuristic and data augmentation approach is more appropriate for the task is to try every heuristic to generate an augmented dataset, then train a classifier on each and check the final classification performance (Qiu et al., 2020; Wei and Zou, 2019). The training process in this brute-force approach, however, may be time-consuming and resource-intensive, especially in complex training scenarios.

Alternatively, we may try to identify qualities that make a heuristic effective. Intuitively, a good heuristic ought to generate augmented samples that are the most similar to the original data distribution. However, this approach may overlook the additional generalization benefit that may come from diverse augmented training examples. Moreover, this approach may not be possible for problem domains with limited resources, where original labeled data is not available for all classes, and one may have to use non-label-preserving heuristics to augment examples for all classes.

On the other hand, from the classification task perspective, a good heuristic should aim to generate near-miss examples (samples of class B hard to distinguish from A). We believe, the "hard to distinguish" samples can be quantified by finding a way to compute the difference between the samples of different classes, to sever as an guideline for choosing between different heuristic approaches.

Let us assume samples of class A are drawn from distribution A, which should be different from distribution B that samples of class B are drawn from. The difference between distribution A and B can be calculated as the KL-divergence (KLD) (Kullback and Leibler, 1951) from B to A as: $D_{KL}(A||B)$. KLD calculates how probability distribution A is different from the reference probability distribution B as the amount of information gained if samples of B are used instead of samples of A.

Thus, a lower $D_{KL}(A||B)$ means distribution A is more similar to distribution B, so samples of class A are *harder to distinguish* from samples of class B. Therefore, the extent to which "hard to distinguish" samples can be generated by heuristic h could be quantified as $D_{KL}(A_h||B_h)$, where A_h and B_h indicate the samples of class A and B augmented using heuristic h, and Equation 2 could be used to identify which heuristic is generating "harder to distinguish" samples and so more suitable for the classification task.

$$\arg\min_h D_{KL}(A_h||B_h) \tag{2}$$

Finally, to transform sentences from their discrete word representation into a continuous distribution representation, we utilize a few of the numerous pre-trained embeddings that nowadays are the de facto approach for encoding sentences into vector space (Cho et al., 2014; Le and Mikolov, 2014; Cer et al., 2018; Devlin et al., 2019).

We examine the applicability of our proposed approach by studying two classification tasks: *sentiment analysis*, as a resource-rich problem domain that allows experimenting with both label-preserving and non-label-preserving heuristics, and *verbosity analysis*, as a resource-limited problem domain that the absence of sizable labeled data limits the options to non-label-preserving heuristics.

4 Augmented Datasets

In this section, we go over some heuristic options for augmenting training corpora for sentiment analysis and verbosity detection domains.

4.1 Augmented Sentiment Corpus

Our sentiment analysis task is to predict whether a sentence expresses *positive*, *negative*, or *neutral* sentiment? For this task, we use the sentences from the Yelp Polarity Dataset (YPD) (Zhang et al., 2015) to create the augmented dataset.

As *label-preserving* heuristics, we use following heuristics proposed by Wei and Zou (2019):

- **Synonym Replacement (SR).** Randomly pick a content word from the sentence and replace it with a synonym chosen at random.
- **Random Insertion (RI).** Randomly choose a content word from the sentence and insert one of its synonyms to a random place in the sentence.
- **Random Swap (RS).** Swap the position of two randomly chosen words in the sentence.
- **Random Deletion (RD).** Delete a randomly chosen word from the sentence.

We apply these heuristics to the positive sentences of YPD to generate more positive examples, and the other way around for generating more negative examples. For each sentence, we repeat each heuristic operation until about 20% of its words are changed ($\alpha = .2$).

Moreover, we propose the following *non-label-preserving* heuristics and apply them to the positive sentences to create the augmented negative examples, and the other way around for generating the augmented positive examples.

- **ALL.** In this heuristic, we replace all the sentiment words of a sentence with their antonyms. To find the sentiment words, we first collected a vocabulary of positive and negative unigrams by combing the labeled words of *Stanford Sentiment Treebank* (Socher et al., 2013) and the *Opinion Lexicon* (Hu and Liu, 2004). This results in a vocabulary of 3,453 positive and 6,000 negative unigrams.

  Then, for a positive sentence in YPD, we replace every word of it that appeared in the positive portion of the collected vocabulary by one of its randomly chosen antonyms, using WordNet (Miller, 1995), to create the augmented negative sentence. We perform similarly but in the opposite direction to create the augmented positive sentences.

Heuristic	Sentence	Label
None	super generous portion !	positive
SR	super generous ~~portion~~ **+slice** !	positive
RI	super **+slice** generous portion !	positive
RS	**portion**$^{\leftarrow}$ generous $^{\rightarrow}$**super** !	positive
RD	super generous ~~portion~~ !	positive
ALL	~~super~~ **+lousy** ~~generous~~ **+meager** portion!	negative
ONE	super ~~generous~~ **+meager** portion !	negative

Table 1: Examples of Sentences Augmented using Label-Preserving (SR, RI, RS, and RD) and Non-Label-Preserving (ALL and ONE) Sentiment Heuristics

- **ONE.** In this heuristic, instead of replacing all sentiment words with their randomly chosen antonym, we first filtered for antonyms that match the POS and sense of the sentiment word, then we pick the antonym that makes the most fluent augmented sentences, ranked by a language model (LM) trained on YPD. Finally, for every sentence, we only replace one of its sentiment words with its POS, sense, and LM filtered antonym.

  Using this heuristic, for example, a sentence with overall positive polarity may still contain a word that expresses a negative opinion about an aspect, so intuitively, this creates "harder to distinguish" examples compared to the **ALL** heuristic.

In total, we generated 50K positive and 50K negative augmented samples using each heuristic. We removed all of the original YPD sentences so that these datasets contain only augmented samples. We refer to each dataset with the same name as the heuristic function it is augmented with. Table 1 shows examples of sentences augmented using the label-preserving and non-label-preserving sentiment heuristics.

4.2 Augmented Verbosity Corpus

The verbosity detection task is to predict whether a sentence is *verbose* or *concise*. Unlike the sentiment analysis domain, the set of existing resources for the verbosity detection problem is much more limited: NUCLE covers grammatical redundancy (Dahlmeier et al., 2013), and Kashefi et al. (2018) has a small corpus called Semantic Pleonasm Corpus (**SPC**) that contains semantic redundancy (i.e., verbosity) labels. Due to its small size, it is primarily suitable as a benchmark.

Since to the best of our knowledge, there is no sizable resource with explicit *verbose* and *concise* labels, to augment a dataset of concise and verbose sentences, we start by trying to identify an existing real-world data source that has verbosity or conciseness characteristics. One domain-specific feature of Yelp that we exploit is the data category called "tips." Since "tips" are very short sentences, they are likely to be concise; we sample for "tips" that contain adjectives because the evaluation corpus (i.e. SPC) mainly focuses on adjectival semantic redundancies.

Based on domain knowledge, we come up with the following *non-label-preserving* heuristics to create verbose samples based on the collected "concise" sentences by adding a superfluous adjective to the concise sentences:

- **Duplicate (DUP).** This heuristic is an obvious case for word redundancy by duplicating an adjective word of the sentence right next to itself.

- **Synonym (SYN).** This heuristic inserts a synonym next to an adjective word of the sentence. The conventional way to get synonyms of a word is to use WordNet, however, since these synonyms may express a different quality of the noun clause compared to the original adjective, augmented construction might not be semantically redundant.

  For this reason, we opt to use sense2vec (Trask et al., 2015), a contextual word-embedding fine-tuned on Yelp "tips". Since the adjective synonyms from sense2vec are matching the context and follow the same intent and emotional state of the original adjective, these two adjacent synonyms are likely to make a pleonastic construction.

- **Near-Miss Negative (NMN).** In this heuristic, we try to create **concise** examples that are "hard to distinguish" from the verbose examples. We trained a language model on the Yelp "tips" and used that to predict the most likely words that can occur right after an adjective of the sentence. Let assume for adjective w_{adj} in sentence s, using LM, we retrieved $\{w_{aug1}, w_{aug2}, ..., w_{aug5}\}$ as a sorted list of most likely words that can appear next to w_{adj} given its context s.

Heuristic	Sentence	Label
None	delicious bread !	concise
DUP	delicious **+delicious** bread !	verbose
SYN	delicious **+tasty** bread !	verbose
NMN	delicious **+redolent** bread !	concise

Table 2: Examples of Sentences Augmented using the Verbosity Heuristics

We then filter for w_{aug}s that are adjective themselves and a synonym of w_{adj}, lets assume the filtered list be $\{w_{aug2}, w_{aug5}\}$. Since LM is trained on Yelp, the w_{aug2} is already observed in the Yelp tips after the w_{adj} in some context. Taking into account that Yelp tips are considered concise, the sequence of $\ldots w_{adj}\ w_{aug2} \ldots$ is also concise. Therefore, we can create concise examples that are containing two adjacent synonyms but are **not** verbose

For each heuristic, we generate only one augmented verbose sample from an original concise sentence. In total, we augmented 100K concise and 100K verbose samples using each heuristic. Since the verbose examples are generated from concise sentences that are included in the augmented corpus, we removed the concise sentences with odd and verbose sentences with even indexes to make sure that non of the concise are verbose sentences in the corpus are corresponding to each other. The final augmented corpus, thus, contains 50K non-parallel samples of each class. We refer to each dataset with the same name as the heuristic function that was used to augment it.

Table 2 shows examples of sentences augmented using the non-label-preserving verbosity heuristics. While duplicating the word "delicious" or adding "tasty" next to it makes the sentence verbose, adding "redolent" does not make it verbose because "redolent" and "delicious" are describing different quality of the "bread."

5 Experiments

The key questions for validating our proposed approach for quantifying the evaluation of heuristic textual data augmentation methods are:

- **Q1.** Can generating "hard to distinguish" examples be an effective way to assess whether a heuristic is generating a suitable augmented training dataset?
- **Q2.** To what extent could the notion of "hard to distinguish" examples be quantified by our proposed metric – the difference between the class distribution of the augmented samples?
- **Q3.** Is calculating the difference of class distributions computationally efficient in practice?

To measure the accuracy of sentiment and verbosity classification in answering **Q1**, we trained an **LSTM** (Liu et al., 2016) and a **CNN** (Kim, 2014) classifier on each the augmented dataset. The classification result for each task and augmented dataset is reported in Section 5.2.

The LSTM and CNN models are trained on augmented corpora separately for each task; the sentiment classifiers are evaluated on a held-out portion of the YPD, and the verbosity classifiers are evaluated on SPC. None of the sentences of the held-out YPD and SPC are used during the creation of the augmented datasets.

To answer **Q2**, we use two pre-trained encoder models: Universal Sentence Encoder (**USE**) (Cer et al., 2018) and Bidirectional Encoder Representations from Transformers (**BERT**) (Devlin et al., 2019), both of which are transformer-based encoder of greater-than-word length text, to transform the sentences into a continuous space so that we can treat them as class distributions and measure their similarity.

5.1 Classification Accuracy

If a good heuristic is the one that generates "hard to distinguish" examples, the dataset augmented using **ONE** should train a better classifier than **ALL** for the sentiment analysis task, and the verbosity classifier trained on **NMN** should outperform the classifiers trained on **SYN** and **DUP**.

Table 3 and Table 4 show the classification accuracy of the neural models trained on different augmented datasets for sentiment and verbosity prediction tasks, and as we expected, heuristics that intuitively generate "harder to distinguish" examples are more suitable for the prediction task and trained a better classifier on both tasks:

Sentiment Classification Accuracy:

$$ACC(ONE) > ACC(ALL)$$

Verbosity Classification Accuracy:

$$ACC(NMN) > ACC(SYN) > ACC(DUP)$$

Dataset		Model	ACC	KLD	
				USE	BERT
Label-Preserving	RS	LSTM	.943	16.78	5.33
		CNN	.966		
		AVG	.954		
	SR	LSTM	.941	19.84	7.27
		CNN	.962		
		AVG	.951		
	RI	LSTM	.936	24.11	9.84
		CNN	.944		
		AVG	.940		
	RD	LSTM	.930	23.99	12.15
		CNN	.938		
		AVG	.934		
Non-Label-Preserving	ALL	LSTM	.683	26.26	13.97
		CNN	.716		
		AVG	.699		
	ONE	LSTM	.808	17.41	9.90
		CNN	.822		
		AVG	.815		

Table 3: Sentiment Classification Accuracy and Difference between Augmented Positive and Negative Distributions

Dataset		Model	ACC	KLD	
				USE	BERT
Non-Label-Preserving	DUP	LSTM	.393	14.86	15.90
		CNN	.442		
		AVG	.417		
	SYN	LSTM	.526	10.23	12.99
		CNN	.551		
		AVG	.538		
	NMN	LSTM	.692	8.91	7.77
		CNN	.738		
		AVG	.715		

Table 4: Verbosity Classification Accuracy and Difference between Augmented Verbose and Concise Distributions

These observations suggest that an augmented dataset generated from a heuristic that produces "harder to distinguish" examples for different classes could train a better classifier (**Q1**).

Since label-preserving heuristics do not change the class label of the samples, the extent to which "hard to distinguish" examples can be generated rely heavily on their existence in the original data. Thus, we cannot intuitively predict which label-preserving heuristic might be a better choice, however, in Section 5.2, we further study whether our purposed heuristic evaluation approach is applicable to label-preserving heuristics.

5.2 Augmented Distribution Difference

To investigate the extent to which "hard to distinguish" examples might be quantified as a difference between the distribution of the augmented samples of different classes, we first encode the augmented sentences into a continuous high dimensional vector space; then, we computed the difference between the distribution of the augmented samples of different classes as the divergence from high dimensional representation of one class to another.

For the sentiment analysis task, we computed the difference between augmented positive and negative distribution as follow, where E is either BERT or USE encoders, and $positive$ and $negative$ indicate augmented positive and negative examples respectively:

$$D_{KL}(E(positive)||E(negative))$$

The distribution difference for the verbosity analysis task is calculated as follow, where $concise$ and $verbose$ indicate augmented concise and verbose examples respectively:

$$D_{KL}(E(concise)||E(verbose))$$

It must be noted that since there is no correspondence between the augmented examples of different classes, we computed the difference as the average KL-Divergence over mini-batches of the size 64 samples from the shuffled augmented dataset for 10 epochs (the same batch and epoch values used for training LSTM and CNN models).

Table 3 shows the distribution difference between augmented *positive* and *negative* samples for the sentiment analysis task. As shown, although the average classification accuracy of models trained on label-preserving heuristics are only marginally different, the divergence between distributions of augmented examples with positive and negative sentiments are following the reverse order for both

BERT and USE representations, with one exception for USE representation of RI compared to RD:

– Label-Preserving Heuristics –

Sentiment Classification Accuracy:

$$ACC(RS) > ACC(SR) > ACC(RI) > ACC(RD)$$

Positive Distribution vs. Negative Distribution:

$$KLD(RS) < KLD(SR) < KLD(RI) < KLD(RD)$$

Since the non-label-preserving heuristics apply significant semantic changes to the original samples to change its class label, it is expected that the choice of heuristic should have a more noticeable impact on the classification accuracy compared to the augmentation using label-preserving heuristics. We also observe the same results for non-label-preserving heuristics: augmented dataset with higher classification accuracy has lower divergence between distributions of their positive and negative examples:

– Non-Label-Preserving Sentiment Heuristics –

Sentiment Classification Accuracy:

$$ACC(ONE) >> ACC(ALL)$$

Positive Distribution vs. Negative Distribution:

$$KLD(ONE) < KLD(ALL)$$

Table 4 shows the distribution difference between augmented *concise* and *verbose* samples for the verbosity prediction task. Here, similar to the sentiment analysis task, we observe that the divergence between distributions of augmented concise and verbose examples are following the reverse order of classification accuracy for both BERT and USE representations:

– Non-Label-Preserving Verbosity Heuristics –

Verbosity Classification Accuracy:

$$ACC(NMN) > ACC(SYN) > ACC(DUP)$$

Verbose Distribution vs. Concise Distribution:

$$KLD(NMN) < KLD(SYN) < KLD(DUP)$$

Encoding		**KLD**	**Classification**	
USE	**BERT**		**LSTM**	**CNN**
33.2s	92.8s	13.4s	2773s	878s
AVG: 63s				
Overall: 76.4s			**AVG:** 1825.5s	

Table 5: Execution Time of Our Proposed Heuristic Suitability Evaluation Approach Compared to the Classifier Training Time

These observations may indicate that the extent to which a heuristic might generate "hard to distinguish" examples could be quantified as the difference (divergence) between the distribution of augmented examples in different classes (**Q2**).

5.3 Computational Efficiency

Now that we have investigated the role of "hard to distinguish" examples in the success of training a classifier (**Q1**) and how to quantify that (**Q2**), it is time to evaluate the computational efficiency of our purposed approach to see how practical it is compared to training a separate classifier for each augmented dataset and pick the best performing one(s) (**Q3**).

To investigate this, we calculated the time for encoding the augmented examples into continuous space and the time requires for computing the KLD and compared them with the time required for training a classifier on an augmented dataset.

Table 5 shows the average execution time of our proposed approach for evaluating the suitability of different data augmentation heuristics and training neural classifiers on augmented datasets. Reported numbers are averaged over sentiment and verbosity prediction tasks for all augmented datasets. Encoding is a one-time process for each augmented dataset, and numbers reported under KLD and Classification columns are the overall execution time after 10 epochs of training on an NVIDIA Tesla P100 GPU.

We observed that encoding and divergence calculation times only depend on the number of samples and the classification task and choice of heuristic is not affecting the execution times. We also observed that the training time for both LSTM and CNN also highly depends on the number of training samples, and changing tasks and augmented dataset only slightly change the training time (standard deviation of 9.4s and 6.8s, respectively).

Execution times are showing that our proposed heuristic evaluation approach is about 25 times faster than training a classifier; this may suggest that our proposed approach could be a low-cost alternative solution for assessing the suitability of the heuristic strategies for augmenting training dataset for different classification tasks, especially for complex training scenarios when training many classifiers on different augmented dataset might not be computationally practical (**Q3**).

6 Conclusion

This paper presents an approach for evaluating the suitability of augmentation heuristics for classifications task via "hard to distinguish" example generation capacity of the heuristics through analyzing the difference of class distribution of the augmented examples.

Experimental results suggest our proposed heuristic evaluation approach could be a low-cost yet effective way of measuring the suitability of an augmented heuristic for a classification task.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. This material is based upon work supported by the National Science Foundation under Grant Number 1735752.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Not Enough Data? Deep Learning to the Rescue! In *AAAI*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Céspedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil Google Research Mountain View. 2018. Universal Sentence Encoder. *Computing Research Repository*, arXiv:1803.11175.

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. In *EMNLP*.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a Large Annotated Corpus of Learner English: The NUS Corpus of Learner English. In *SIGEDU*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data Augmentation for Low-Resource Neural Machine Translation. In *NAACL*.

Benoît Frénay and Michel Verleysen. 2014. Classification in the Presence of Label Noise: a Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845.

Hongyu Guo, Yongyi Mao, and Richong Zhang. 2019. Augmenting Data with Mixup for Sentence Classification: An Empirical Study. *Computing Research Repository*, arXiv:1905.08941.

Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. 2018. Using Trusted Data to Train Deep Networks on Labels Corrupted by Severe Noise. In *NIPS*.

Minqing Hu and Bing Liu. 2004. Mining and Summarizing Customer Reviews. In *KDD*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward Controlled Generation of Text. In *ICML*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. In *NAACL*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. TinyBERT: Distilling BERT for Natural Language Understanding. *Computing Research Repository*, arXiv:1909.10351.

Omid Kashefi, Andrew T Lucas, and Rebecca Hwa. 2018. Semantic Pleonasm Detection. In *NAACL*.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *EMNLP*.

Sosuke Kobayashi. 2018. Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations. In *NAACL*.

S. Kullback and R. A. Leibler. 1951. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data Augmentation using Pre-trained Transformer Models. *Computing Research Repository*, arXive: 2003.02245.

Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *ICML*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent Neural Network for Text Classification with Multi-Task Learning. In *IJCAI*.

Franco M. Luque. 2019. Atalaya at TASS 2019: Data Augmentation and Robust Embeddings for Sentiment Analysis. In *TASS: Workshop on Sentiment Analysis at SEPLN*.

George A. Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Siyuan Qiu, Binxia Xu, Jie Zhang, Yafang Wang, Xiaoyu Shen, Gerard de Melo, Chong Long, and Xiaolong Li. 2020. EasyAug: An Automatic Textual Data Augmentation Platform for Classification Tasks. In *The Web Conference*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *ACL*.

Connor Shorten and Taghi M. Khoshgoftaar. 2019. A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6(1):1–48.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *EMNLP*.

Andrew Trask, Phil Michalak, and John Liu. 2015. sense2vec - A Fast and Accurate Method for Word Sense Disambiguation In Neural Word Embeddings. *Computing Research Repository*, arXiv:1511.06388.

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 *EMNLP*.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *EMNLP*.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. 2019. Unsupervised Data Augmentation for Consistency Training. *Computing Research Repository*, arXive: 1904.12848.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond Empirical Risk Minimization. In *ICLR*.

Xiang Zhang, Junbo Zhao, and Yann Lecun. 2015. Character-level Convolutional Networks for Text Classification. In *NIPS*.