Towards Reversal-Based Textual Data Augmentation for NLI Problems with Opposable Classes

Alexey Tarasov
Informatica CLAIRE
1 Windmill Lane
Dublin 2, Ireland
atarasov@informatica.com

Abstract

Data augmentation methods are commonly used in computer vision and speech. However, in domains dealing with textual data, such techniques are not that common. Most of the existing methods rely on rephrasing, i.e. new sentences are generated by changing a source sentence, preserving its meaning. We argue that in tasks with opposable classes (such as "Positive" and "Negative" in sentiment analysis), it might be beneficial to also "invert" the source sentence, reversing its meaning, to generate examples of the opposing class. Methods that use somewhat similar intuition exist in the space of adversarial learning, but are not always applicable to text classification (in our experiments, some of them were even detrimental to the resulting classifier accuracy). We propose and evaluate two reversal-based methods on an NLI task of recognising a type of a simple logical expression from its description in plain-text form. After gathering a dataset on MTurk, we show that a simple heuristic using notion of negating the main verb has potential not only on its own, but that it can also boost existing state-of-the-art rephrasing-based approaches.

1 Introduction

In natural language processing (NLP), the high performance of a machine learning solution often depends on the quality and quantity of training data, but its collection is not always trivial (Wei and Zou, 2019). For some tasks, such as sentiment analysis, extensive corpora already exist, and can be used at least as a starting point. However, in the area of natural language interfaces (NLIs), tasks are often very specific, and training data often has to be collected from scratch, which can be a major limiting factor.

Data augmentation is a family of techniques that take an initial dataset (often limited in size) and automatically generate more examples, with the hope that they will introduce some realistic variability, thus reducing reliance on possibly costly and time-consuming data collection.

In some areas, good data augmentation approaches are available and widely used. For instance, to augment an image, one can scale it (Lecun et al., 1998) or crop it (Szegedy et al., 2015). Data augmentation is not limited just to computer graphics. For instance, Lee et al. (2005) proposed an augmentation approach for a more narrow area of schema matching. However, in NLP, usage of data augmentation is somewhat limited (Kobayashi, 2018). Currently, it is a very active area of research, and multiple different approaches are used. Almost all of them, however, rely on the same fundamental principle: to augment a sentence, generate sentences that have the same meaning, but use slightly different phrasing.

One of the limitations of such "rephrasing-based" approaches is that when applied to a sentence labelled with a certain class label, they can only generate sentences that belong to the same class. For instance, Wang and Yang (2015) proposed to replace certain words in a sentence with their synonyms. Applying such an approach to a positive review "This is a good movie" can result in "This is a fantastic movie", "This is a good film" and so on, but all of them will still be positive reviews. In this paper, we propose a different approach: instead of preserving the meaning of the sentence intact, we attempt to reverse its meaning, so that the "opposite" class can also benefit from data augmentation. To the best of our knowledge, limited attention has been paid to investigating such approaches, with the exception of few papers in the domain of adversarial learning (Jia and Liang 2017, Niu and Bansal 2018).

Reversing polarity of a free-text snippet, such as a movie review or a tweet, can be challenging and

Proceedings of the First Workshop on Natural Language Interfaces, pages 11–19

https://doi.org/10.18653/v1/P17

Pair	Classes	Sentence example	Expression example	Count
Particular value	=	X must be ten characters	$LENGTH(X) = 10$	93
	!=	X can not be 2	$X \neq 2$	126
Inequality	$<$	X shall not exceed 0	$X \leq 0$	239
	$>$	X is longer than 10	$LENGTH(X) > 10$	320
Null check	IS NULL	X is blank	$X\ IS\ NULL$	36
	NOT NULL	X cannot be missing	$X\ IS\ NOT\ NULL$	37
			Total	**851**

Table 1: Details of the dataset collected on MTurk for classifying the type of a logical expression, based on its textual description. Classes are divided into three pairs, and classes in the same pair are exact opposites of each other. I.e. negating a sentence belonging to one class should almost always result in a sentence belonging to the other class in the same pair.

even impossible, if done automatically. In this preliminary investigation we are limiting ourselves to short, relatively technical sentences, which are very common in NLI space. Using reversal of meaning for data augmentation can be useful for the tasks which have "naturally opposable" classes, for instance, in voice assistant dealing with opposite actions ("Set the alarm for 7am" vs. "Unset the alarm for 7am" or "Turn the volume down" vs. "Turn the volume up").

Our motivating example is understanding a logical statement, expressed as a short sentence in natural language. We were dealing with the first step of this task, recognising one of six statement types, which constitute three "naturally opposable" pairs. One of such pairs is "Equal"/"Not equal". By taking an example that belongs to "Equal" class such as "X is equal to two", we can reverse its meaning to "X is not equal to two", and in such way augment "Not equal" class as well. As far as we know, it is the first attempt to use such a technique for textual data augmentation in text classification tasks.

The main contributions of this paper are the following:

1. Dataset for a task of classifying type of logical statement, gathered on MTurk platform and consisting of 851 sentences, belonging to six classes.

2. Preliminary investigation of two approaches—*Main verb inversion* and *Adjective and adverb antonymy*—for performing textual data augmentation for tasks with "naturally opposable" classes. Our evaluation strongly suggests that *Main verb inversion* can increase the performance of a classifier and can also generate data, which cannot be acquired by using state-of-the-art rephrasing-based approaches.

The rest of the paper is structured as follows. In Section 2, we outline the task and describe data collection. Section 3 follows with an overview of related research. Section 4 describes our approaches in detail, and is followed by Section 5 outlining our evaluation methodology. Experimental results are discussed in Section 6, while Section 7 concludes the paper and proposes directions for future work.

2 Task and data collection

The motivational application behind our experiments is the task of translating a sentence in natural language into a logical expression that otherwise would have to be specified using complex and not very user-friendly language, such as SQL (or a formula editor in a tool such as Excel). This task can be solved in different ways, and some of them rely on inferring statement type (or its intent) as the first step.

We were concerned with six such types that were divided into three pairs, where classes in each pair are exact opposites. It means that if the meaning of a sentence from one pair class is reversed, we get a sentence that belongs to the second class in the same pair. An overview of all classes with examples is given in Table 1. We were interested in 31 logical expressions, some examples of which are listed in *Expression example* column.

A variety of free-text phrases can be used to describe the same logical expression. For instance, "$X\ IS\ NULL$" can be phrased as "X is blank", "X should not be empty", "X should be populated in all cases" and so on. We wanted to get a reliable expression type detector that would be robust to such examples of language variability.

We collected the data on Amazon Mechanical Turk platform. A single HIT presented a logical statement (such as "$LENGTH(X) > 0$" or "$X < 0$"), and a turker had to provide four significantly different phrases, that describe the logic in natural language.

First, we performed a quick pilot collection to make sure that our instructions and setup make sense, involving only Master turkers. No problems were encountered, so we proceeded with the main data collection, involving only turkers with acceptance rate on previous tasks of at least 60%. Each HIT was completed by at least 10 workers, and we paid \$0.20 for every successful completion[1].

We had to reject about 35% of submitted HITs due to imprecise or spammy answers. Additionally, some of the accepted answers were same, for instance, expression "$X\ IS\ NULL$" was often transcribed as "X must be empty". After eliminating such duplicates, we got a dataset of 851 sentences, achieving good balance inside each pair of classes. The resulting dataset is publicly available online[2].

Next section describes existing data augmentation methods. We discuss methods that try to preserve the meaning of the original sentence (majority of them), as well as few exceptions.

3 Related research

Textual data augmentation was used in a number of different areas, including text classification (Wei and Zou, 2019), textual (Yu et al., 2018) or visual question answering (Kafle et al., 2017), reading comprehension systems (Jia and Liang, 2017) and machine translation (Fadaee et al., 2017, Gao et al., 2019). In fields like computer vision or speech, there are well-established methods that work across many applications, such as introducing random noise into an audio clip or cropping an image. However, according to Kobayashi (2018), in the field of NLP, it is very difficult to come up with an approach that would be easily applicable to various tasks. It can explain existence of a variety of augmentation strategies, most of which can be broadly categorised into two big categories: strategies that rephrase a sentence preserving its original meaning, and strategies that deliberately change the meaning of a sentence.

[1]Median completion time for a HIT was 204 seconds.

[2]https://github.com/alexey-tarasov-irl/acl2020_nli_workshop

3.1 "Preserve the meaning" augmentation

All these approaches attempt rephrasing a sentence, while keeping its original semantics. It can be done in a variety of ways:

1. **Generative approaches** employ a deep generative model (Bowman et al., 2016, Hu et al., 2017) to generate sentences with desired attributes from a continuous space. According to Wu et al. (2019), they often generate sentences that aren't readable and do not correspond to the desired class labels.

2. **Random permutation**: new sentences are generated by applying a very simple randomised heuristic to a source sentence, such as deleting (Iyyer et al., 2015, Wei and Zou, 2019, Xie et al., 2017) or swapping words (Artetxe et al., 2018, Niu and Bansal, 2018).

3. **Backtranslation**: a source sentence is translated into a different language (pivotal language), and then the result is translated back into the language of the source sentence (Sennrich et al., 2016, Yu et al., 2018). For example, using German as pivotal language can result in a sequence like "X is lower than 0" ➔"X ist kleiner als 0" ➔"X is less than 0".

4. **Synonym usage**: very commonly used strategies which usually involve selecting a word in a source sentence and then replacing it with a synonym. Sometimes words to be replaced are chosen randomly (Wei and Zou, 2019), while other researchers impose some limitations, such as only changing the headword (Kolomiyets et al., 2011). Ways to find synonyms range from relatively low-tech, such as using WordNet synsets (Wei and Zou, 2019, Zhang et al., 2015), to much more advanced approaches involving word embeddings (Wang and Yang, 2015) or bi-directional deep learning models (Kobayashi, 2018, Wu et al., 2019).

All these strategies have their own advantages and drawbacks, so many text augmentation approaches use a hybrid strategy, such as Easy Data Augmentation (EDA) by Wei and Zou (2019). It uses a combination of random permutation strategies and synonyms. One of its parameters n_{aug} is the maximum number of new sentences to generate per each source sentence. It is a recent and simple

algorithm that embodies a lot of very commonly used text augmentation techniques. The results of Wei and Zou (2019) indicate that it achieves performance similar to much more complex algorithms, but is much quicker and doesn't rely on external dataset or language models.

3.2 "Change the meaning" augmentation

In some applications, it might be beneficial to deliberately change the meaning of a sentence, instead of preserving it. It is often done in the domain of *adversarial learning* for two purposes:

- **Purpose #1:** investigate whether a model can be confused by deliberately misleading data.
- **Purpose #2:** make the model robust against such adversarial attacks.

Niu and Bansal (2018) used dialogue models to predict the next turn, based on current context. They used reversal of meaning as a way to strengthen the model, making it more robust to changes in phrases that are very subtle yet change the meaning completely. They investigated two strategies[3]:

1. **Add negation**: for the first verb[4] in the sentence (going left to right), that doesn't have an outgoing `neg` arc in the dependency graph, the negation is artificially added. If no negation is detected, the original sentence is returned as augmented sentence (i.e. the approach always returns one sentence per each source sentence, effectively doubling the size of the original set).

2. **Antonym**: all words in the original sentence are picked one by one, going left to right[5]. For each such word, all synsets that have it are extracted, and all words in those synsets are explored for antonyms. The original word in the sentence is replaced by a random antonym from that set, and the process is over once one word in the original sentence is successfully replaced[6]. Thus, this strategy can increase the size of the dataset by 100% at the most.

Both approaches have been tested in four different conditions, varying datasets used for training and testing:

1. **Original train, original test**: the performance of the system on original test data, when no augmentation was performed.

2. **Original train, augmented test**: investigation into whether the system is robust enough to handle adversarial inputs (purpose #1 mentioned above).

3. **Augmented train, augmented test**: experiment to prove that augmenting training data with adversarial sentences makes the system more robust (purpose #2 above).

4. **Augmented train, original test**: checking whether augmenting training data helps with the model doing better on "usual", non-adversarial data.

In our setup, we were looking into augmenting training data, in order to make the classifier perform better on original data (Condition #4). However, Niu and Bansal (2018) failed to achieve statistically significant increase in #4, which suggests that both *Add negation* and *Antonym* might not be beneficial for our task.

Another augmentation approach that deliberately changes the meaning was proposed and evaluated by Jia and Liang (2017), for question answering systems. Their goal was not to make them more robust, but to show how easy it can be to confuse them. For each question in a corpus, they attempted to generate an adversarial sentence by replacing nouns and adjectives with antonyms from WordNet (very similar to *Antonym* by Niu and Bansal 2018), and change named entities and numbers to the nearest word in GloVe word vector space. For instance, "What ABC division handles domestic television distribution?" would become "What NBC [ABC replaced by a nearby word NBC] division handles

[3]The original paper offers a limited description of the approaches, so we relied on the accompanying source code located at `https://github.com/WolfNiu/AdversarialDialogue`.

[4]TreeBank POS. tags `VB`, `VBD`, `VBG`, `VBN`, `VBP`, `VBZ`.

[5]The paper states that it happens only for verbs, nouns, adjectives and adverbs, but the accompanying code actually just goes through all words, using part-of-speech only while searching for suitable synsets.

[6]In the original paper, each selected antonym had to also be present somewhere in the training corpus, usually, in a different sentence. It might be feasible if a large dataset is available, but in our case it would have resulted in almost no new sentences introduced. This is why we relaxed this condition in our experiments, and let *Antonyms* use antonyms, even if they are not present in any sentence in the dataset.

foreign [WordNet antonym to 'domestic'] television distribution?" Then, they generated a fake answer, which couldn't possibly be the right answer for this adversarial question. The experiments in the paper show that question answering systems can be easily fooled and often would produce that fake answer.

Work by Kaushik et al. (2020) serves as good evidence that inverting sentence meaning can be beneficial for text classification, both in terms of overall accuracy and robustness to adversarial data. However, instead of using automated rules, they asked crowdsourced workers to revert the meaning of a movie review, so that the document still remains coherent and a minimum of modifications is made.

To the best of our knowledge, automated methods from this subsection have never been applied to text classification tasks. Next section covers two approaches we propose in this paper.

4 Our approaches

Approach by Jia and Liang (2017), described in the previous section, is not directly applicable to our task, as replacing nouns and numbers won't result in effective negation of sentences in our dataset. The only useful aspect—antonyms of adjectives—is also present in paper by Niu and Bansal (2018), which also contains other useful insights.

We propose two approaches, that enhance *Add negation* and *Antonym* by Niu and Bansal (2018) described in Section 3.2:

1. Main verb inversion (enhancement of *Add negation*): similarly to *Add negation*, we add negation if it's not there, but in addition we also remove it if it is present.

2. Adjective and adverb antonymy (enhancement of *Antonym*): in our experience, *Antonym* often produced sentences that did not make grammatical sense, sentences with grammatical mistakes or sentences that were not proper negations of the original sentence. Table 2 provides a few examples of such issues. The most common root causes of such issues are the following:

1. Some antonyms selected from WordNet belonged to the wrong synset of the verb that was picked for replacement. For instance, the verb "can" is not only a modal verb, but is also an informal US expression for removing someone from their job. This is why for "can", in sentences like "X can be negative", *Antonym* picked the synset with the words "fire" and "give notice". The synset has "hire" as antonym, and that is the word that made its way into the augmented sentence ("X hire be positive"), which didn't make sense. Similar behaviour was observed for other common words such as "will".

2. When the adjective is replaced, its comparative/superlative form is not preserved (e.g. if "lower" is selected for replacement, it's replaced with "high", not "higher").

3. Due to its left-to-right nature, *Antonym* often picks an improper word for replacement. For instance, in sentence "X is no bigger than 2", it can't find any antonyms for "X" and "is", but picks "yes" as an antonym to "no", which results in "X is yes bigger than 2".

It might seem that both *Add negation* and *Antonym* by Niu and Bansal (2018) can result in a dataset of much lower quality, compared to the original. However, their intention was to pollute a sentence enough to change its meaning in some way, or make it incomprehensible, to deliberately confuse dialogue systems. However, we are trying to achieve something much more complicated. In our task, it's not enough to break the meaning of a sentence. We aim for coming up with valid sentences that properly negate the source sentence.

This is why our main assumption is that not producing a sentence at all is better than producing a sentence that doesn't make sense. We only replace adverbs and adjectives, and only if it's the only adverb/adjective in the sentence, and it is directly connected to the root. For each such sentence, we produce a new sentence for each antonym found in WordNet, and preserve comparative/superlative adjective forms.

5 Experiment methodology

The goal of our experiments was to investigate whether reversal-based data augmentation can boost accuracy in text classification tasks. We were concerned with two questions:

1. Is there a benefit in using reversal-based approaches, compared to not using data augmentation at all (Experiment #1)?

2. Can reversal-based approaches be a useful addition to already existing rephrasing-based augmentation techniques (Experiment #2)?

Source sentence	A correct augmentation	Augmentation by *Antonyms* (Niu and Bansal, 2018)
X can be negative	X can be positive	X **hire** be positive
X must be lower than 0	X must be higher than 0	X must be **high** than 0
X is no bigger than 2	X is no smaller than 2	X is **yes** bigger than 2

Table 2: Examples where *Antonym* approach (Niu and Bansal, 2018) provided highly incorrect sentences (bold text highlights mistakes that were made). See Section 4 for a detailed discussion.

5.1 Experiment #1: reversal-based augmentation vs. no augmentation

We used a CNN[7] proposed by Kim (2014), which is commonly used for evaluating data augmentation approaches (Park and Ahn 2019, Wei and Zou 2019). We ran it separately for each augmentation approach, conducting 5-fold cross validation augmenting only training data, leaving test partition intact. We allowed the training algorithm to run for 50 epochs of batch gradient descent (batch size = 64). A single sentence encoded using 50-dimensional GloVe embeddings (Pennington et al., 2014) was the input to the CNN (*Sentence example* column from Table 1).

We benchmarked both of our reversal-based augmentation approaches, proposed in Section 4, against the following baselines:

1. No augmentation
2. Add negation (Niu and Bansal, 2018)
3. Antonym (Niu and Bansal, 2018)

Our dataset was well-balanced by design; however, in many real-life applications it might not be the case. Potentially, augmentation approaches can be especially beneficial if applied to underrepresented classes. To simulate such a condition, in each training split, before performing augmentation, we artificially removed a fraction of instances belonging to classes ">", "!=" and "IS NULL" (leaving 25%, 50%, 75% intact) and only then performed data augmentation. We also tested the condition when no such undersampling was performed, and 100% of instances of those classes were retained.

Each experiment was conducted twenty-five times to counteract multiple random factors present in this setup, with average macro F1 reported as performance metric. Wilcoxon test was used to determine statistical significance of differences in performance.

5.2 Experiment #2: reversal-based on top of rephrasing-based

Even if reversal-based augmentation approaches are useful on their own, it is possible that they can't bring additional benefit if a rephrasing-based approach has already been used. To investigate this, overall, we followed the same methodology as in Experiment #1. Here is how we derived training data for each experimental condition.

In each cross-fold validation split, we applied EDA to training data[8], with default values of all parameters (including $n_{aug} = 9$). Then we applied an augmentation approach to the same training data, and merged its results with those coming from EDA. As before, test splits were not augmented, and we reported macro F1 score, averaged across twenty five experiments.

If reversal-based augmentation approaches work, CNN performance on such sets is expected to be higher than on just EDA on its own. However, performing EDA with $n_{aug} = 9$ and using it as a baseline would not have been fair: EDA in conjunction with any other augmentation approach is expected to result in a bigger dataset than EDA on its own. To make sure that any possible differences in accuracies cannot be attributed to this, we experimentally found a value of $n_{aug} = 11$, which guaranteed that the EDA baseline had a dataset, which is larger than any other set, resulting from applying augmentation.

6 Results

Results of Experiment #1 are given in Table 3. Both approaches by Niu and Bansal (2018) exposed very unreliable performance. In total, eight experiments involved them (four undersampling scenarios mul-

[7]Parameters (tuned on the full dataset): three convolution layers with window sizes of three, four and five (128 filters each); dropout rate of 0.25; Adam optimizer ($\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$).

[8]Using code from https://github.com/jasonwei20/eda_nlp

Augmentation approach	Dataset size	Relative change	Artificial undersampling, F1			
			25%	50%	75%	100%
No augmentation (baseline)	851	—	27.78	47.79	53.69	56.05
Antonym (Niu and Bansal, 2018)	1,591	+87%	**42.24▲**	49.25	51.41▼	54.16▼
Adjective and adverb antonymy (ours)	1,173	+38%	32.20▲	46.90	53.33	54.82
Add negation (Niu and Bansal, 2018)	1,702	+100%	45.20▲	50.29▲	52.87	53.18▼
Main verb inversion (ours)	1,517	+78%	**55.32▲**	**61.59▲**	**62.52▲**	**63.57▲**

Table 3: Results of Experiment #1: no augmentation baseline is compared to four reversal-based augmentation approaches. If an approach was significantly better than the baseline and another approach in the pair, its F1 is given in bold. ▼/▲ denote when approach was significantly worse/better than the baseline (Wilcoxon text, $p < 0.01$). *Relative change* indicates how big was the resulting dataset after augmentation (e.g. +100% means that it was twice the size of the original set).

Augmentation approach	Dataset size	Relative change	Artificial undersampling, F1			
			25%	50%	75%	100%
EDA (Wei and Zou, 2019), $n_{aug} = 11$	10,212	+1,100%	51.72	64.20	68.42	70.02
EDA (Wei and Zou, 2019), $n_{aug} = 9$						
+ Add negation (Niu and Bansal, 2018)	9,361	+1,000%	53.45	63.75	68.51	69.82
+ Main verb inversion (ours)	9,176	+978%	**57.65▲**	**67.36▲**	**70.46▲**	**72.06▲**

Table 4: Results of Experiment #2: EDA baseline is compared to EDA in conjunction with two verb-negation-oriented approaches. If an approach was significantly better than the others, its F1 is given in bold. ▲ denotes approaches that were significantly better than EDA baseline (Wilcoxon test, $p < 0.01$). *Relative change* indicates how big was the resulting dataset after augmentation.

tiplied by two approaches). In three out of eight, they worsened the performance of the classifier, in another two they didn't make any difference, and only in remaining three they made it significantly better. In contrast, both *Main verb inversion* and *Adjective and adverb antonymy* improved the performance significantly in more than half of the experiments, compared to no augmentation baseline. Additionally, *Main verb inversion* consistently showed results that were significantly better than both the baseline and *Add negation*. Usage of *Adjective and adverb antonymy* never harmed the performance, but it also rarely improved it. This is why we dropped both *Antonym* and *Adjective and adverb antonymy* from Experiment #2.

In Experiment #2 (Table 4), *Add negation* (Niu and Bansal, 2018) failed to improve the results of plain EDA significantly. At the same time, our *Main verb inversion* was significantly better than EDA in all experiments, which strongly suggests that it can derive data, not easily accessible by a rephrasing-based approach, such as EDA.

Overall, the results allow us to recommend *Main verb inversion* as a promising direction for textual data augmentation in classification tasks. Compared to no augmentation baseline, it improved macro F1 by 7.53–27.54pp (or by 13.42–99.15%), depending on how balanced the dataset is. *Main verb inversion* was especially beneficial when used to imbalanced datasets. When used on top of EDA, *Main verb inversion* was able to improve the F1 score by 2.04–5.93pp (or 2.92–11.47%).

7 Conclusions and future work

In this paper we addressed a problem of detecting a type of a logical statement from its textual description. We gathered our own task-specific dataset on MTurk, and then tried to boost the accuracy of the resulting classifier by applying textual data augmentation. We used two approaches (*Add negation* and *Antonym*) by Niu and Bansal (2018) from the current research in adversarial learning. In our experiments, neither of them could show stable improvement over a no-augmentation baseline. Even worse, often they had a detrimental effect on the macro F1 score of the resulting classifier.

We proposed two approaches: *Adjective and adverb antonymy* and *Main verb inversion*. The former failed to expose any benefit in our experiments; however, the latter consistently performed

better than baselines. Despite its simplicity, it could achieve significantly better results than the no-augmentation baseline (improvement was ranging from 13% to 99%). *Main verb inversion* also showed the capability to introduce information into the training set, which is not available to state-of-the-art rephrasing-based approaches. A combination of EDA and *Main verb inversion* was 3–11% better than EDA on its own.

Main verb inversion showed promising performance, but it might be difficult to use it to negate sentences, expressed in less technical language (such as tweets or movie reviews). This is why enhancing its capabilities to other application areas seems to us like the primary direction for future work. It's unlikely that it can become a widely used cross-application approach in its current shape, but we hope that our findings will be thought-provoking for researchers who want to pursue reversal-based augmentation further. One of the possible improvements is to rely on a black-box model instead of heuristic rules (e.g. a sequence-to-sequence model that takes a sentence and returns the corresponding inverted sentence).

While verb negation/inversion showed good performance, approaches based on directly seeking antonyms proved to be ineffective. It might be interesting to investigate the reasons of their failure in more detail.

Acknowledgements

The author would like to thank Informatica CLAIRE team, especially, Bojan Furlan, Igor Balabine, AnHai Doan and Darshan Joshi for discussions, support and encouragement. Special thanks go to Awez Syed who sparkled author's interest in data augmentation. We would also like to express the deepest appreciation to Ashlee Bailey Brinan for her editorial comments that greatly improved the paper.

We thank three anonymous reviewers for their comments, and workers on MTurk for completing our tasks and making this work possible.

References

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In *Procs of ICLR*.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Procs of CoNLL*, pages 10–21.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-Resource neural machine translation. In *Procs of ACL*, pages 567–573.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft Contextual Data Augmentation for Neural Machine Translation. In *Procs of ACL*, pages 5539–5544.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *ICML*, volume 4, pages 2503–2513.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daume III. 2015. Deep Unordered Composition Rivals Syntactic Methods for Text Classification. In *Procs of ACL*, pages 1681–1691.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 2021–2031.

Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan. 2017. Data Augmentation for Visual Question Answering. In *Procs of Natural Language Generation conference*, pages 198–202.

Divyansh Kaushik, Eduard Hovy, and Zachary C. Lipton. 2020. Learning the Difference that Makes a Difference with Counterfactually-Augmented Data. In *Procs of ICLR*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Procs of EMNLP*, pages 1746–1751.

Sosuke Kobayashi. 2018. Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations. In *Procs of NAACL*, pages 452–457.

Oleksandr Kolomiyets, Steven Bethard, and Marie Francine Moens. 2011. Model-portability experiments for textual temporal analysis. In *Procs of ACL-HLT*, volume 2, pages 271–276.

Yann Lecun, Leon Bottou, Yoshua Bengio, Patrick Ha, and Patrick Haffner. 1998. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11).

Yoonkyong Lee, Mayssam Sayyadian, Anhai Doan, and Arnon S. Rosenthal. 2005. ETuner: Tuning schema matching software using synthetic scenarios. In *Procs of VLDB*.

Tong Niu and Mohit Bansal. 2018. Adversarial oversensitivity and over-stability strategies for dialogue models. In *Procs of CoNLL*, CoNLL, pages 486–496.

Dongju Park and Chang Wook Ahn. 2019. Self-supervised contextual data augmentation for natural language processing. *Symmetry*, 11(11).

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Procs of EMNLP*, pages 1532—-1543.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Procs of ACL*, pages 86–96.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Procs of IEEE CVPR*.

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 *Procs of EMNLP*, pages 2557–2563.

Jason Wei and Kai Zou. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *Procs of EMNLP*, pages 6382–6388.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional BERT Contextual Augmentation. *Lecture Notes in Computer Science*, 11539 LNCS:84–95.

Ziang Xie, Sida I. Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y. Ng. 2017. Data noising as smoothing in neural network language models. In *Procs of ICLR*.

Adams Wei Yu, David Dohan, Minh Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. QaNet: Combining local convolution with global self-attention for reading comprehension. In *Procs of ICLR*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In *Procs of NIPS*, pages 649–657.