Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Data augmentation techniques in natural language processing

Lucas Francisco Amaral Orosco Pellicer [*], Taynan Maier Ferreira, Anna Helena Reali Costa

Data Science Center USP & Itaú (C^2D), Universidade de São Paulo, São Paulo, SP, Brazil

A R T I C L E   I N F O

Article history:
Received 9 February 2022
Received in revised form 24 June 2022
Accepted 2 November 2022
Available online 19 November 2022

Keywords:
Data augmentation
Natural language processing
Back-translation
Machine learning

A B S T R A C T

Data Augmentation (DA) methods – a family of techniques designed for synthetic generation of training data – have shown remarkable results in various Deep Learning and Machine Learning tasks. Despite its widespread and successful adoption within the computer vision community, DA techniques designed for natural language processing (NLP) tasks have exhibited much slower advances and limited success in achieving performance gains. As a consequence, with the exception of applications of back-translation to machine translation tasks, these techniques have not been as thoroughly explored by the wider NLP community. Recent research on the subject still lacks a proper practical understanding of the relationship between the various existing DA methods. The connection between DA methods and several important aspects of its outputs, such as lexical diversity and semantic fidelity, is also still poorly understood. In this work, we perform a comprehensive study of NLP DA techniques, comparing their relative performance under different settings. We analyze the quality of the synthetic data generated, evaluate its performance gains and compare all of these aspects to previous existing DA procedures.

1. Introduction

When building Machine Learning (ML) models to address supervised learning tasks, one has the objective to be able to predict unseen inputs based on previously seen inputs — i.e., the goal is to minimize the so called generalization error [1], or in other words, reduce overfitting. Many strategies have been developed in order to increase the generalization power of ML models [2]: dropout, batch normalization, transfer learning, pretraining, One-shot and Zero-shot learning are some of them. Data Augmentation (DA), the focus of this paper, is yet another strategy to reduce overfitting.

DA can be defined as any method for increasing the diversity of training examples without explicitly collecting new data [3]. In consonance with one of the most important results from Statistical Learning Theory, which states that discrepancy between training and generalization error diminishes with increasing training examples [1], DA has successfully been used in the ML and Deep Learning (DL) communities to synthetically inflate data for training and, as a result, obtain models with greater generalization power.

Since DA tackles the issue of overfitting from the training dataset itself, it is a general and task-agnostic approach, whose application varies from Image Processing [4,5] to Sound and Speech Recognition [6,7], from Time Series [8] to Natural Language Processing (NLP) [9,10]. Within the Computer Vision community, DA has been successfully used for several years now, being part of the training process of models responsible for some of the greatest achievements in Image Classification tasks, such as the AlexNet [11], All-CNN [12], and ResNet [13] models.

These remarkable accomplishments have led researchers to investigate the underlying theoretical principles governing DA, trying to shed some light into its relationship to aspects such as model learning process, decision surface, among others. These researches show that DA improves generalization by both increasing invariance and penalizing model complexity [14]. DA also can be considered a form of implicit regularization, closely related to explicit regularization techniques such as Weight Decay and Dropout. In fact, the works of [15,16] indicate that, under certain circumstances, DA and Dropout can be considered equivalent methods. Other studies, on the other hand, state that DA exhibit superior performance in comparison to explicit regularization methods [17].

More recently, state-of-the-art DA techniques in Image Processing have shifted from prior-knowledge-oriented handcrafted transformations to techniques that learn the augmentation transformations themselves. That is, instead of leveraging domain-specific knowledge to define the set of rules for generating artificial data, new methods *learn* the needed transformations. The development of an end-to-end approach, where DA NLP techniques are optimized to output artificial text that is best suited for the learning process of the final model seems a natural

[*] Corresponding author.
E-mail addresses: lucas.pellicer3394@gmail.com, lucas.pellicer@usp.br (L.F.A.O. Pellicer).

https://doi.org/10.1016/j.asoc.2022.109803

Nomenclature

BART	*Bidirectional and Auto-Regressive Transformers*
BERT	*Bidirectional Encoder Representations from Transformers*
BT	*Back-Translation*
cGAN	*Conditional GAN*
CNN	*Convolutional Neural Network*
DA	*Data Augmentation*
DeepBT	*Deep Back-Translation*
EDA	*Easy Data Augmentation*
ERM	*Empirical Risk Minimization*
GAN	*Generative Adversarial Net*
GPT	*Generative Pre-Trained NN*
LSTM	*Long short-term memory*
ML	*Machine Learning*
NLP	*Natural Language Processing*
NN	*Neural Network*
NMT	*Neural Machine Translation*
PPO	*Proximal Policy Optimization*
PPDB	*Paraphrase Database*
SA	*Sentiment Analysis*
SSMBA	*Self-Supervised Manifold Based Data*
T5	*Text-to-Text Transfer Transformer*
TREC	*Text Retrieval Conference*

and promising research path [18]. In summary, DA applications in image processing tasks have been an active area of research with new methods being proposed every year, some of them with highly promising results. AutoAugment [19] and Population Based Augmentation [20] are some of the most recent ones, just to name a few.

Despite unquestionable success in computer vision tasks, NLP research has not yet benefited as largely from DA systems. General NLP tasks and challenges are often characterized by the low – or often unsuccessful – usage of DA techniques. This trend seems to be changing in recent years though. When analyzing the solutions proposed for some of the SemEval Tasks over the period of 2017–2021, e.g., we observe the following:

1. SemEval-2017 Task 5: there is no mention to the use of DA methods by any of the participants [21];
2. SemEval-2018 Task 1: among 75 teams, only 2 teams acknowledge the use of some kind of DA procedure [22];
3. SemEval-2019 Task 5: within 74 participants, only one of them indicates using some kind of DA [23].
4. SemEval-2020: though both Task 9 and Task 4 have reported the use of DA among the top performing systems (XLP system and BUT-FIT/LMVE systems, respectively), these submissions account for just a fraction of all the competing participants.
5. SemEval-2021: whereas on Task 5 the majority of participating teams adopted some DA strategy, on Task 1 only one system (DeepBlueAI) is reported to have used DA methods [23].

Despite experiencing growing interest, we hypothesize that the still low adoption of DA techniques in NLP tasks and challenges results from the fact that NLP-specific DA methods have not been so successful as the ones used in the computer vision community, within which its presence is ubiquitous. The reason for that may be twofold: on the one hand, NLP-specific DA strategies have shown themselves difficult to develop. On the other hand, DA research has been still primarily focused on computer vision tasks.

To address this research gap and provide practitioners and researchers general guidelines on its use, we expand our previous work [24], conducting an in-depth investigation of some of the most important NLP DA techniques. We compare their output and relative performance under various settings and study their sensibility to different parameters. We investigate the relationship between DA, which is an implicit regularization technique, with an explicit regularization technique, namely the dropout procedure. We also further discuss and evaluate Deep Back-Translation (DeepBT), the DA technique for NLP tasks first proposed in [24]. We apply DeepBT to benchmark datasets and compare its outputs to results generated by previous existing methods.

We perform a broad and in-depth investigation of NLP DA methods by reviewing relevant literature and tackling some of the most significant research gaps. We used the previous [3] review work on DA. We are citing some of the main works contained in this review that focus on text classification and include other new techniques in our review. Therefore, we have three main objectives to be tackled in this work:

1. **NLP DA literature review:** first, we provide readers with an overview of the literature related to DA techniques in NLP tasks. This review starts with a brief introduction to the theoretical framework that supports DA methods, followed by an overview of the main approaches that have been explored as NLP DA techniques.
2. **Research Gaps distillation:** second, and based on the aforementioned review, we summarize the main research gaps found on NLP DA literature. We highlight research gaps found not only in terms of techniques, but also in terms of problems tackled and the lack of comparative studies.
3. **Deep experiments and results for DA algorithms:** finally, we study and compare several techniques to verify comparative advantages and explain their performance.

Hence, with this work we hope to be able to advance the state-of-the-art on the subject of DA in NLP and to deepen the understanding of techniques that overcome the bottleneck of limited labeled data.

With the aim of facilitating text understanding, we compile below the main acronyms used along this work.

The remainder of this paper is organized as follows. We present the Theoretical Framework supporting DA in Section 2. We follow in Section 3 with a literature overview of the DA research landscape, specially those works more closely related and applied to NLP. This is followed by a summary of the main research gaps found in the aforementioned overview. In Sections 4 and 5 we present a thorough evaluation and comparison of NLP DA methods by criteria such as computational cost, lexical diversity and semantic fidelity, addressing some of the aforesaid research gaps. We close this paper in Section 6, presenting our main contributions and pointing to open research questions and promising investigations paths.

2. Theoretical framework

In this section we examine the theoretical background that supports DA methods. We start with a brief overview of Machine Learning and Statistical Learning in 2.1, followed by theoretical considerations related to DA in 2.2.

2.1. Machine learning and statistical learning

Learning algorithms are procedures that are able to learn from data. More precisely, one can consider a procedure as a learning algorithm whenever it is able to satisfy the following condition: to "learn from experience E with respect to some class of tasks *T* and performance measure P, if its performance at tasks in T, as measured by P improves with experience E" [25]. ML is known as the study of learning algorithms.

This above definition is able to embrace the wide variety of applications and problems solved with ML. Task *T* includes common problems, such as Classification, Regression, Machine Translation, among various others. The performance measure P is responsible for quantifying the algorithms' success in its learning task. Depending on the task *T* being tackled, among common performance measures we could cite error rate, accuracy, quadratic error and cosine similarity. The experience E determines the source from which the process will learn. In supervised learning, the algorithms' experience E is a dataset containing features associated with a label or target [1].

When building a supervised ML model, the final purpose is to learn from previously seen inputs to be able to best predict unseen inputs. To this end, the ML model is trained on a training set and one tries to minimize the **training error** – the error calculated on observed data in the training phase. The real goal, however, is to minimize the error the model will perform on unobserved data — the so called **generalization error**. The generalization error can be defined as the expected value of the error on inputs drawn from a distribution expected to be found on the prediction phase [1].

Many practical problems – such as classification, pattern recognition, regression and density estimation, among others – are particular cases of a more general problem related to the process of function estimation from a given collection of data [26]. This type of problem can be analyzed under the general statistical framework of minimizing expected loss using observed data, which we develop below.

In supervised learning we aim at finding a function $f \in \mathcal{F}$ from a set of functions that best describes the relationship between random feature vector X and target random vector Y, which follows a fixed, but unknown, joint distribution $P(X, Y)$ [26,27]. To this end, we can define a loss function l that penalizes the difference between the predicted outcome $f(x)$ and the actual outcome y. Therefore, our goal is to minimize the average of the loss function l over the distribution P, a measure known as *risk functional* or *expected risk*:

$$R(f) = \int l(f(x), y)dP(x, y). \tag{1}$$

Since the distribution $P(X, Y)$ is unknown, in order to minimize the risk functional in Eq. (1) we can make use of an induction principle called Empirical Risk Minimization (ERM), which approximates the risk functional in Eq. (1) by the *empirical risk* functional defined as

$$R_{emp}(f) = \frac{1}{n}\sum_{i=1}^{n} l(f(x_i), y_i). \tag{2}$$

Hence, for finding the desired function f, we rely on a given set of training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$ formed by a set of independent identically distributed (i.i.d) observations drawn from the joint distribution $P(X, Y)$.

Therefore, solving the learning problem using ERM consists of estimating the function $f \in \mathcal{F}$ that minimizes the *empirical risk*:

$$\arg\min_{f\in\mathcal{F}} R_{emp}(f). \tag{3}$$

ERM is equivalent to minimizing the expectation of the loss function with respect to an empirical distribution $P_{emp}(x, y)$ formed by assembling δ functions located on each example [28]:

$$dP_{emp}(x, y) = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}(x)\delta_{y_i}(y). \tag{4}$$

A natural extension to the ERM framework is to replace the delta functions $\delta_{x_i}(x)$ by some estimate of the density in the vicinity of the point x_i, $P_{x_i}(x)$:

$$dP_{est}(x, y) = \frac{1}{n}\sum_{i=1}^{n} dP_{x_i}(x)\delta_{y_i}(y). \tag{5}$$

This more general approach leads us to the definition of *vicinal risk* function, defined as

$$R_{vic}(f) = \int l(f(x), y)dP_{est}(x, y) = \frac{1}{n}\sum_{i=1}^{n}\int l(f(x), y_i)dP_{x_i}(x). \tag{6}$$

Consequently, similar to the reasoning involving ERM, solving the learning problem with *Vicinal Risk Minimization* consists of estimating [28]:

$$\arg\min_{f\in\mathcal{F}} R_{vic}(f). \tag{7}$$

Minimizing empirical risk can lead to overfitting. To avoid this, one can add a complexity penalty term to the risk function we want to minimize. This process is known as regularization and the modified risk minimization process is the Regularized Risk Minimization principle [29]:

$$R_{reg}(f) = R_{emp} + \lambda C(f). \tag{8}$$

One of the most important conclusions drawn from Statistical Learning Theory is synthesized by the following Theorem [29], valid for finite hypothesis spaces $\mathcal{H}$: for any distribution P, and any dataset $\mathcal{D}$ of size N, drawn from P, the probability p that our estimate of the error rate will be more than ϵ wrong, in the worst case, is upper bounded as follows:

$$p(\max_{h\in\mathcal{H}} \mid R_{emp}(\mathcal{D}, h) - R(P, h) \mid > \epsilon) \leq 2\ dim(\mathcal{H})\ e^{-2N\epsilon^2}, \tag{9}$$

where $dim(\mathcal{H})$ is the dimension of the hypothesis space $\mathcal{H}$.

In other words, the gap between training and generalization error is bounded from above by a quantity that grows with model capacity and decreases as the number of training examples increases [1]. That is, the more observations are available at training time, the most the gap between training and generalization error is closed. This theorem leads naturally to the following question: is there any way of obtaining more observations without incurring into the high costs or limitations associated with collecting additional real world data? This is precisely the purpose of DA methods, whose theoretical background we will address in the following section.

2.2. Data augmentation

In various ML applications it is known that model output should not change when some set of transformations are applied to the input variables [30]. This property, called **invariance**, is generally task-specific. As an example, handwritten digit classifiers predictions should be invariant to small rotations of the inputs but not to reflection.

When presented with enough data – including numerous inputs–output pairs subjected to this invariant transformations – it is possible for the model itself to learn this invariances. However, considering that the number of combinations grows exponentially with the number of transformations, this is rarely the case due to limited data availability [30]. One can therefore induce the model to exhibit the necessary invariances without the need of collecting additional training data. There are four main approaches to this end:

1. Adding to the error function a regularization term in order to penalize changes in model output when the input is modified;
2. Building and using features that do not change when the required transformations are applied to the input;
3. Training with models in which invariance properties are naturally present due to its architecture (e.g. Convolutional Neural Networks (CNN), with its mechanisms like weight sharing, is able to incorporate some types of invariance);
4. Finally, the training set can be augmented using artificially created inputs transformed according to the desired invariances.

The last option is exactly the definition of a DA process. Despite being a subject whose theoretical framework has not been yet fully developed or unified, there has been growing interest in understanding the theoretical foundations for DA based on the principles of Statistical Learning. We will briefly comment some of the formal mathematical perspectives used so far, concentrating in the results obtained in [14,31].

In [14] the DA process is first modeled as a Markov Chain process. Under this approach, the authors show that Kernel classifiers appear naturally, even when dealing with other types of classifiers. Therefore, the authors investigate the effect of DA in Kernel classifiers using the ERM principle. Under the hypothesis that the applied augmentations are local – and thus not significantly modifying the feature map – and using first- and second-order Taylor approximation to the ERM principle, they draw two main conclusions: the first-order Taylor approximation of the ERM is equivalent to training the model on the average feature of all the transformed versions of the inputs; the second-order approximation is equivalent to applying a data-dependent regularization term to the objective. In conclusion, the authors show that DA methods have two effects: (a) increasing invariance by averaging the features of augmented inputs, and (b) penalizing model complexity via a regularization term based on data variance [14].

The DA theoretical framework developed in [31] arrives in similar conclusions and goes even further. Studying DA in a group-theoretical formulation, the authors show how DA leads to sample efficient learning and explain connections to other concepts of ML and Statistics such as sufficiency, equivariance, regularization, among others.

3. Data augmentation research landscape

In this section we present a literature review of DA methods, highlighting several recent researches that relate to the present work. We start exploring a variety of existing DA methods, taking special care to always consider and examine works that applied these methods to NLP problems. We close in Section 3.4 summarizing the main research gaps found in the literature.

The following subsections and taxonomy adoption is inspired by [32], which surveys and categorizes recent research on data augmentation applied to text classification tasks. Though focused primarily on classification problems, we regard the mentioned taxonomy as the most comprehensive and up-to-date method-oriented taxonomy, since other works, such as [33], give a more task-oriented overview of the field. Fig. 1 illustrates the organization of the mentioned taxonomy we will be using. It divides NLP DA methods first on the space on which the augmentation process is performed. First, we have Feature Space DA methods, which rely on the representation of the data, such as word embeddings or activations vectors of neural networks. Data Space methods, on the other hand, act directly on text data on its raw form, applying transformation to the characters, words, phrases or documents themselves. In the following sections, we will be explaining each of the categories the taxonomy is based on, along with presenting some of the most prominent works on each of them.

3.1. Feature space

Generative Adversarial Nets (GANs) are a type of NN architectures and training techniques that consists of a framework for estimating generative models first proposed by [34]. Since its inception, GAN-related research has been a rapidly evolving investigation topic, with impressive progress being made in training and applications. Several architecture variations have been developed upon the original, such as Conditional GANs (cGAN), Deep Convolutional GANs (DCGAN), Wasserstein GANs (WGAN) and Big GANs (BigGAN), just to name a few [35,36].

Due to its capability of generating highly realistic synthetic data, GANs have been applied as yet another method of DA. Within NLP tasks, [37] used a cGAN architecture to synthesize data on a low resource scenario in order to improve sentiment classifier generalization. The goal of the author was to train the cGAN models to mimick the Doc2Vec representations of the texts used in experiments, instead of obtaining the sentences themselves, thus acting on the feature space.

Besides GAN-based DA techniques, recent methods also propose interpolating representations on the feature space in order to generate new synthetic data. A very recent and promising class of DA methods are mixed sample – also known as mixed-example – techniques. These constitute a class of methods that artificially generate data by combining pairs of inputs drawn from the original set. They can be considered a more general approach to DA, since they are not necessarily label-preserving processes. The first general techniques for augmenting through linear combination of examples are *mixup* [27] and Between-Class ("BC") learning [38] – which are equivalent and were developed in parallel –, along with the improved "BC+" learning method [39].

In essence, linearity-based methods such as *mixup* are based on Vicinity Risk Minimization and consist of using a specific type of vicinal distribution. Sampling from this distribution produces feature-target vectors

$$\begin{cases} \tilde{x} = \lambda x_i + (1-\lambda)x_j & \text{where } x_i, x_j \text{ are raw input vectors} \\ \tilde{y} = \lambda y_i + (1-\lambda)y_j & \text{where } y_i, y_j \text{ are one-hot label encodings,} \end{cases} \tag{10}$$

where (x_i, y_i) and (x_j, y_j) are feature-target vectors randomly drawn from the original dataset and $\lambda \in [0, 1]$. As shown in Fig. 2, this technique leads to decision boundaries with linear transitioning from class to class. When the *mixup*-hyperparameter α, which controls the strength of interpolation, goes to zero, traditional ERM is recovered.

FMix is another mixed-example technique which uses binary masks obtained by applying a threshold to low frequency images sampled from Fourier space. First proposed in [40], FMix has also shown remarkable results in various tasks, achieving state of the art results on image classification, audio classification and Sentiment Analysis (SA).

More recently, mixed-example DA strategies that combine inputs and targets in non-linear ways were developed and applied to image processing tasks [41].

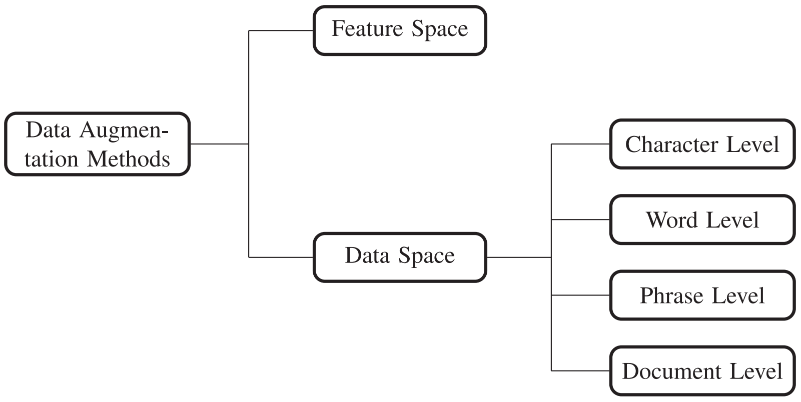

Fig. 1. Taxonomy of data augmentation methods that is adopted along this work.
Source: Adapted from [32].

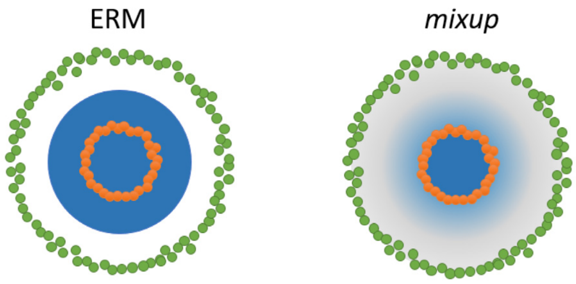

Fig. 2. Comparison of using ERM and *mixup* on a toy problem. Green and orange represent classes 0 and 1, respectively. Blue shading indicates the result of $p(y = 1|x)$ using ERM and *mixup*. *Mixup* encourages the model to behave linearly in-between training examples. In this example, *mixup* used $\alpha = 1$.
Source: Adapted from [27].

Despite proving very effective in improving generalization error, even after the use of other DA techniques, the exact reason why mixed-example methods are so successful remains as an open research question.

The applicability of mixed-example DA techniques to different types of data – image, sound, text, etc. – varies from method to method. BC learning [39] and *mixup* [27], e.g., are broader approaches, employable in various types of data. More general and non-linear mixing methods, such as those proposed in [41], on the other hand, operate only on images. In spite of the existence of NLP-applicable mixed-example DA methods, this field remains almost untouched. To the best of our knowledge, the works that have been developed exploring this family of DA techniques on text are very few and recent — most of them have been published or are in review process in 2020.

In [40], the authors propose FMix, a novel DA method, and test its efficacy in image classification, sound classification and Sentiment Analysis (SA) tasks. The results obtained in several SA tasks show *FMix* and *MixUp* obtaining gains compared to the baseline, but there is no clear winner between these DA techniques: their results are in most cases very similar.

The work of Guo et al. [42] proposes two variations to *Mixup* – wordMixup and senMixup. While in wordMixup word embeddings are interpolated, in senMixup sentences embeddings are interpolated to generate new synthesized data. These methods are applied to five different text datasets and perform better than their own no-DA implementation — although not always better than other authors' no-DA implementations.

Inspired by linguistic principles related to compositionality, the authors of [43] propose "Good-Enough Compositional Data Augmentation" (or GECA). This method, which can be considered a form of mixed-example DA, consists of generating new training examples by replacing fragments with other that appear in similar environment. The proposed protocol has the advantages of being model-agnostic and useful in a variety of tasks.

The work of Guo et al. [44] proposes SeqMix, a sequence-level variation of *MixUp* which performs soft combination of input/output sentences from the training set. The random combination of sentences prevents the model from memorizing long segments and encourage them to rely on compositions of subparts to predict the output. Another relevant contribution of this paper is to provide a theoretical framework that unifies several other DA strategies for compositionality (WordDrop, SwitchOut and GECA).

3.2. Data space

As already mentioned, Data Space DA techniques deal with the generation of synthesized data applying transformations directly to its raw form inputs. These modifications can occur on a variety of levels, ranging from character level to the document level, passing through word and phrase level transformations.

3.2.1. Character level

On its smallest scale, DA transformations applied to NLP taks can apply on a character level. These modifications may be drawn by some predefined rules or by noise injection. The work [45], for example, focuses on operations that simulate spelling errors or phonetic similarity in English. Other works, like [46], perform changes to simulate typing errors (Keyboard Augmenter) based on the distance of the letters on the QWERTY keyboard. Finally, other techniques mimic the behavior of optical character recognition systems [46] in which image recognition can confuse characters with similar formats.

Since the seminal paper of [47] showing the counter-intuitive vulnerability of state-of-the-art Neural Networks (NN) to small perturbations in its inputs, the field of *adversarial attack* has been an active area of research, especially within the Deep Learning community. Adversarial examples, synthesized data generated by adding imperceptibly small perturbations in original examples for the sole purpose of misleading trained models, raised the question of the generalization ability of NNs. Following works, such as [48], helped deepen our understanding on how adversarial examples affect both linear models and NNs. More than that, the authors leveraged the knowledge of the adversarial flaw as an opportunity to fix it, giving rise to *adversarial training.*

Adversarial training consists in the process of augmenting original data with adversarial examples and training the model on this augmented dataset. Adversarial training differs from traditional DA techniques because, whereas traditional methods synthesize data with transformations that are expected to occur in the test set, adversarial training uses inputs unlikely to occur [48]. Instead, this technique uses adversarial examples to expose and correct the flaws in model decision functions and provide additional regularization benefits beyond what is achieved by using dropout exclusively.

Several works have been developed to apply adversarial training to NLP models. The work by [49] proposes MHA, an adversarial example generator based on Markov Chain Monte Carlo (MCMC) sampling. As another example, [50] implements in a Python package several NLP adversarial training techniques from the literature. In [51] the authors perform a systematic survey of adversarial attacks research applied to NLP problems since 2017, when the first of such an application was published.

Adversarial training methods acting on the Data Space may act on different levels and were presented in this section for brevity and conciseness. While some of the methods implemented in [50] act on the character level, the work of [49] acts on the word level, which is the subject of the next topic.

3.2.2. Word level

Word replacements could take form based on a thesaurus, which could guide word replacements based on semantic closeness to frequently seen meaning. In [52], e.g., a thesaurus obtained from the WordNet [53] is used to replace words according to a probability distribution, augmenting datasets used for text classification tasks.

Easy Data Augmentation (EDA) is yet another example of simple manipulation techniques, which combines several manipulations in a single unified method [54]. This method consists of applying a set of simple operations to the original text in order to generate new synthetic texts. For any given sentence in the training set, one of the following operations is randomly chosen and performed:

1. **Synonym Replacement:** randomly chosen words are replaced by one of its synonyms (also chosen at random) from the WordNet dictionary. Stop words are not considered in this operation.
2. **Random Insertion:** a random word from the given sentence is chosen (stop words are not considered). A random synonym of this word is then inserted in a randomly defined position in the sentence.
3. **Random Swap:** randomly chosen words are defined and their respective positions are swapped.
4. **Random Deletion:** randomly remove words from the sentence, following some probability parameter.

The operations are all randomly applied according to the parameter α, which controls the percentage of words changed in any given sentence. Though maybe original in its proposal as a pure DA technique, the use of this set of operations closely resembles the noise injection procedure proposed by [18] as an auxiliary task to improve Neural Machine Translation (NMT) models. The work [55] demonstrated that substitutions for words that are synonyms that are rarer in frequency also obtain better results in machine translation tasks, especially for low-resource languages.

Other word manipulation algorithms are based on more complex models than synonym substitutions or other random operations. These techniques represent words by embeddings to insert and replace them according to the similarity they present with other words in the embedding space. The work of [56] leverages frame-semantic and lexical embeddings (such as the popular Word2Vec [57] Word Embedding) to replace original words by their k-nearest-neighbors vectors on the embedding space. The authors assess this strategy comparing it with training the same model without data augmentation using a two-tailed student's t test and achieve statistically significant superior performance on categorization models on social media text.

Word2Vec, Glove, and FastText Representations represent words. However, they are static representations that do not change according to context. The work of [58] builds upon the assumption that sentences are natural even when their words are replaced by other words with paradigmatic relations – i.e., words with the same word class – to propose *contextual augmentation*. Instead of synonyms or similar word embeddings, this technique replaces words by the words predicted by a Language Model given the context surrounding the original words to be replaced. More specifically, the authors use a bi-directional LSTM-RNN Language Model to generate new words and compare these method to traditional synonym replacement technique on several classification datasets. The proposed augmentation technique, though superior to plain synonym replacement, brought marginal gains on some of the tested scenarios.

More recent works build upon the aforementioned ideas and take advantage of state-of-the-art architectures to achieve even better results. In [59], the authors use BERT (Bidirectional Encoder Representations from Transformers) to overcome some of the bottlenecks found in the LSTM-based LM used by [58]. Based on the original BERT architecture, but with differences on the inputs representation and training procedures, the authors propose *conditional BERT*. This LM, after well-trained, is used to augment sentences: given a labeled sentence, a few random words are masked and conditional BERT is used to predict new words that are compatible with the label of the given sentence. This method is then compared to the same methods explored in [59] and on the same datasets and achieves superior results on all evaluated scenarios.

3.2.3. Phrase level

Other methods generate synthetic data by acting on the phrase-level of original data. More closely inspired by two common DA manipulations used in image processing, the work of [60] proposes NLP-equivalents of cropping (focusing on a particular item) and rotating. Using dependency trees, the authors crop forming smaller sentences from the original ones and rotate/swap words around some defined root to inflate data for training part-of-speech tagger models. Despite directly inspired by successful techniques in image processing, these NLP-equivalent manipulations did not perform so well in text data, bringing negligible gains in most of the tested scenarios.

Paraphrases are ways of rewriting a text to maintain the original semantic meaning. The construction and identification of paraphrases are areas of great importance in NLP studies, such as summarization or question–answering. Due to their sentence rewriting characteristics, paraphrasers can behave like good DA [61] algorithms since they introduce lexical diversity, maintaining fidelity to the original meaning. The work of [62] introduces PPDB, a paraphrase database that can act as an important resource to generate synthetic data. Combining several English-to-foreign bitext corpora, the authors were able to create a database with tens of millions of lexical, phrasal and syntactic paraphrases. To build the mentioned database, the authors of PPDB use a bilingual pivoting method that assume that two English strings *e1* and *e2* that translate to the same foreign string f can be assumed to have the same meaning [62]. Thus, pivoting over f make it possible to extract a diverse set of paraphrases $\langle e1, e2\rangle$. These paraphrases can in turn be used to substitute original text, thus achieving DA.

3.2.4. Document level

Considered crucial to NMT tasks nowadays [63], Back-Translation (BT) is another method for generating auxiliary synthetic data. First introduced by [64], the term BT was initially conceived specifically within the context of machine translation, whereby monolingual data was leveraged by translating *Target Language* → *Source Language* (hence the term *Back*) in order to obtain additional training data for the *Source Language* → *Target Language* final translation task. The first implementation of Back-Translation as a DA method for down-stream tasks seems to be the work of [65], which used BT to rephrase original sentences (i.e. generating paraphrases), producing extra data and obtaining state-of-the-art results on question–answering tasks. Supported by its remarkable success in NMT tasks, there has been an emergence of numerous variations to the traditional BT method.

Iterative BT (IterativeBT), proposed in [66], is a process where models are successively trained using data Back-Translated by the previous model. This cyclical training yields generation of models which are able to improve at each iteration, although the quality of the BT system in use being crucial to the success of the proposed approach. The authors report improvements in NMT systems that apply IterativeBT both in high-resource as in low-resource scenarios.

Noised BT (NoisedBT),[1] presented by [18] is a variation where noise is injected to the Back-Translated text. In the seminal paper, three types of noise are used: random deletion of words, random replacement of words and random swapping of words. Despite the fact that final noised sentences are not realistic, the authors argue that the superior performance obtained by noise injection could be attributed to the model becoming robust to reordering and substitutions occurring naturally on texts. The authors also report an intriguing finding when comparing model improvements obtained by adding artificially generated text data with those obtained by adding real data: in cases where the domains match, synthetic data can be nearly as effective as real human translated data.

Tagged BT (TaggedBT) [67], heavily influenced by the works of [18,68], proposes another hypothesis for the superiority of noise injection in BT postulated in [18]: instead of increased text diversity, noise injection would instead benefit the final model by signaling which data is synthetic and which is original data. To assess the validity of this assumption, the authors trained a translation model where artificially generated source data had no noise injection and was rather tagged with a reserved token. The fact that this method leaded to similar or even slightly higher performance supported the hypothesis of the authors.

DA methods can be evaluated by different perspectives, two of the most important being the validity and the diversity of the data they generate when augmenting the original training set [69]. Robust DA techniques should generate a diverse set of examples to prevent overfitting. On the other hand, this diversity should not hinder synthetic data validity. As a consequence, DA methods should strive for adequate balance within the diversity-validity trade-off in order to be successful [70].

Exploring the diversity-validity trade-off, our previous work [24] proposed a new method for DA, named DeepBT, which adds more layers of intermediate translations between the original text and the final paraphrase. Using capital letters to represent original and final languages (which are always the same) and arrows to represent translations to languages L, while in the original BT we always have

$$A \rightarrow L \rightarrow A, \tag{11}$$

[1] The term Noised BT was not used in [18], but coined in [67].

in DeepBT we could have n intermediate layers of translations:

$$A \rightarrow L_1 \rightarrow L_2 \rightarrow \cdots \rightarrow L_n \rightarrow A. \tag{12}$$

Fig. 3 illustrates the difference between traditional BT and a 2-layer DeepBT.

Following the rationale of the BT method, which generates paraphrases with the same meaning and label as the original one, the DeepBT technique is created under the hypothesis that using several intermediate languages between the original and destination one could increase diversity on generated paraphrases without hurting validity. With greater diversity in training data we assume that we could reduce overfitting and achieve greater performance.

With the increasing capabilities of language generation models, generative methods have become interesting and promising alternatives to BT-based methods as strategies to document level DA. In [71], authors propose language-model-based data augmentation (LAMBADA), which consists of a data augmentation meta-learning technique using generative pre-trained neural networks (GPT). The main idea of this algorithm is to make the GPT language model learn how to generate textual data from some label, with a classifier filtering the generated phrases with better quality. LAMBADA has several similarities with Meta-Learning algorithms, mainly due to using a discriminator classifier to control the generated data. The LAMBADA algorithm is described in Algorithm 1.

Algorithm 1: LAMBADA Algorithm

```
Train initial classifier $h$ on original dataset $D_{ori}$ ;
Fine-tuning a language model $LM$ (usually GPT-2) on
  original dataset ;
Use the fine-tuned language model to generate $N$ synthetic
  data;
Use classifier $h$ to filter quality synthetic data generated in
  previous step, based on threshold $L$. The filtered data
  makes up the synthetic dataset $D_{syn}$ ;
Merge the original dataset $D_{ori}$ with the synthetic dataset
  $D_{sint}$, forming the final dataset $D_{final}$;
Return: Final dataset $D_{final}$;
```

The first step of classifier training is to build a classifier from the currently available data. This classifier will have the role of separating the good results of a class from the harmful results.

The next step is fine-tuning the GPT language model (generally, we use GPT-2). In this step, the GPT model receives training so that the generated texts are conditioned to the original observations. In other words, the training updates the weights of the GPT network so that the word probabilities are more consistent with the original data that the user wants to apply DA.

In step 3, the GPT model generates new data from parts or excerpts of the original texts. Finally, the last step is filtering the texts generated with quality by the classifier of step 1 to generate the augmented dataset, while the data generated with low quality are discarded.

Data Boost is a DA algorithm that uses text generator models [72]. Generally, GPT-2 generates and changes synthetic texts, similar to LAMBADA; however, it does not condition the language model like LAMBADA. In contrast, Data Boost uses reinforcement learning to change the decoder policy of the GPT-2 token selection model. For policy optimization, Data Boost adopts frequentist modeling of words according to their use in texts of some class, called Salience Gain [72]. The reinforcement learning algorithm defines Salience Gain as a reward function that guides the PPO algorithm [73] for GPT-2 decoder. Thus, Data Boost gains a competitive advantage in not having to condition language models like LAMBADA, a very computationally expensive algorithm.

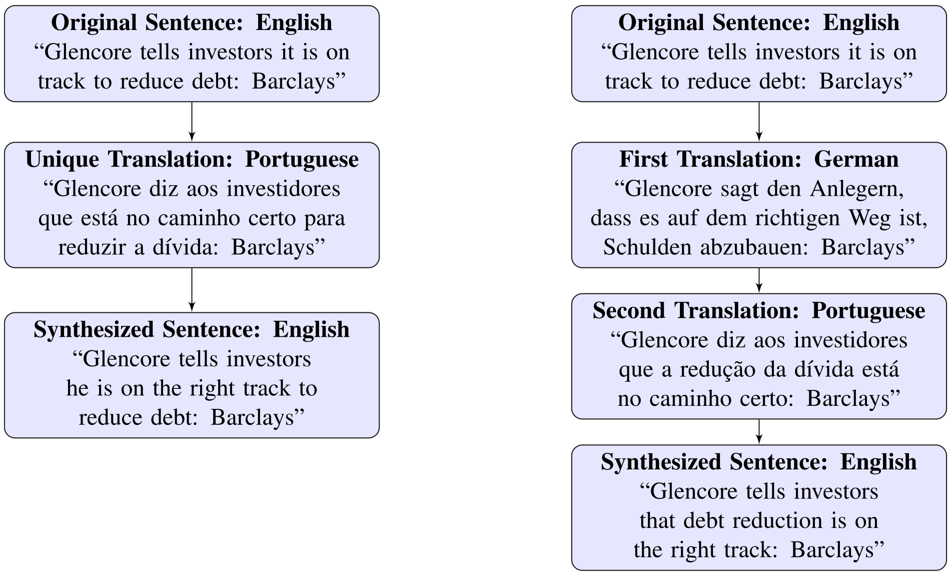
Original Sentence: English
“Glencore tells investors it is on track to reduce debt: Barclays”
Unique Translation: Portuguese
“Glencore diz aos investidores que está no caminho certo para reduzir a dívida: Barclays”
Synthesized Sentence: English
“Glencore tells investors he is on the right track to reduce debt: Barclays”
Original Sentence: English
“Glencore tells investors it is on track to reduce debt: Barclays”
First Translation: German
“Glencore sagt den Anlegern, dass es auf dem richtigen Weg ist, Schulden abzubauen: Barclays”
Second Translation: Portuguese
“Glencore diz aos investidores que a redução da dívida está no caminho certo: Barclays”
Synthesized Sentence: English
“Glencore tells investors that debt reduction is on the right track: Barclays”

Fig. 3. Graphic representation of Deep- and traditional BT procedures. In DeepBT (right flowchart) we stack several intermediate layers of translation, whereas in traditional BT (left flowchart) there is always only one translation between original and synthesized text. As in this example, the additional translation layer present in DeepBT is able to generate synthesized text different from the text generated by traditional BT.

More recent works carry out training in text-to-text networks to specialize in automatic paraphrase generation. Parrot [74] is a new Python paraphrasing framework built on models with fine-tuning paraphrases. Parrot seeks to generate words with adequate meaning, fluency, and linguistic diversity in English. Among the trained models, we can highlight the paraphraser in T5 architecture. The package parrot is useful for DA.

Another paraphraser used in DA is BART trained in the ProtAugment algorithm [75]. ProtAugment is an unsupervised meta-learning technique that can be used for DA in intent detection models. The algorithm combines the generation of paraphrases from the BART model trained on MSR and PAWS with sentences generated from the traditional back-translation. The BART model is a self-supervised learning that is an area that can present a new efficiency gain for DA algorithms [3].

Basic manipulations DA methods are generally characterized by easy of implementation, since there is almost no change at all to the way the problem is postulated and solved besides the simple modifications performed in training data.

3.3. Meta learning

A great number of DA strategies rely on manually specifying the necessary transformations that generate artificial label-preserving data. This type of procedures have several drawbacks. First, one often has to have significant domain knowledge of the dataset to be able to chose the appropriate transformations, relying purely on experience and intuition. Second, the trial and error process of exploring different augmentation techniques f requently result in prohibitive computing resources or timing, specially when using Deep Learning models. Another unwanted characteristic of handcrafted DA transformations is not being universal. As a consequence, successful data modifications cannot be easily applied to other datasets with similar performance improvements. Meta Learning DA techniques consist of a set of various DA methods that try to learn from training data the most appropriate transformations so as to achieve better generalization power.

For example, in [19] the authors propose AutoAugment, a strategy to automate the process of generating synthetic data. In this work policies express augmentation operations and Reinforcement Learning is used to find the transformations that yield the best validation accuracy on a target NN. The work of [76] adapts AutoAugment – whose basic architecture is depicted in Fig. 4 – to a dialogue generation setting, and choose operations such as Stop word Dropout and Grammar Errors as the policy search space.

In [20] the authors propose Population Based Augmentation, whose goal is to learn a schedule of augmentation policies, instead of a fixed policy as in AutoAugment. This choice is accountable to the far superior computational efficiency of this method when compared to AutoAugment.

New meta-learning methods have been proposed. An important consideration is to choose a good set of augmentation functions, as redundant or overly aggressive augmentation can slow down training and introduce biases into the dataset.

3.4. Research gaps

Given the above literature review, here we want to highlight some of the still open research questions related to NLP DA methods. We summarize below the most patent research gaps found in the NLP DA literature:

1. **Lack of comparative studies between different classes of NLP DA methods**: When proposing new methods, authors generally restrain their comparative assessments to the same class of DA methods they are contributing to, failing to contrast their findings to other existing categories of DA techniques. Works proposing variations to the traditional BT method generally compare their results only to other BT-based techniques. Despite BT-based methods being commonplace at the time, the work introducing EDA [54] did not contrast EDA results against any BT-based method, comparing it only against a no DA setting. The same is true for MSDA methods. In [40] the authors compare FMix, the proposed technique, to *mixup* and CutMix — EDA and BT-based methods were left without mention. As a consequence, there is no unified overview of how all these NLP DA methods compare to each other in any aspect (quality of synthesized data, performance gain, etc.).

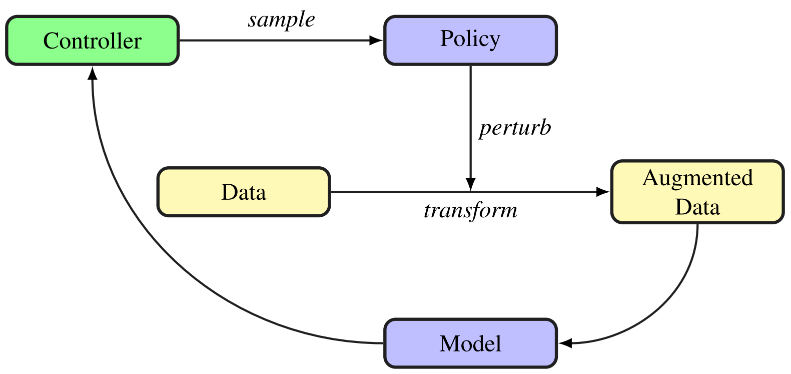

Fig. 4. System design of AutoAugment, a technique for generating data-augmented examples using Reinforcement Learning. A Controller samples a Policy to perturb original Data, generating Augmented Data. The artificially generated data is used for training the model, whose performance is fed back to the Controller.
Source: Adapted from [76].

Notable exceptions to this lack of comparative studies are the works of [77,78]. However, besides being relatively limited in the scope and number of methods compared (between 4 and 5 on both works), these works can already be considered somewhat outdated considering the time when they were published and the rapid evolution of NLP DA research landscape since then.

2. **Comprehensive evaluation criteria:** Contrary to what is usual in image processing, NLP DA works rarely address any topic beyond model performance. Qualitative evaluations of the artificially generated data — are they by any metric similar to the original data? Are they comprehensible to a human, maintaining their original content? – receive little to no attention. The works of [79,80] constitute efforts in this direction, but both of them focus mostly in analyzing simple DA heuristics or DA evolving simple manipulations, leaving important DA methods unmentioned. Other simple but important practical considerations, such as time and resource usage of competing methods, are often not mentioned by researchers. Thus, considering the lack of works that take these variables into account and as highlighted in [32], enhancing the understanding of DA methods in NLP – both in terms of qualitative assessment as well as in practical implementation terms – constitute an important research agenda.
3. **Accessibility and Reproducibility:** as noted by both [3, 32], research on NLP-applied DA has suffered from low reproducibility, transparency and low adoption of good practices. Researchers often do not make their code and datasets publicly available nor report variation among results (e.g. standard deviation). As summarized by [32], these limitations often hinder the usability of DA applications among practitioners.

In summary, to the best of our knowledge, there has not been any research performing a thorough investigation and comparison of NLP DA methods in a transparent and reproducible way, and with comprehensive evaluation criteria. We start to tackle these research opportunities in the following sections with a comparative analysis of NLP DA techniques.

4. Comparative analysis of NLP data augmentation techniques

In this section, we carry out a series of experiments to comparatively evaluate the performance of different DA algorithms in different applications. For this, we selected algorithms that cover families of NLP DA techniques that are widely and successfully used, but which have not yet been compared with each other. We selected the following algorithms:

- Data Space: Character Level:
  - Keyboard Errors Augmenter (QWERTY Keyboard).
- Data Space: Word Level:
  - Easy Data Augmentation (EDA) [54]
  - Embedding Augmenter [56] (we are using Glove-300 dimension model);
  - Contextual Embedding BERT Augmenter [56];
  - Self-Supervised Manifold Based Data (SSMBA) with BERT uncased model [81].
- Data Space: Phrase Level:
  - PPDB Augmenter [82,83] with Phrasal, Lexical and Syntactic rules with medium English package.[2]
- Data Space: Document Level:
  - Back-Translation (English to Czech for more diversity [61]);
  - Deep Back Translation (English to Czech to Russian, again in order to achieve more diversity) [24];
  - T5 Paraphraser Parrot;
  - BART Paraphraser ProtAugment [75];
  - LAMBADA with GPT-2 language model [71].

Different criteria are used in the comparative analysis we do: processing and time requirements, lexical diversity introduced by the DA algorithm, semantic fidelity of the synthetic data in relation to the original data, and performance gain in the DA task.

We did not include Feature Space DA algorithms in the comparison for a few main reasons. First, Data Space algorithms are most commonly used in many problems. Secondly, analyses of the diversity of the generated texts and their semantic fidelity do not make sense to be carried out with Feature Space techniques, as they are applied to the outputs of the neural network layers, performing techniques such as mixup or noise addition. So, in general, they do not generate new texts. Finally, to test Feature Space DA algorithms, we must change not only the training scheme, but also the classifier itself used in the tasks, which would probably not make the comparison fair between all algorithms.

All test code is available in the GitHub https://github.com/lucasfaop/survey_text_augmentation repository. The algorithms were implemented with the help of Python's NLPAUG package [84], a library that unifies many DA algorithms.

[2] Available at http://paraphrase.org/#/download.

For a standardized comparison, we chose three benchmark datasets of great importance in the context of NLP, described in the next section.

4.1. Application domains and benchmark datasets

We have chosen three benchmark datasets of authentic texts with varying complexity to solve and of great importance in the context of NLP:

- The SemEval 2018 Task 3A datasets [85] for identifying ironic tweets,
- The TREC question classification [86] and
- The emotions detection dataset [87].

The SemEval 2018 Task 3A [85] corpus presents 3,834 extracted actual tweets, classified into ironic (label 1) and non-ironic (label 0), with a test dataset of 784 tweets divided into ironic and non-ironic. Here we are dealing with a binary classification problem, and the distribution of the label is practically balanced (50.2% of ironic tweets and 49.8% of non-ironic ones). Although the problem is binary and balanced, irony identification is complex due to many factors. Initially, irony is not an easy figure of speech to identify; even humans make many mistakes. To find irony is essential to interpret the entire context of the sentence and identify possible references, complex tasks, or classification algorithms. In addition, the texts contain emojis, abbreviations, neologisms, links, and mentions of profiles on Twitter, all characteristics that make the task more challenging. The best result (without DA) achieves performance of 71% for F1-macro for this task [85], using a BERT classifier (state-of-art of transformers architecture).

The other dataset selected is the TREC question classification [86], consisting of open-domain, fact-based questions divided into broad semantic categories. The base contains 5452 questions labeled for training and another 500 questions labeled for testing. Average length of each sentence is 10 words, vocabulary size of 8700. This problem is considered easy to medium complexity. The base response variable is what the question is about it. The variable has 6 possible classes:

- **Abbreviation:** questions ask about meanings and names of abbreviations.
- **Descriptions:** questions are about procedures and how they work.
- **Humans:** ask for names of people and authors.
- **Location:** questions are about geographic places.
- **Numeric:** questions are about numeric data (size, weight, dates, conversions).
- **Entities:** questions are about names of specific things such as animals, colors, organs, techniques, and others.

The class distribution is 22.9% for **Entities**, 22.4% for **Humans**, 21.3% for **Descriptions**, 16.4% for **Numerics**, 15.3% for **Location** and 1.5% for **Abbreviation**.

Finally, the last dataset chosen is the emotion recognition dataset [87]. The dataset presents texts that represent some feelings: sadness, joy, love, anger, fear, and surprise. Emotion recognition or sentiment analysis is one of the most classic problems in NLP, and we have chosen this dataset to analyze how much Data Augmentation can improve the performance of models. The response variable of this dataset is the emotion present in the text with the following distribution: 35.2% of joy, 27.5% of sadness, 13.75% of anger, 10.6% of fear, 8.9% of love, and 4.05% by surprise. This dataset is considered a medium to high complexity problem due to the contextual similarity of some feelings.

4.2. Data augmentation and preparation procedure

In this section, we explain the procedure for generating synthetic data using DA algorithms and how we compare the algorithms.

As stated before, the dataset is separated into a training dataset and a test dataset. We separate a percentage of the training dataset (10%, 20%, 30%, 40%, 50%, 75%, and 100%) where we apply each of the algorithms selected in the previous section, and we also do not apply any DA to serve as a baseline of comparison. For each algorithm, we generate five synthetic data for each original data. Thus, the comparison of algorithms will be standardized.

For the Keyboard Errors Augmenter, we put the change parameters following the QWERTY distance of the characters, and we defined a maximum of 30% as the limit for replacing the original characters.

The EDA algorithm uses all operations with words and with $\alpha = 0.05$.

Changing algorithms with embedding models also use the threshold of changing a maximum of 30% of the original words. For example, the static Embedding Augmenter algorithm uses the Glove model with 300 dimensions to find the most similar words. On the other hand, the BERT algorithm uses the base BERT model itself to generate the embeddings.

The BT and DeepBT algorithms consist of operations with the M2-M100 multilingual translator model, which presents many language options for translations. Model M2-M100 are using the parameters at default values in the decoder in all tests. Traditional BT performs operations from English to Czech, while Deep BT performs translations from English, Czech and Russian.

The T5 Parrot and BART ProtAugment paraphrasing models are in the default settings, with the exception of the `do_diversity` parameter in Parrot. Thus, Parrot is expected to try to generate paraphrases with greater diversity.

Finally, the LAMBADA algorithm uses a BERT classifier and the GPT-2 language model to perform the discriminator classifier and generator model of the technique, respectively. The BERT classifier is trained in 2 epochs, while the GPT-2 model performs 6 epochs of fine-tuning for language conditioning. Finally, only texts that reach a threshold of 0.5 in the classifier are kept; the others are discarded.

The performance comparison consists of training a DistilBERT model (basically the traditional BERT, but with fewer layers and parameters), trained in only one epoch. For training this classifier, we adopted the fine-tuning approach: we use the pre-trained DistilBERT model and include a new classification layer in the model where training adjusts its weights. After training the model, we perform inference on the testing dataset, which is never "seen" by any algorithms. We repeated this process ten times to collect the average performance of the algorithms. Finally, as an evaluation metric, we use F1-macro as a form of comparison in all three datasets.

5. Results

In this section we start by showing some illustrative examples of how DA algorithms work. Then, we perform a series of analyses, namely: computational cost of each algorithm, diversity of synthetic data generated, and performance gain achieved with the use of DA.

Table 1
Synthesized Data generated by different DA techniques in SemEval2018.

Augmenter	Label	Text
Original	Ironic	Nothing makes me happier then getting on the highway and seeing break lights light up like a Christmas tree.
Keyboard Error	Ironic	N8thiMg makeq me hApLier then getting on the highway and seeiMg bGeak lightQ light up like a Christmas tree.
EDA	Ironic	Makes me happier then on the highway and seeing break light like a tree.
Embedding	Ironic	Nothing makes me emotional saner long when getting blaize on the highway and suddenly however setpieces up like a Christmas tree.
BERT Aug	Ironic	Nothing makes me happier then getting on the seat and seeing highway lights light up of several floating toy.
SSMBA	Ironic	That makes me happier then getting on the highway and seeing my lights lit up like a christmas candle.
PPDB	Ironic	Nothing makes me happier subsequently targeting on the road and testing break insights reflects up like a Christmas tree.
Back translation	Ironic	There is nothing that would make me happier when I get on the highway and see broken lights shine like a Christmas tree.
Deep back translation	Ironic	Nothing would make me happier when I got on the highway and saw the light spreading like a Christmas tree.
T5 Parrot	Ironic	No one makes me happier than getting on the highway and seeing the break lights light up like a christmas tree
BART ProtAugment	Ironic	Nothing makes me happier then getting on the highway and seeing the lights light up like a Christmas tree.
LAMBADA	Ironic	Nothing makes me happier then being in a bad yuppie movie

Table 2
Synthesized Data generated by different DA techniques in TREC dataset.

Augmenter	Label	Text
Original	ENTY	What films featured the character Popeye Doyle?
Keyboard error	ENTY	Wyat films featured the chZdacter Popeye Doyle?
EDA	ENTY	Films what the character Doyle?
Embedding	ENTY	What directing featured the often Popeye Doyle?
BERT Aug	ENTY	Note, what films featured the comedian character popeye doyle?
SSMBA	ENTY	What films have the character jessyea?
PPDB	ENTY	What films attracted the features Popeye Doyle?
Back translation	ENTY	Which films included the character of Popeye Doyle?
Deep back translation	ENTY	What films show the character of the pope Doyle?
T5 Parrot	ENTY	What movies have featured the character of popeye doyle?
BART ProtAugment	ENTY	What films featured Popeye Doyle as a character?
LAMBADA	ENTY	What films featured a Black Panther, a Superman and a Wonder Woman?

Table 3
Synthesized Data generated by different DA techniques in Emotion Recognition dataset.

Augmenter	Label	Text
Original	Joy	I can have for a treat or if i am feeling festive
Keyboard error	Joy	I can have for a trewt or if i am fe@lint feet7ve
EDA	Joy	Can I for have or if feeling am festive
Embedding	Joy	I can soother have for a treat or feza if cory-wright i am feeling festive
BERT Aug	Joy	I can have bought a treat or if may remember feeling festive
SSMBA	Joy	I would have for a great fare if i am feeling festive
PPDB	Joy	I can removed for a treat or if i bet recognizing festive
Back translation	Joy	I can have for treatment or when I feel a holiday
Deep back translation	Joy	I can be treated or if I feel holiday
T5 Parrot	Joy	If i'm feeling festive i can have it for a treat
BART ProtAugment	Joy	I can have for a treat or if i feel festive.
LAMBADA	Joy	I can have a lot of fun cooking and laughing

5.1. Illustrative examples of DA algorithms

Initially, it is interesting to analyze the functioning of the algorithms and make a more qualitative analysis of the synthetic data obtained from the application of each DA algorithm. Tables 1, 2, and 3 show some examples of the application of DA algorithms on original sentences in the selected datasets.

The first algorithm we look at is Keyboard Error. As explained earlier, Keyboard Error applies random changes to characters, simulating typos (thus being categorized at the Character Level of the given taxonomy). In the examples, we can see the type of transformation that the algorithm performs on the data: the algorithm exchanges characters for other nearby characters on the computer keyboard. Some examples of character changes are p for l, n for m, a for w, or z, among others. Despite being a simple technique, Keyboard Error's impact is relatively significant. The synthetic data obtained are difficult to read, often losing their semantic meaning. The main idea of the algorithm is that with the addition of this noise, the classifiers achieve a greater generalization capacity and become more robust.

The following technique is EDA. This technique also introduces noise into the text to make the models more robust, but at the word level. In the examples given in the tables, we can see the EDA changes, such as change in word order (such as "movies" in the TREC example) or removal of words (such as "treat" that was deleted from the original text in the Emotion Recognition dataset).

Other word-level algorithms are Embedding Augmenter and Contextual BERT Embedding. Both make insertions and changes to words by encoding embeddings of the words contained in the original sentence. The difference between the two algorithms is that Contextual BERT Embedding uses a dynamic (contextual) embedding, while Embedding Augmenter uses a fixed embedding. In the examples of application of the algorithms, we see that both make substitutions and insertions of words, generally preserving the semantic sense and the syntax of the original sentences. The main difference between them is that Contextual BERT Embedding makes more diverse changes, such as the adjective "comedian" or the verb "note" inserted in the TREC example. This greater diversity is due to a dynamic embedding that allows a complete understanding of the original phrases. However, both algorithms can make changes that impact the original sense, as seen in the SemEval 2018 database examples.

SSMBA is another word-level algorithm. Using the corruption function, masking some tokens, and applying BERT to recompose the sentence, the algorithm can change the original sentence, but it does not change its meaning much. The SSMBA changes are masked token replacements in the selected examples, making SSMBA similar to the Embedding and Contextual Bert Embedding algorithms in the results. However, SSMBA much more preserves the meanings of the original sentences. This balance of diversity and fidelity of meaning is an essential characteristic of the SSMBA augmenter.

At the phrasal-level of the DA taxonomy there is the PPDB algorithm. This algorithm is based on many sentences and expressions with similar meanings. This collection of phrases and expressions considers the semantic role of each word (adjective, verb, adverb, and others). The goal is that the algorithm can change or insert expressions with the least possible impact on the original meaning. In the PPDB examples shown in tables, we can see that the algorithm makes many changes to the original sentences, changing words to expressions, such as "highway" for "on the road" or "I am feeling" for "I bet recognizing" compared to word-level algorithms. However, despite making several changes, PPDB does not impact the original meaning as much as Embedding or Contextual BERT Embedding, getting closer to SSMBA in keeping the original meaning.

Finally, we present some document-level algorithms from the DA taxonomy. Translation-based algorithms (Back Translation (BT) and Deep Back Translation (DeepBT)) use translations to alter the original texts. When looking at the application examples, we noticed that the changes in the sentences are subtle and manage to keep their original meanings well, but often they do not offer much diversity, which can be a problem depending on the situation. Another important note is that depending on the quality of the translators, some translations may not be as accurate and may include words out of context, such as "holiday" in the Emotion Recognition examples.

Other document-level algorithms are the T5 Parrot Automatic Paraphraser and the BART ProtAugment. Both also make subtle changes to sentences and manage to keep the original meanings intact. These paraphrasers are best at keeping the original meaning, but they have less diversity than other algorithms. T5 Parrot generates more diverse phrases than ProtAugment, which can be an advantage. However, ProtAugment manages to maintain the original semantic content of texts very well.

To complete the document-level algorithms, we present LAMBADA, which excels in the diversity of generated sentences — better than any other algorithm tested. However, LAMBADA does not undertake to maintain the meaning of the original document. This feature can offer a significant risk of introducing unwanted biases into the training dataset, such as generating sentences that do not maintain the original labels.

5.2. Computational costs

This section discusses the computational effort and time to compile DA algorithms. This comparison is essential to evaluate the trade-off of using these algorithms to increase the performance of ML models. We compile all algorithms using Google Colab Pro with GPU Tesla P100-PCIE-16 GB with memory of 16 GB. We selected some criteria for comparison of computational requirements: use of external files, external models, the requirement to use GPU to speed up processing, memory usage (size of files/models in general), and processing speed. In this way, we can have an objective notion of the most computationally expensive algorithms, those that use more memory, and those that are faster or slower. The Table 4 presents the complete comparison of the algorithms.

As we can see, Keyboard Error and EDA algorithms are the least expensive and the fastest algorithms. These characteristics can be competitive advantages over other algorithms. Embedding and BERT Augmenter are algorithms that require moderate memory due to their use of embeddings. However, the use of GPU speeds up BERT Augmenter's processing. LAMBADA is one of the most computationally demanding algorithms, as it requires classifier training and a language model, making it unfeasible without the use of GPU.

We can compare algorithms with paraphrasing behavior. PPDB is the fastest algorithm and does not need to use GPU, but it has extensive memory usage, which can be a constraint. Those based on translation (Back Translation or Deep Back Translation) can use external models (such as M2M100 that we are using) or APIs (e.g., Google Translator), but we emphasize that the use of GPU can significantly accelerate the processing of the algorithm with external models (not with APIs). Finally, the T5 Parrot and BART ProtAugment paraphrasing models are recommended for GPU usage, and have lower memory cost and higher speed than translation-based models.

5.3. Diversity generation

Essential features in DA algorithms are their generation of lexical diversity and semantic fidelity. DA algorithms need to generate lexical diversity to increase the generalization power of trained ML models. If diversity generation is low, the algorithms may not perform satisfactorily. On the other hand, algorithms need to present semantic fidelity, in order to be able to take advantage of the labels of the original text, when pertinent. Note, however, that the DA algorithms assume that the original data do not have biases or errors; otherwise, errors and biases will be reinforced. For example, if a text is generated with a positive sentiment label, but in reality this text may have a negative meaning, this error will greatly disrupt the training of sentiment classifier models.

The assessment of lexical diversity and semantic fidelity is complicated by the absence of objective metrics for these criteria. The ParaBank work [88] demonstrated that an exciting metric for lexical diversity is BLUE without brevity punishment. BLUE is a metric widely used for translation problems whose value is higher when the translated phrase is the same or very similar to the reference phrase. BLUE can therefore be indicative of lexical diversity because the smaller the BLUE between two texts, the more they are different. Thus, we can collect the BLUE measure between the original and synthetic texts, and the lower the value, the more diverse the two texts are. We removed the brevity penalty, as in the ParaBank work [88], so as not to give short texts an advantage in the evaluation.

In the semantic fidelity criterion, we use the cosine similarity in the two sentences represented by embeddings [89]. We decided to use the trained Albert model to find semantic similarities between the texts. Albert manages to generate embedding representations in a latent space that, with the similarity of the cosine, indicate how similar the contents of the texts are. Thus, the greater the similarity of cosines (close to 1), the greater the congruence of meaning between the two sentences.

Table 5 shows the results of the analysis regarding lexical diversity and semantic fidelity.

The results show an intriguing behavior of the algorithms, and we can group them according to their behavior. Initially, we can observe that some algorithms have a more "conservative" behavior in lexical diversity in order to maintain greater semantic fidelity (high BLUE and high cosine similarity). The T5 Parrot and BART ProtAugment paraphrasers are good examples, as semantic fidelity is a vital requirement for paraphrasers. The BT and

Table 4
Computational requirements and processing time comparison between all algorithms. it/s stands for the number of iterations per second — the higher, the faster the algorithm.

Augmenters	External files	External models	Requires GPU	Memory	Processing velocity
Keyboard error	No	No	No	–	950.53 it/s
EDA	Optional (Synonym file)	No	No	–	155.71 it/s
Embedding (Glove)	Yes (Glove 200d)	No	No	0.6 GB	0.46 it/s
BERT Aug	No	Yes (Bert-uncased)	Yes	1.5 GB	1.08 it/s
SSMBA	No	Yes (Bert-uncased)	Yes	0.9 GB	9.17 it/s
PPDB	Yes (ppdb-en-all)	No	No	7.5 GB	122.29 it/s
Back translation	No	Yes or API	Yes (using model)	3.5 GB	0.94 it/s
Deep back translation	No	Yes or API	Yes (using model)	3.5 GB	0.65 it/s
T5 Parrot	No	Yes (T5-paraphraser-parrot)	Yes	1.3 GB	1.43 it/s
BART ProtAugment	No	Yes (bart-protaugment)	Yes	0.8 GB	5.23 it/s
LAMBADA	No	Yes (gpt2-uncased)	Yes (training)	1.5 GB	0.89 it/s

Table 5
Comparison of lexical diversity (more diversity with low BLUE) and semantic fidelity (more fidelity with cosine similarity close to 1). BLUE w/o bp is BLUE without brevity punishment.

Dataset	SemEval (Ironic dataset)		TREC		Emotions detection	
Augmenter	BLUE w/o bp	Cosine Sim	BLUE w/o bp	Cosine Sim	BLUE w/o bp	Cosine Sim
Keyboard error	38.05	0.544	37.4	0.591	40.89	0.636
EDA	28.81	0.671	26.74	0.728	29.93	0.76
Embedding	39.08	0.607	37.6	0.684	42.02	0.654
BERT Aug	25.23	0.637	23.38	0.694	41.21	0.763
SSMBA	32.62	0.719	27.17	0.741	48.4	0.791
PPDB	49.1	0.728	41.33	0.766	40.9	0.771
Back Translation	48.29	0.844	41.12	0.818	26.93	0.783
Deep Back Translation	54.44	0.851	40.38	0.802	37.41	0.817
T5 Parrot	46.48	0.84	28.8	0.856	46.81	0.859
BART ProtAugment	79.13	0.936	68.71	0.896	83.94	0.951
LAMBADA	9.29	0.28	14.33	0.337	14.33	0.337

DeepBT algorithms also behave like this, which is an expected result since ParaNet [61] and ParaBank [88] are translation-based paraphrasers — i.e., translations are basic paraphrasers tools. Another algorithm with apparently conservative behavior is PPDB, as it was also modeled to behave like a paraphraser.

In contrast, LAMBADA is an algorithm of high lexical diversity and low semantic fidelity (low BLUE and low cosine similarity). This result may be due to the use of the GPT language model, since it writes new texts using only part of the original text. Therefore, GPT does not necessarily write a text with the same meaning as the original, even using the discriminator classifier to exclude lower quality sentences. The classifier avoids sentences generated with an unwanted label, but it does not prevent a change in meaning. Thus, LAMBADA is a high-risk, high-return strategy in relation to lexical diversity, but at the cost of lower semantic fidelity.

Finally, the rest of the algorithms are neither lexically diverse nor have high semantic fidelity. In this group, EDA and BERT Augmenter stand out for better balancing the compromise between the two metrics, which can be a competitive advantage in terms of performance. These metric values are very consistent with our first impressions analyzing the synthetic texts generated in Section 5.1. Thus, the metrics demonstrate that the algorithms present the same behaviors as those inferred from the qualitative analysis of the texts.

These metrics and assessments are relevant to verify the impacts of augmentation on the data. Figs. 5 and 6 illustrate in two different ways, t-SNE and LDA, how the algorithms act in the Emotion Recognition dataset in relation to lexical diversity and semantic similarity. T-Distributed Stochastic Neighbor Embedding (T-SNE) is a method specifically used for visualization purposes only. It is well suited for the visualization of high-dimensional datasets. T-SNE is a non-linear data viewer. Linear Discriminant Analysis (LDA) is a method that can also be used to reduce the dimensionality of a database. It focuses on maximizing the separability among known categories by creating projections that make it easier to visualize these separations.

The two figures show the augmented Emotion Recognition dataset views for EDA, Embedding Aug, and LAMBADA (top) and PPDB, Deep Back Translation, and BART ProtAugment (bottom). The former were chosen because they present great lexical diversity but less semantic similarity, and the latter show the opposite with little lexical diversity but great semantic similarity.

In Fig. 5 we notice that the data generated by the DA algorithms (small diamonds) are farther from the original points (large circles of the same colors) in the algorithms at the top, especially LAMBADA. On the other hand, for the algorithms at the bottom, it is noticeable that the generated data is practically in the same positions as the original data, especially BART ProtAugment. This characteristic can impact the generalization capacity of models.

In Fig. 6 the LDA projection in two dimensions shows that the bottom algorithms present a better separability between the classes. The effect of augmentation can be observed as follows: the more spread out the generated data are (small diamonds), invading clusters of different colors, the greater lexical diversity and less semantic similarity they will have in relation to the original data. In particular, it is observed that BART ProtAugment presents very little lexical diversity, but high semantic similarity, corroborating the numerical data presented in Table 5.

The two visual analyses show a major challenge for DA techniques, which is to insert lexical diversity into a dataset, while maintaining semantic similarity to preserve the labels of the original data. This compromise between diversity and class fidelity is not simple to resolve. Depending on the problem to which the model is applied and the amount of data available, an algorithm that generates greater lexical diversity may become more recommendable than algorithms that generate greater semantic similarity, and vice versa. There is no single DA technique that should be recommended for all cases. When the designer does not have the intuition of the most appropriate AD technique for his data and problem, meta-learning techniques can be used, at the expense of a higher computational cost (see Section 3.3).

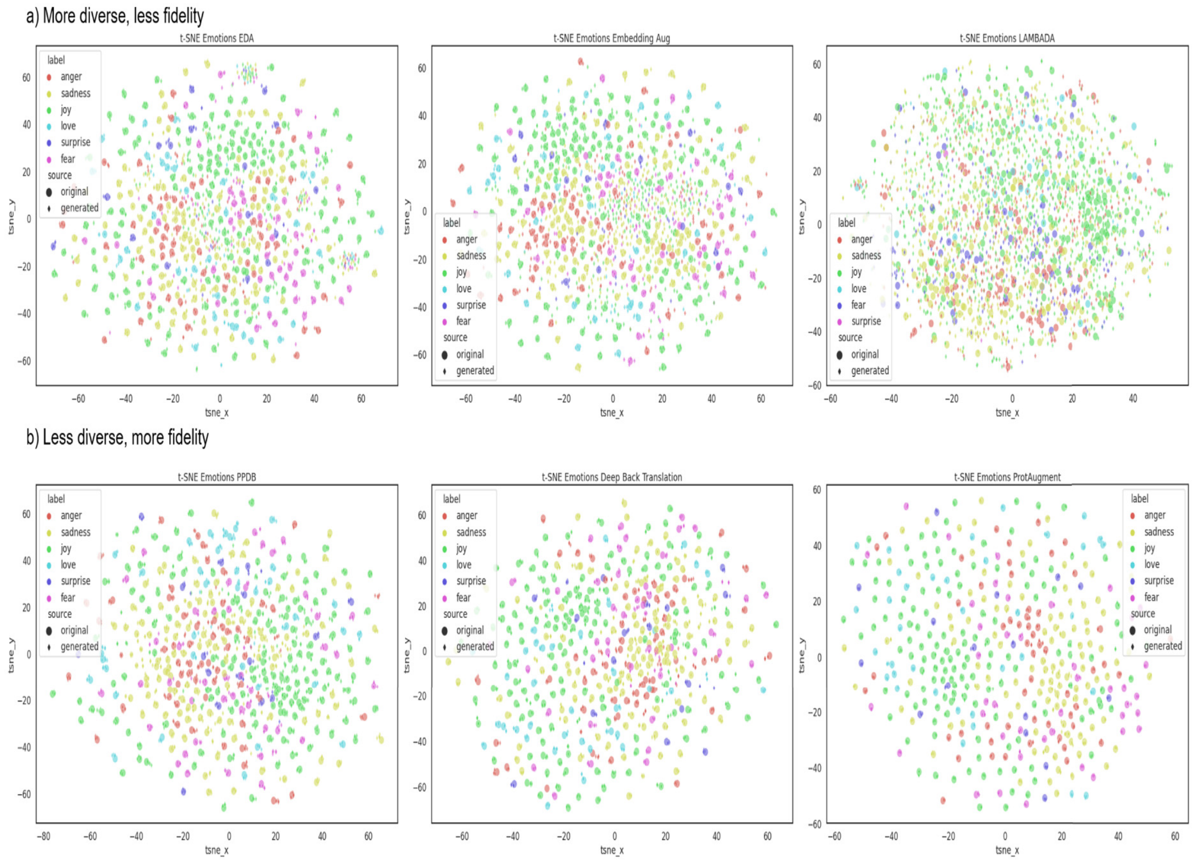

Fig. 5. T-SNE projection of the Emotion Recognition dataset with application of DA algorithms. The three upper graphs (EDA, Embedding Aug and LAMBADA) are from algorithms with profiles with great lexical diversity but less semantic similarity, while the three lower graphs (PPDB, Deep Back Translation and BART ProtAugment) are algorithms with little lexical diversity but great semantic similarity. The interest here lies in observing the arrangement of the generated data (small diamonds) in relation to the original data (large circles of the same color as the generated diamonds). Those with greater lexical diversity are more spread out in space. Those with greater semantic similarity present the diamonds on the circles of the same color. Note that LAMBADA presents greater lexical diversity and less semantic similarity.

5.4. Performance

The last comparison is about the performance gain when using these DA algorithms. It is relevant to review some tests performed in these comparisons, described in Section 4.2. We separate the training dataset into portions corresponding to percentages of the set (10%, 20%, 30%, 40%, 50%, 75%, and 100%), then applied the algorithms to generate five texts for each original observation. The idea is to evaluate the performance of the system in terms of the amount of original data. Finally, we join the original dataset with the synthetic dataset to train a DistilBert classifier. We compute the F1-macro metric in the test for comparison. We repeated this procedure ten times for each algorithm and percentage of the original set.

In addition to applying DA algorithms, we also perform classifier training without the use of DA. In this way, the performance of the classifier without DA serves as a baseline of comparison for the performance gain.

Tables 6–8 show the average results of F1-macro in the classification of the test dataset in SemEval 2018, TREC, and Emotion Recognition dataset, respectively. The first row of each table is the algorithm's performance without applying DA, serving as a baseline for comparison. The tables highlight the algorithms with the best performance with $\star$ and the second and third best with $\ast$. The three worst algorithms are † highlighting.

A clear conclusion from this analysis is that DA is much more decisive and influential in data-scarce scenarios (10%, 20% and 30%) for the SemEval and TREC datasets. As available data increase, the influence of DA decreases. For SemEval, this performance drop ends at 50% and then increases again. However, for the TREC dataset, the performance gain decrease continues up to 100% of the available data. In contrast to this behavior, the Emotion Recognition dataset has a performance increase as data availability increases, likely due to the low amount of original data in training.

The second conclusion is that it is not simple to indicate which algorithm has the best performance gain. By carefully analyzing the results, we can see that the performance of the algorithms varies significantly, both as a function of data availability and the datasets used.

For example, paraphrasers perform best on the Emotion Recognition and TREC datasets, especially BART ProtAugment, which has the most prominent performance gain of all algorithms on a few different data percentages. However, these same algorithms that stand out in the two datasets present the worst performances in the SemEval 2018 dataset. It is not easy to justify the difference in performance of the paraphrases in the different datasets. The choice of DA probably depends much more on the nature of the problem in which the DA will be used. This conclusion is similar to the work of Data Boosting [72] which showed that Data Boosting does not work well on some datasets as metaphor identification.

An interesting behavior is the LAMBADA algorithm. The algorithm performs better in situations with greater data scarcity (10%, 20% or 30%), ranking among the top three algorithms in

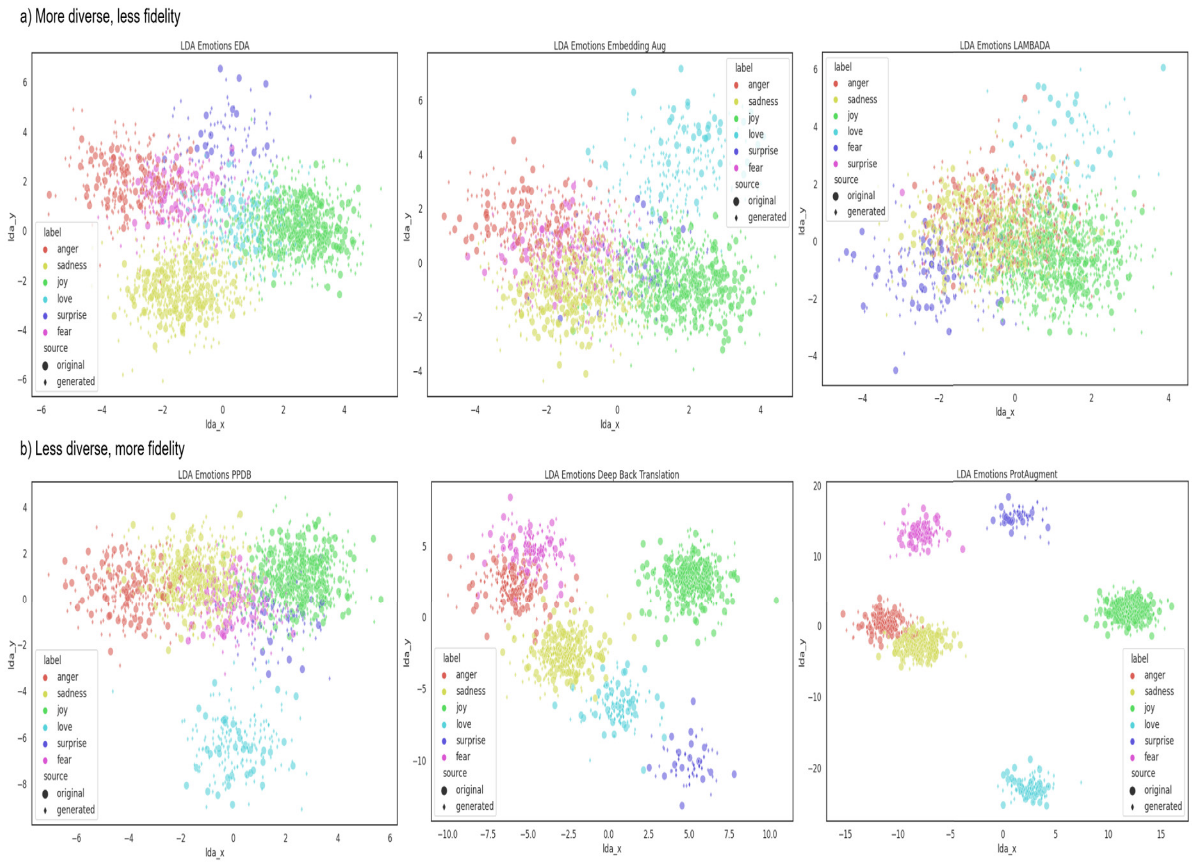

Fig. 6. LDA projection of the Emotion Recognition dataset with application of DA algorithms. The three upper graphs (EDA, Embedding Aug and LAMBADA) are from algorithms with profiles with great lexical diversity but less semantic similarity, while the three lower graphs (PPDB, Deep Back Translation and BART ProtAugment) are algorithms with little lexical diversity but great semantic similarity. The effect of augmentation can be observed in the scattering of the generated data (small diamonds) and the distribution of the original data (large circles). While algorithms that generate data with greater lexical diversity show significant overlaps between classes (i.e., clusters of different colors), those with greater preservation of semantics show very well-defined clusters. Note that BART ProtAugment has little lexical diversity (the diamonds are tightly clustered) and high semantic similarity (diamonds are on those circles that have the same color).

Table 6
Comparison Performance with average and standard deviation (avg ± sd) of F1-macro for 10 executions in SemEval 2018 dataset (irony detection). The first line is the algorithm's performance without the application of DA (Baseline). We have highlighted the best performing algorithms with ⋆ and the second and third best with ∗ in each percentage. On the other hand, the three worst algorithms are highlighted with †.

Augmenter	Percentage						
	10%	20%	30%	40%	50%	75%	100%
W/o Aug (Baseline)	54.4% ± 6.1%	60.2% ± 4.2%	62% ± 1.9%	62.8% ± 2.9%	63.8% ± 1.8%	64.3% ± 1.9%	65.3% ± 1.2%
Keyboard error	60.8% ± 1.7%	62.9% ± 2%†	63.8% ± 1.5%†	64.8% ± 1.6%∗	65.9% ± 1.3%∗	69.1% ± 0.8%*	69.4% ± 1.5%∗
EDA	62.8% ± 1.9%∗	64.5% ± 1.3%	64.4% ± 1.7%	64.7% ± 1.3%	65.5% ± 1.7%	67.7% ± 1.2%	68.8% ± 1.9%
Embedding	61.7% ± 1.2%	65.1% ± 0.8%	65.4% ± 1.5%∗	65.5% ± 1.6%∗	65.6% ± 1.7%	68.3% ± 1.3%∗	68.1% ± 1.6%
BERT Augment	63.5% ± 1.4%*	65.3% ± 1.9%∗	65.6% ± 1.5%∗	67.2% ± 1.5%*	68.6% ± 1.4%*	68.3% ± 1.0%∗	70.0% ± 1.5%∗
SSMBA	61.7% ± 2.1%	64.1% ± 0.7%	66.1% ± 1.7%*	64.5% ± 1.1%	66.5% ± 1.6%∗	69.1% ± 0.8%*	70.1% ± 1.3%*
PPDB	61.4% ± 2.4%	63.6% ± 1.3%†	64.8% ± 1.3%	64.1% ± 1.3%	65.0% ± 1.2%	65.8% ± 0.7%	66.0% ± 1.1%†
Back translation	60.5% ± 1.5%†	65.0% ± 1.6%	64.1% ± 1.3%	64.0% ± 2.0%†	64.8% ± 1.6%	66.6% ± 1.1%	68.6% ± 2.2%
Deep back translation	60.9% ± 1.5%	64.0% ± 1.8%	65.0% ± 1.3%	64.4% ± 1.1%	64.0% ± 0.6%†	66.9% ± 1.0%	67.3% ± 0.8%
T5 Parrot	58.7% ± 5.8%†	65.1% ± 1.0%∗	65.1% ± 1.4%	63.6% ± 1.8%†	64.5% ± 1.2%†	66.4% ± 1.4%†	67.1% ± 1.7%†
BART ProtAugment	60.2% ± 1.9%†	63.2% ± 1.5%†	62.7% ± 1.1%†	63.1% ± 2.0%†	64.4% ± 0.9%†	66.1% ± 1.3%†	67.6% ± 1.7%
LAMBADA	62.5% ± 1.5%∗	66.5% ± 0.7%*	63.6% ± 0.7%†	64.2% ± 0.8%	65.3% ± 0.9%	65.4% ± 0.5%†	66.4% ± 1.3%†

performance gain. With higher data availability (75% or 100%), LAMBADA did not achieve the same performance compared to other algorithms, being closer to the worst. This behavior may be a consequence of the strategy of bringing a lot of lexical diversity without compromising fidelity to the original meaning. In situations with little data, more diversity may be more relevant than in situations with more data available.

Both Contextual BERT Augmenter and SSMBA present outstanding results over the other algorithms. The two algorithms are closer to being the best than the worst in the different percentage situations and in the three datasets. This result may indicate an excellent competitive advantage of these algorithms over the others, suggesting that they are the most robust algorithms in terms of the amount of data available and the nature of the problem.

Finally, a very relevant conclusion is that the performance of algorithms with a lot of data (100%) does not show such a significant difference. This result is vital, because if it is necessary to apply DA in a dataset with a lot of available data, as the performance difference was not so relevant, the computationally

Table 7
Comparison Performance with average and standard deviation (avg ± sd) F1-macro 10 executions for TREC dataset. The first line is the algorithm's performance without the application of DA (Baseline). We have highlighted the best performing algorithms with $\star$ and the second and third best with $*$ in each percentage. On the other hand, the three worst algorithms are highlighted with $\dagger$.

Augmenter	Percentage						
	10%	20%	30%	40%	50%	75%	100%
W/o Aug (Baseline)	19.5% ± 5.5%	29.7% ± 5.2%	49.3% ± 5.4%	60.6% ± 2.9%	65.8% ± 2.4%	70.2% ± 0.8%	71.8% ± 0.8%
Keyboard error	56.2% ± 3.2%	68.7% ± 1.3%	73.9% ± 5.2%	83.1% ± 5.7%	86% ± 3.2%	89.6% ± 0.9%$*$	90.3% ± 0.7%*
EDA	57.6% ± 3.4%	67.3% ± 1.3%†	73% ± 5.1%	83.9% ± 1.7%	86.2% ± 1.2%	88.8% ± 0.9%	89.3% ± 0.7%†
Embedding	55.4% ± 4.1%†	67.6% ± 1%†	72.2% ± 3.8%	74.6% ± 6.4%	85.8% ± 0.6%	88% ± 1.3%	89% ± 1.1%†
BERT Aug	62.4% ± 1.6%$*$	68.9% ± 3.3%	78.6% ± 6.1%	85.2% ± 0.9%	86.4% ± 1.2%	88.5% ± 0.7%	89.7% ± 0.7%
SSMBA	63.6% ± 2.1%$*$	68.6% ± 1.0%	82.3% ± 4.8%$*$	85.6% ± 0.9%$*$	86.6% ± 0.9%$*$	87.6% ± 1.6%	89.9% ± 0.9%
PPDB	56.1% ± 2.9%†	56.7% ± 3.2%†	75.4% ± 4.8%	71.9% ± 1.2%†	73.9% ± 4.6%†	89.8% ± 0.8%$*$	90.1% ± 0.9%$*$
Back translation	56.7% ± 6.6%	69.3% ± 0.8%$*$	71.3% ± 1%†	73.1% ± 0.9%†	74.8% ± 3.2%†	85.2% ± 6%†	89.5% ± 1.2%
Deep back translation	52.8% ± 6.9%†	68% ± 1.3%	71.7% ± 1.1%†	72.6% ± 0.8%†	74% ± 0.9%†	82.2% ± 4.4%†	89.8% ± 1%
T5 Parrot	62.4% ± 2.2%$*$	68.4% ± 1%	81.8% ± 4.2%$*$	86.2% ± 1%$*$	87.2% ± 0.8%$*$	88.7% ± 0.8%	89.6% ± 0.6%
BART ProtAugment	63.9% ± 2%*	70% ± 0.6%*	83.5% ± 4.8%*	88.4% ± 1.1%*	88.4% ± 1%*	89.8% ± 1%*	90% ± 1%$*$
LAMBADA	59.3% ± 1.3%	69.3% ± 0.8%$*$	72.3% ± 0.6%†	77.6% ± 4.9%	85.5% ± 0.9%	87.1% ± 1%†	88.9% ± 0.8%†

Table 8
Comparison Performance with average and standard deviation (avg ± sd) F1-macro 10 executions for Emotions Detection dataset. The first line is the algorithm's performance without the application of DA (Baseline). We have highlighted the best performing algorithms with $\star$ and the second and third best with $*$ in each percentage. The three worst performing algorithms, on the other hand, were highlighted with $\dagger$.

Augmenter	Percentage						
	10%	20%	30%	40%	50%	75%	100%
W/o Aug (Baseline)	9.7% ± 2%	9.6% ± 2.6%	10% ± 2.8%	11.8% ± 4.2%	14.6% ± 6.1%	22.6% ± 1.2%	24.3% ± 0.3%
Keyboard error	12.2% ± 3.7%†	21.3% ± 4.6%†	25.1% ± 1.7%	32.4% ± 7.6%	44.6% ± 5.8%	65.3% ± 5.3%	78.5% ± 2.2%
EDA	18.3% ± 3.9%*	23.7% ± 0.3%	28.8% ± 4.4%	45.3% ± 4.2%	56.7% ± 3.4%	75.4% ± 3%	82.3% ± 0.7%
Embedding	13.5% ± 4.3%	23.7% ± 0.4%	29% ± 3.7%	43.8% ± 5.3%	53% ± 3.6%	74.5% ± 2%	82.5% ± 0.9%$*$
BERT Aug	14.1% ± 4%	23.9% ± 0.3%$*$	33.3% ± 6.5%$*$	43.1% ± 6%	58% ± 2.7%	76.7% ± 2.1%$*$	81.7% ± 0.7%
SSMBA	16.6% ± 4.5%$*$	24.1% ± 0.2%$*$	29.8% ± 3.8%	46.3% ± 2.7%$*$	61.6% ± 2.4%$*$	77.0% ± 1.0%$*$	83.1% ± 0.7%$*$
PPDB	13.1% ± 4.1%	24.4% ± 2.9%*	33.9% ± 4%$*$	45.8% ± 3.3%$*$	58.5% ± 3.5%$*$	72.2% ± 2.2%	79.8% ± 1.3%
Back translation	13.9% ± 4.4%	21.6% ± 2.3%†	23.8% ± 0.5%†	26.9% ± 3.2%†	35.3% ± 3.1%†	62.2% ± 3.6%†	70% ± 2.2%†
Deep back translation	9.5% ± 2.5%	20.8% ± 4.3%†	23.8% ± 0.5%†	24.8% ± 0.7%†	29.9% ± 4.6%†	55.1% ± 3.2%†	7.9% ± 4%†
T5 Parrot	12.4% ± 4.5%†	23.5% ± 0.4%	27% ± 3.1%	41.9% ± 5.5%	51% ± 2.7%	69.2% ± 1.9%	78.9% ± 2.5%
BART ProtAugment	16.1% ± 3.8%$*$	23.7% ± 0.5%	34.3% ± 6%*	50.9% ± 2.8%*	63.5% ± 3.2%*	79.7% ± 1.6%*	83.7% ± 0.8%*
LAMBADA	12.0% ± 3.9%†	22.9% ± 0.7%	23.2% ± 0.8%†	24.4% ± 0.1%†	33.8% ± 5.2%†	56.9% ± 2.1%†	71.4% ± 1.1%†

lighter algorithms (such as EDA and Keyboard Error) become much more attractive in these situations.

6. Conclusions and future work

In this work, we carried out an in-depth study of DA techniques in NLP, performing a systematic and comparative evaluation of them. We summarize below our main findings and contributions:

1. **NLP DA literature review**: We carried out a comprehensive literature review on the subject. More importantly, instead of just providing a general overview of several works on the DA topic, we condense and highlight the main research gaps found in the literature.
2. **Systematic assessment of NLP DA techniques**: by carrying out an in depth study of various NLP DA methods, we add to the state-of-the-art on NLP DA techniques an unprecedented systematic comparative study, addressing research gap 1 discussed in Section 3.4.
3. **Comparative analysis of DA algorithms in terms of computational efforts**: This analysis makes it possible to establish compromises between the use of DA algorithms and the gain from their application. Our results show that the use of more expensive DA algorithms does not offset the computational costs if there is a more significant amount of data available. This contribution covers in part the research gap 2 described in Section 3.4.
4. **Assessing the impact of generating lexical diversity and semantic fidelity**: Our studies demonstrate a trade-off between text diversity and maintaining its semantic meaning in the generation of synthetic data. In addition, our tests show that DA algorithms have characteristic behavior profiles that impact their performance. This contribution covers the remaining of research gap 2 described in Section 3.4.
5. **Comparison of performance gain using DA**: Our performance gain comparisons allow us to analyze the advantages of using DA. It was possible to verify that the use of DA algorithms is more critical the greater the scarcity of data. We emphasize that algorithms with low lexical diversity but high semantic fidelity present average performances, and algorithms with high lexical diversity and low semantic fidelity, such as LAMBADA, are more influential in low data availability. EDA and BERT Argumenter stand out not only for their good balance between lexical diversity and semantic fidelity, but also for their good performance in various scenarios, both with high and low data availability. This contribution covers in part the research gap 2 described in Section 3.4.
6. **Accessibility and Reproducibility:** We provide or clearly indicate the source of the codes and databases used in this study in order to contribute to the reproducibility and transparency of our experiments with DA algorithms. Also, following good research practices, we explicitly report variation among results of different executions of the same experiment. All our code is publicly available at GitHub repository https://github.com/lucasfaop/survey_text_augmentation, with helpful library NLPAUG [84]. This contribution covers research gap 3 described in Section 3.4.

Finally, Table 9 summarizes the advantages and disadvantages of each of the chosen and tested algorithms.

Table 9
Pros and Cons of all algorithms.

Augmenter	Pros	Cons
Keyboard error	Easy and quick to use. Low computational cost. Noise improves generalization.	Generated texts are difficult to read. May lose all textual meaning.
EDA	Easy and quick to use. Low computational cost. Noise improves generalization.	Generated texts are difficult to read. May lose all textual meaning.
Embedding	Easy-to-check word changes. Generated phrases retain syntax.	Can insert out-of-context words. Slow processing.
BERT Augmenter	Contextual changes. Robustness.	Needs GPU. Slow processing.
SSMBA	Contextual changes. Robustness.	Needs GPU. Difficult to use.
PPDB	Easy and quick to use. Changes in words or expressions. Fast processing.	High memory use. Performance is not as good as other algorithms.
Translation approaches (Back translation and deep back translation)	Keeps the meaning of the sentence. Generates sentences with a lot of semantic and syntactic sense. Changes in words or expressions.	Slow processing. Low diversity generation. Translation dependent. Needs GPU when using a translation model. Some APIs may be paid or with limited use.
Paraphrasers (T5 Parrot and BART ProtAugment)	Much lighter than translator models. Preserves semantics and syntax. Easy to use.	Needs GPU. Low diversity generation. Dependent on the quality of the paraphraser's external knowledge.
LAMBADA	High level of diversity generation. In general, it generates very clear sentences. Better results with low data availability.	High risk of changing label. Difficult to reuse Needs two training models. Needs GPU.

In summary, in this work we addressed several of the most important research gaps found in the NLP DA literature. With this findings, we hope to encourage further use of DA as an auxiliary method in NLP tasks and to raise NLP DA techniques to greater maturity. In addition, we hope that this work can serve as an inspiration and a starting point for future studies on the subject, in order to address some of the remaining open questions on the subject, which we summarize below and that can be explored in future work:

1. **Scarcity of mixed-example research within the NLP domain:** despite unquestionable success in image processing tasks [27,40,41,90], to the best of our knowledge there are only four works applying this class of method to NLP [40, 42–44], some of which are still under revision at the time of this writing. Therefore, little is known regarding the efficacy of applying this family of DA techniques to NLP problems.
2. **Influence of dataset size on DA effectiveness**: there is abundant evidence that the relative gains brought by DA methods decrease rapidly with increasing dataset size — that is, the gains brought by DA methods tend to be relevant in smaller datasets (or in fractions of the original dataset) and marginal on larger datasets (or when using the original full dataset) [24,54,60,70]. Further research should deepen our understanding about this behavior and propose methods where this trend, if present, is less pronounced.
3. **Effectiveness of combined use of different DA techniques**: little work has been done to understand to what extent – if any – the combined use of distinct DA methods could benefit model performance. Two of the few works to perform such an investigation are the works of [91] – indicating gains when combining Back-Translation and Switch-Out – and of [70] – showing the benefits of the compounded use of DA techniques for Named Entity Recognition tasks. Clearly, if beneficial, the combined use of different DA techniques could bring faster advances to the subject than developing new DA methods from scratch, which constitutes a reasonable motivation for further research.
4. **Policy optimization algorithm with lexical diversity and semantic fidelity**: Policy optimization algorithms could be an exciting proposal whose reward could balance diversity and semantic fidelity, similar to Data Boost [72] or AutoAugment [19]. Thus, this new algorithm could adapt to more conservative behaviors such as translation-based methods, or riskier ones such as LAMBADA, as appropriate.
5. **Theoretical framework and systematic studies for NLP DA techniques**: some – if not most – of the proposed NLP DA techniques are derived from heuristics, domain knowledge and common sense, instead of formally derived. Research would greatly profit from the development of a formal theoretical framework to support and explain DA methods in NLP. This could allow for better insights into such methods and a proper understanding of the similarities and differences between each technique. One of the few works to derive its proposal – a technique called Switch-Out – from a formal probabilistic framework is the work of [91], proving alongside that some of the previously proposed DA NLP methods were special cases of their more general proposition. In regard to systematic studies, [92] constitutes an interesting work that tries to investigate the issue of generalization in NLP from a very structured and methodical approach. In this work, the authors design a series of toy learning problems to inspect the effectiveness of DA in NLP when training on counterexamples. We strongly believe that approaching DA NLP research using more formal theoretical tools (like [91]) or more systematic procedures (like [92]) constitute promising research paths.

CRediT authorship contribution statement

Lucas Francisco Amaral Orosco Pellicer: Conceptualization, Methodology, Software, Investigation, Writing – original draft.

Taynan Maier Ferreira: Conceptualization, Methodology, Software, Investigation, Writing – original draft. **Anna Helena Reali Costa:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the Coordination for the Improvement of Higher Education Personnel (CAPES, Finance Code 001) and the National Council for Scientific and Technological Development (CNPq grant N. 310085/2020-9), Brazil. This work was carried out in part at the *Centro de Ciência de Dados* (C^2D) of the *Escola Politécnica* at USP, with the support of Itaú Unibanco S.A., and at the Center for Artificial Intelligence (C4AI-USP), with support from the State of Sao Paulo Research Foundation (FAPESP grant N. 2019/07665-4) and IBM Corporation.

References

[1] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, The MIT Press, 2016.

[2] C. Shorten, T. Khoshgoftaar, A survey on image data augmentation for deep learning, J. Big Data 6 (2019) http://dx.doi.org/10.1186/s40537-019-0197-0.

[3] S.Y. Feng, V. Gangal, J. Wei, S. Chandar, S. Vosoughi, T. Mitamura, E.H. Hovy, A survey of data augmentation approaches for NLP, 2021, arXiv abs/2105.03075.

[4] L. Taylor, G. Nitschke, Improving deep learning with generic data augmentation, in: 2018 IEEE Symposium Series on Computational Intelligence (SSCI), 2018, pp. 1542–1547.

[5] A. Mikołajczyk, M. Grochowski, Data augmentation for improving deep learning in image classification problem, in: 2018 International Interdisciplinary PhD Workshop (IIPhDW), 2018, pp. 117–122.

[6] F. Bao, M. Neumann, T. Vu, Cyclegan-based emotion style transfer as data augmentation for speech emotion recognition, in: Proc. Interspeech 2019, 2019, pp. 2828–2832, http://dx.doi.org/10.21437/Interspeech.2019-2293.

[7] J. Salamon, J.P. Bello, Deep convolutional neural networks and data augmentation for environmental sound classification, IEEE Signal Process. Lett. 24 (3) (2017) 279–283.

[8] Q. Wen, L. Sun, X. Song, J. Gao, X. Wang, H. Xu, Time series data augmentation for deep learning: A survey, 2020, arXiv:2002.12478.

[9] S. Kobayashi, Contextual augmentation: Data augmentation by words with paradigmatic relations, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 452–457, http://dx.doi.org/10.18653/v1/N18-2072.

[10] A. Sugiyama, N. Yoshinaga, Data augmentation using back-translation for context-aware neural machine translation, in: Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 35–44, http://dx.doi.org/10.18653/v1/D19-6504.

[11] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: Proceedings of the 25th International Conference on Neural Information Processing Systems - Vol. 1, NIPS '12, Curran Associates Inc., Red Hook, NY, USA, 2012, pp. 1097–1105.

[12] J.T. Springenberg, A. Dosovitskiy, T. Brox, M.A. Riedmiller, Striving for simplicity: The all convolutional net, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings, 2015, URL: http://arxiv.org/abs/1412.6806.

[13] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778, http://dx.doi.org/10.1109/CVPR.2016.90.

[14] T. Dao, A. Gu, A. Ratner, V. Smith, C. De Sa, C. Re, A kernel theory of modern data augmentation, in: K. Chaudhuri, R. Salakhutdinov (Eds.), Proceedings of the 36th International Conference on Machine Learning, in: Proceedings of Machine Learning Research, vol. 97, PMLR, Long Beach, California, USA, 2019, pp. 1528–1537, URL: http://proceedings.mlr.press/v97/dao19b.html.

[15] D. Zhao, G. Yu, P. Xu, M. Luo, Equivalence between dropout and data augmentation: A mathematical check, Neural Netw.: Off. J. Int. Neural Netw. Soc. 115 (2019) 82–89.

[16] K.R. Konda, X. Bouthillier, R. Memisevic, P. Vincent, Dropout as data augmentation, 2015, arXiv abs/1506.08700.

[17] A. Hernández-García, P. König, Further advantages of data augmentation on convolutional neural networks, in: V. Kůrková, Y. Manolopoulos, B. Hammer, L. Iliadis, I. Maglogiannis (Eds.), Artificial Neural Networks and Machine Learning – ICANN 2018, Springer International Publishing, Cham, 2018, pp. 95–103.

[18] S. Edunov, M. Ott, M. Auli, D. Grangier, Understanding back-translation at scale, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 489–500, http://dx.doi.org/10.18653/v1/D18-1045.

[19] E.D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, Q.V. Le, AutoAugment: Learning augmentation policies from data, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 113–123, http://dx.doi.org/10.1109/CVPR.2019.00020.

[20] D. Ho, E. Liang, X. Chen, I. Stoica, P. Abbeel, Population based augmentation: Efficient learning of augmentation policy schedules, in: K. Chaudhuri, R. Salakhutdinov (Eds.), Proceedings of the 36th International Conference on Machine Learning, in: Proceedings of Machine Learning Research, vol. 97, PMLR, Long Beach, California, USA, 2019, pp. 2731–2741, URL: http://proceedings.mlr.press/v97/ho19b.html.

[21] K. Cortis, A. Freitas, T. Daudert, M. Huerlimann, M. Zarrouk, S. Handschuh, B. Davis, SemEval-2017 Task 5: Fine-Grained Sentiment Analysis on Financial Microblogs and News, in: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), Association for Computational Linguistics, Stroudsburg, PA, USA, 2017, pp. 519–535, http://dx.doi.org/10.18653/v1/S17-2089.

[22] S. Mohammad, F. Bravo-Marquez, M. Salameh, S. Kiritchenko, SemEval-2018 task 1: Affect in tweets, in: Proceedings of the 12th International Workshop on Semantic Evaluation, Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 1–17, http://dx.doi.org/10.18653/v1/S18-1001.

[23] V. Basile, C. Bosco, E. Fersini, D. Nozza, V. Patti, F.M. Rangel Pardo, P. Rosso, M. Sanguinetti, SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter, in: Proceedings of the 13th International Workshop on Semantic Evaluation, Association for Computational Linguistics, Minneapolis, Minnesota, USA, 2019, pp. 54–63, http://dx.doi.org/10.18653/v1/S19-2007.

[24] T.M. Ferreira, A.H.R. Costa, Deepbt and NLP data augmentation techniques: A new proposal and a comprehensive study, in: R. Cerri, R.C. Prati (Eds.), Intelligent Systems - 9th Brazilian Conference, BRACIS 2020, Rio Grande, Brazil, October 20-23, 2020, Proceedings, Part I, in: Lecture Notes in Computer Science, vol. 12319, Springer, 2020, pp. 435–449, http://dx.doi.org/10.1007/978-3-030-61377-8_30.

[25] T.M. Mitchell, Machine Learning, McGraw-Hill, New York, 1997.

[26] V.N. Vapnik, An overview of statistical learning theory, IEEE Trans. Neural Netw. 10 (5) (1999) 988–999.

[27] H. Zhang, M. Cisse, Y.N. Dauphin, D. Lopez-Paz, Mixup: Beyond empirical risk minimization, in: International Conference on Learning Representations, 2018, URL: https://openreview.net/forum?id=r1Ddp1-Rb.

[28] O. Chapelle, J. Weston, L. Bottou, V. Vapnik, Vicinal risk minimization, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems 13, MIT Press, 2001, pp. 416–422, URL: http://papers.nips.cc/paper/1876-vicinal-risk-minimization.pdf.

[29] K.P. Murphy, Machine Learning : A Probabilistic Perspective, MIT Press, Cambridge, Mass. [u.a.], 2013.

[30] C.M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag, Berlin, Heidelberg, 2006.

[31] S. Chen, E. Dobriban, J.H. Lee, A group-theoretic framework for data augmentation, 2019, arXiv:1907.10905.

[32] M. Bayer, M.-A. Kaufhold, C. Reuter, A survey on data augmentation for text classification, ACM Comput. Surv. (2022) http://dx.doi.org/10.1145/3544558, Just Accepted.

[33] S.Y. Feng, V. Gangal, J. Wei, S. Chandar, S. Vosoughi, T. Mitamura, E. Hovy, A survey of data augmentation approaches for NLP, in: Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Association for Computational Linguistics, 2021, pp. 968–988, http://dx.doi.org/10.18653/v1/2021.findings-acl.84, Online. URL: https://aclanthology.org/2021.findings-acl.84.

[34] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 27, Curran Associates, Inc., 2014, pp. 2672–2680, URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

[35] L. Gonog, Y. Zhou, A review: Generative adversarial networks, in: 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2019, pp. 505–510.

[36] Z. Pan, W. Yu, X. Yi, A. Khan, F. Yuan, Y. Zheng, Recent progress on generative adversarial networks (GANs): A survey, IEEE Access 7 (2019) 36322–36333.

[37] R. Gupta, Data augmentation for low resource sentiment analysis using generative adversarial networks, in: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 7380–7384.

[38] Y. Tokozume, Y. Ushiku, T. Harada, Learning from between-class examples for deep sound recognition, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, OpenReview.net, 2018, URL: https://openreview.net/forum?id=B1Gi6LeRZ.

[39] Y. Tokozume, Y. Ushiku, T. Harada, Between-class learning for image classification, in: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, IEEE Computer Society, 2018, pp. 5486–5494, http://dx.doi.org/10.1109/CVPR.2018.00575, URL: http://openaccess.thecvf.com/content_cvpr_2018/html/Tokozume_Between-Class_Learning_for_CVPR_2018_paper.html.

[40] E. Harris, A. Marcu, M. Painter, M. Niranjan, A. Prügel-Bennett, J. Hare, FMix: Enhancing mixed sample data augmentation, 2020, arXiv:2002.12047.

[41] C. Summers, M.J. Dinneen, Improved mixed-example data augmentation, in: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), 2019, pp. 1262–1270.

[42] H. Guo, Y. Mao, R. Zhang, Augmenting data with mixup for sentence classification: An empirical study, 2019, CoRR aba/1905.08941. URL: http://arxiv.org/abs/1905.08941. arXiv:1905.08941.

[43] J. Andreas, Good-enough compositional data augmentation, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2020, pp. 7556–7566, http://dx.doi.org/10.18653/v1/2020.acl-main.676, Online. URL: https://www.aclweb.org/anthology/2020.acl-main.676.

[44] D. Guo, Y. Kim, A. Rush, Sequence-level mixed sample data augmentation, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, 2020, pp. 5547–5552, http://dx.doi.org/10.18653/v1/2020.emnlp-main.447, Online. URL: https://www.aclweb.org/anthology/2020.emnlp-main.447.

[45] Z. Xie, S.I. Wang, J. Li, D. Lévy, A. Nie, D. Jurafsky, A.Y. Ng, Data noising as smoothing in neural network language models, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings, OpenReview.net, 2017, URL: https://openreview.net/forum?id=H1VyHY9gg.

[46] C. Coulombe, Text data augmentation made simple by leveraging NLP cloud APIs, 2018, arXiv abs/1812.04718.

[47] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I.J. Goodfellow, R. Fergus, Intriguing properties of neural networks, in: Y. Bengio, Y. LeCun (Eds.), 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014, URL: http://arxiv.org/abs/1312.6199.

[48] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015, URL: http://arxiv.org/abs/1412.6572.

[49] H. Zhang, H. Zhou, N. Miao, L. Li, Generating fluent adversarial examples for natural languages, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 5564–5569, http://dx.doi.org/10.18653/v1/P19-1559, URL: https://www.aclweb.org/anthology/P19-1559.

[50] J.X. Morris, E. Lifland, J.Y. Yoo, Y. Qi, TextAttack: A framework for adversarial attacks in natural language processing, 2020, arXiv:2005.05909.

[51] W.E. Zhang, Q.Z. Sheng, A. Alhazmi, C. Li, Adversarial attacks on deep-learning models in natural language processing: A survey, ACM Trans. Intell. Syst. Technol. 11 (3) (2020) http://dx.doi.org/10.1145/3374217.

[52] X. Zhang, J. Zhao, Y. LeCun, Character-level convolutional networks for text classification, in: C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 28, Curran Associates, Inc., 2015, pp. 649–657, URL: https://proceedings.neurips.cc/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.

[53] C. Fellbaum, WordNet and wordnets, in: K. Brown (Ed.), Encyclopedia of Language and Linguistics, second ed., Elsevier, Oxford, 2005, pp. 665–670, URL: http://wordnet.princeton.edu/.

[54] J. Wei, K. Zou, EDA: Easy data augmentation techniques for boosting performance on text classification tasks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 6382–6388, http://dx.doi.org/10.18653/v1/D19-1670.

[55] M. Fadaee, A. Bisazza, C. Monz, Data augmentation for low-resource neural machine translation, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Association for Computational Linguistics, Vancouver, Canada, 2017, pp. 567–573, http://dx.doi.org/10.18653/v1/P17-2090, URL: https://aclanthology.org/P17-2090.

[56] W.Y. Wang, D. Yang, 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, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 2557–2563, http://dx.doi.org/10.18653/v1/D15-1306, URL: https://www.aclweb.org/anthology/D15-1306.

[57] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, in: Y. Bengio, Y. LeCun (Eds.), 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings, 2013, URL: http://arxiv.org/abs/1301.3781.

[58] S. Kobayashi, Contextual augmentation: Data augmentation by words with paradigmatic relations, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 452–457, http://dx.doi.org/10.18653/v1/N18-2072, URL: https://aclanthology.org/N18-2072.

[59] X. Wu, S. Lv, L. Zang, J. Han, S. Hu, Conditional BERT contextual augmentation, in: Lecture Notes in Computer Science, Springer International Publishing, 2019, pp. 84–95, http://dx.doi.org/10.1007/978-3-030-22747-0_7.

[60] G.G. Şahin, M. Steedman, Data augmentation via dependency tree morphing for low-resource languages, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 5004–5009, http://dx.doi.org/10.18653/v1/D18-1545, URL: https://www.aclweb.org/anthology/D18-1545.

[61] J. Mallinson, R. Sennrich, M. Lapata, Paraphrasing Revisited with Neural Machine Translation, Association for Computational Linguistics, 2017, http://dx.doi.org/10.18653/v1/e17-1083.

[62] J. Ganitkevitch, B. Van Durme, C. Callison-Burch, PPDB: The paraphrase database, in: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Atlanta, Georgia, 2013, pp. 758–764, URL: https://aclanthology.org/N13-1092.

[63] M. Graça, Y. Kim, J. Schamper, S. Khadivi, H. Ney, Generalizing back-translation in neural machine translation, in: Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers), Association for Computational Linguistics, Florence, Italy, 2019, pp. 45–52, http://dx.doi.org/10.18653/v1/W19-5205.

[64] R. Sennrich, B. Haddow, A. Birch, Improving neural machine translation models with monolingual data, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Berlin, Germany, 2016, pp. 86–96, http://dx.doi.org/10.18653/v1/P16-1009.

[65] A.W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, Q.V. Le, QANet: Combining local convolution with global self-attention for reading comprehension, 2018, CoRR aba/1804.09541. URL: https://arxiv.org/pdf/1804.09541.

[66] V.C.D. Hoang, P. Koehn, G. Haffari, T. Cohn, Iterative back-translation for neural machine translation, in: Proceedings of the 2nd Workshop on Neural Machine Translation and Generation, Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 18–24, http://dx.doi.org/10.18653/v1/W18-2703.

[67] I. Caswell, C. Chelba, D. Grangier, Tagged back-translation, in: Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers), Association for Computational Linguistics, Florence, Italy, 2019, pp. 53–63, http://dx.doi.org/10.18653/v1/W19-5206.

[68] K. Imamura, A. Fujita, E. Sumita, Enhancement of encoder and attention using target monolingual corpora in neural machine translation, in: Proceedings of the 2nd Workshop on Neural Machine Translation and Generation, Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 55–63, http://dx.doi.org/10.18653/v1/W18-2707.

[69] Q. Xie, Z. Dai, E.H. Hovy, T. Luong, Q. Le, Unsupervised data augmentation for consistency training, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12,

2020, Virtual, 2020, URL: https://proceedings.neurips.cc/paper/2020/hash/44feb0096faa8326192570788b38c1d1-Abstract.html.

[70] X. Dai, H. Adel, An analysis of simple data augmentation for named entity recognition, in: Proceedings of the 28th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online), 2020, pp. 3861–3867, URL: https://www.aclweb.org/anthology/2020.coling-main.343.

[71] A. Anaby-Tavor, B. Carmeli, E. Goldbraich, A. Kantor, G. Kour, S. Shlomov, N. Tepper, N. Zwerdling, Do not have enough data? Deep learning to the rescue!, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, Association for the Advancement of Artificial Intelligence (AAAI), 2020, pp. 7383–7390, http://dx.doi.org/10.1609/aaai.v34i05.6233.

[72] R. Liu, G. Xu, C. Jia, W. Ma, L. Wang, S. Vosoughi, Data boost: Text data augmentation through reinforcement learning guided conditional generation, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, 2020, http://dx.doi.org/10.18653/v1/2020.emnlp-main.726.

[73] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017, CoRR abs/1707.06347. URL: http://arxiv.org/abs/1707.06347. arXiv:1707.06347.

[74] P. Damodaran, Parrot: Paraphrase generation for NLU, 2021.

[75] T. Dopierre, C. Gravier, W. Logerais, ProtAugment: Intent detection meta-learning through unsupervised diverse paraphrasing, in: C. Zong, F. Xia, W. Li, R. Navigli (Eds.), Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021, Association for Computational Linguistics, 2021, pp. 2454–2466, http://dx.doi.org/10.18653/v1/2021.acl-long.191.

[76] T. Niu, M. Bansal, Automatically learning data augmentation policies for dialogue tasks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 1317–1323, http://dx.doi.org/10.18653/v1/D19-1132, URL: https://www.aclweb.org/anthology/D19-1132.

[77] V. Marivate, T. Sefara, Improving short text classification through global augmentation methods, 2019, CoRR abs/1907.03752. URL: http://arxiv.org/abs/1907.03752. arXiv:1907.03752.

[78] Y. Qu, D. Shen, Y. Shen, S. Sajeev, J. Han, W. Chen, CoDA: Contrast-enhanced and diversity-promoting data augmentation for natural language understanding, 2020, CoRR abs/2010.08670. URL: https://arxiv.org/abs/2010.08670. arXiv:2010.08670.

[79] O. Kashefi, R. Hwa, Quantifying the evaluation of heuristic methods for textual data augmentation, in: Proceedings of the Sixth Workshop on Noisy User-Generated Text (W-NUT 2020), Association for Computational Linguistics, 2020, pp. 200–208, http://dx.doi.org/10.18653/v1/2020.wnut-1.26, Online. URL: https://aclanthology.org/2020.wnut-1.26.

[80] S.Y. Feng, V. Gangal, D. Kang, T. Mitamura, E.H. Hovy, GenAug: Data augmentation for finetuning text generators, 2020, CoRR aba/2010.01794. URL: https://arxiv.org/abs/2010.01794. arXiv:2010.01794.

[81] N. Ng, K. Cho, M. Ghassemi, SSMBA: Self-supervised manifold based data augmentation for improving out-of-domain robustness, in: Proc. of EMNLP, 2020, URL: https://arxiv.org/abs/2009.10195.

[82] E. Pavlick, P. Rastogi, J. Ganitkevitch, B. Van Durme, C. Callison-Burch, PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Association for Computational Linguistics, Beijing, China, 2015, pp. 425–430, http://dx.doi.org/10.3115/v1/P15-2070, URL: https://aclanthology.org/P15-2070.

[83] J. Ganitkevitch, C. Callison-Burch, The multilingual paraphrase database, in: The 9th Edition of the Language Resources and Evaluation Conference, European Language Resources Association, Reykjavik, Iceland, 2014, URL: http://cis.upenn.edu/~ccb/publications/ppdb-multilingual.pdf.

[84] E. Ma, NLP augmentation, 2019, https://github.com/makcedward/nlpaug.

[85] C. Van Hee, E. Lefever, V. Hoste, SemEval-2018 task 3: Irony detection in english tweets, in: Proceedings of the 12th International Workshop on Semantic Evaluation, Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 39–50, http://dx.doi.org/10.18653/v1/S18-1005, URL: https://aclanthology.org/S18-1005.

[86] X. Li, D. Roth, Learning question classifiers, in: COLING 2002: The 19th International Conference on Computational Linguistics, 2002, URL: https://www.aclweb.org/anthology/C02-1150.

[87] E. Saravia, H.-C.T. Liu, Y.-H. Huang, J. Wu, Y.-S. Chen, CARER: Contextualized affect representations for emotion recognition, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 3687–3697, http://dx.doi.org/10.18653/v1/D18-1404, URL: https://www.aclweb.org/anthology/D18-1404.

[88] J.E. Hu, R. Rudinger, M. Post, B.V. Durme, PARABANK: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation, 33 (2019) 6521–6528. http://dx.doi.org/10.1609/aaai.v33i01.33016521.

[89] N. Reimers, I. Gurevych, Making monolingual sentence embeddings multilingual using knowledge distillation, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2020, URL: https://arxiv.org/abs/2004.09813.

[90] D. Hendrycks, N. Mu, E.D. Cubuk, B. Zoph, J. Gilmer, B. Lakshminarayanan, AugMix: A simple data processing method to improve robustness and uncertainty, in: Proceedings of the International Conference on Learning Representations (ICLR), 2020.

[91] X. Wang, H. Pham, Z. Dai, G. Neubig, SwitchOut: an efficient data augmentation algorithm for neural machine translation, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 856–861, http://dx.doi.org/10.18653/v1/D18-1100, URL: https://www.aclweb.org/anthology/D18-1100.

[92] R. Jha, C. Lovering, E. Pavlick, Does data augmentation improve generalization in NLP?, 2020, arXiv:2004.15012.