Substructure Substitution: Structured Data Augmentation for NLP

Haoyue Shi **Karen Livescu** **Kevin Gimpel**
Toyota Technological Institute at Chicago, IL, USA, 60637
{freda,klivescu,kgimpel}@ttic.edu

Abstract

We study a family of data augmentation methods, substructure substitution (SUB^2), that generalizes prior methods. SUB^2 generates new examples by substituting substructures (e.g., subtrees or subsequences) with others having the same label. This idea can be applied to many structured NLP tasks such as part-of-speech tagging and parsing. For more general tasks (e.g., text classification) which do not have explicitly annotated substructures, we present variations of SUB^2 based on text spans or parse trees, introducing structure-aware data augmentation methods to general NLP tasks. For most cases, training with a dataset augmented by SUB^2 achieves better performance than training with the original training set. Further experiments show that SUB^2 has more consistent performance than other investigated augmentation methods, across different tasks and sizes of the seed dataset.[1]

1 Introduction

Data augmentation has been found effective for various natural language processing (NLP) tasks, such as machine translation (Fadaee et al., 2017; Gao et al., 2019; Xia et al., 2019, *inter alia*), text classification (Wei and Zou, 2019; Quteineh et al., 2020), syntactic and semantic parsing (Jia and Liang, 2016; Shi et al., 2020; Dehouck and Gómez-Rodríguez, 2020), semantic role labeling (Fürstenau and Lapata, 2009), and dialogue understanding (Hou et al., 2018; Niu and Bansal, 2019). Such methods enhance the diversity of the training set by generating examples based on existing ones, and can make the learned models more robust against noise (Xie et al., 2020). Most existing work focuses on word-level manipulation (Kobayashi, 2018; Wei and Zou, 2019; Dai and Adel, 2020, *inter alia*) or global sequence-to-sequence style generation (Sennrich et al., 2016).

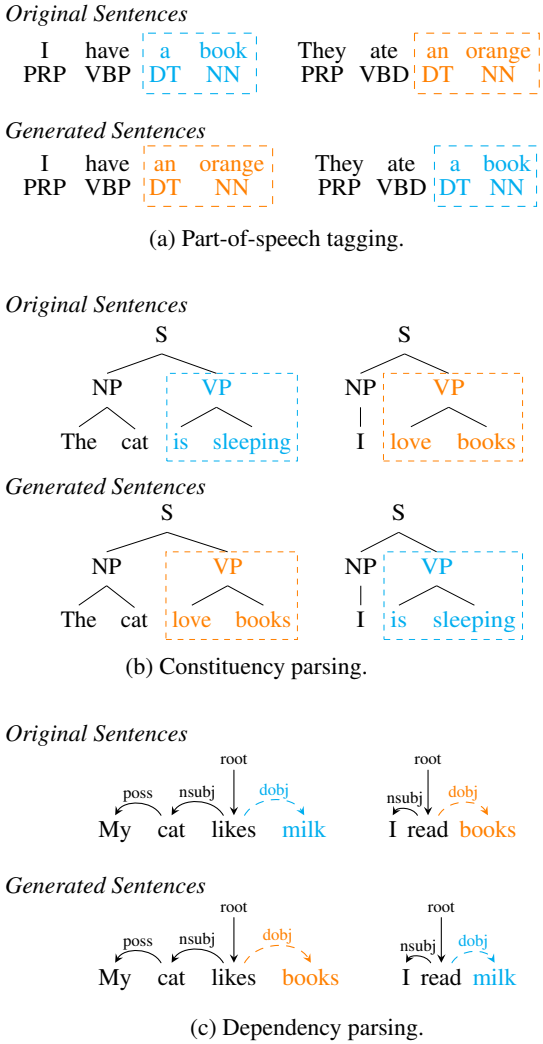
In this work, we study a family of general data augmentation methods, substructure substitution (SUB^2), which generates new examples by substituting same-label substructures (Figure 1). While some instances within this family have been proposed before for certain tasks, we generalize the idea and investigate it for a variety of tasks and settings. SUB^2 naturally fits structured prediction tasks such as part-of-speech tagging and parsing, where substructures exist in the annotations of the tasks. For more general NLP tasks such as text classification, we present variations of SUB^2 which (1) define substructures based on text spans or parse trees for existing examples, and (2) generate new examples by substructure substitution based on the substructures and various kinds of constraints.

While data augmentation methods can often be task-specific or have inconsistent performance, extensive experiments show that SUB^2 consistently helps models achieve competitive or better performance than training on the original dataset across structured prediction tasks and original dataset sizes. We further study the effect of different constraints for the variations of SUB^2 in text classification. While there is no consistently winning combination of constraints, SUB^2 remains dominant on both investigated few-shot text classification datasets.

In addition, when combined with XLM-R (Conneau et al., 2019), a cross-lingual pretrained language model, SUB^2 establishes new state-of-the-art results for sentiment analysis and low-resource part-of-speech tagging. Finally, the experimental setups we define can serve as a benchmark for future work on NLP with little annotated data.

[1]Project page: https://home.ttic.edu/~freda/project/sub2

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 3494–3508



(a) Part-of-speech tagging.

(b) Constituency parsing.

(c) Dependency parsing.

(d) Text classification. To apply SUB^2, we use text spans as substructures, with both the *number of words* in the span and the *text classification label* as constraints (see Sec. 3.2).

Figure 1: Illustration of SUB^2 for investigated tasks. We generate new examples by same-label substructure substitution, whether or not the generated examples are semantically or syntactically acceptable. Best viewed in color.

2 Related Work

Data augmentation aims to generate new examples based on available ones, without actually collecting new data. Such methods reduce the cost of dataset collection, and usually boost model performance on desired tasks. Most existing data augmentation methods for NLP tasks can be classified into the following categories:

Token-level manipulation. Token-level manipulation methods have been widely studied in recent years. They typically create new examples by substituting (word) tokens with ones having the same desired features, such as synonym substitution (Zhang et al., 2015; Wang and Yang, 2015; Fadaee et al., 2017; Kobayashi, 2018) or substitution with words having the same morphological features (Silfverberg et al., 2017). Such methods have been applied to generate adversarial or negative examples which help improve the robustness of neural network–based NLP models (Belinkov and Bisk, 2018; Shi et al., 2018a; Alzantot et al., 2018; Zhang et al., 2019; Min et al., 2020, *inter alia*), or to generate counterfactual examples which help mitigate bias in natural language (Zmigrod et al., 2019; Lu et al., 2020).

Other token-level manipulation methods introduce noise, such as random token shuffling and deletion (Wang et al., 2018; Wei and Zou, 2019; Dai and Adel, 2020). Models trained on the augmented datasets are expected to be more robust against the considered noise.

Constrained text generation. Recent work has explored generating new examples by training a conditional text generation model (Bergmanis et al., 2017; Liu et al., 2020a; Ding et al., 2020; Liu et al., 2020b, *inter alia*), or applying post-processing on the examples generated by pretrained models (Yang et al., 2020; Wan et al., 2020; Yoo et al., 2020). In the data augmentation stage, given task-specific constraints, such models generate associated text accordingly. The generated examples, together with the original datasets, are used to further train models for the primary tasks. A representative method is back-translation (Sennrich et al., 2016), which is effective for not only machine translation, but also style transfer (Prabhumoye et al., 2018; Zhang et al., 2020a), conditional text generation (Sobrevilla Cabezudo et al., 2019), text classification (Iyyer et al., 2018), and grammatical error correction (Xie et al., 2018). Relatedly, automatic question generation has been used in data augmentation for question answering (Yang et al., 2017; Song et al., 2018).

Another approach to example generation is to generate new examples based on predefined templates (Kafle et al., 2017; Asai and Hajishirzi, 2020), where the templates are designed following heuristic, and usually task-specific, rules.

Soft data augmentation. As an alternative to explicit generation of concrete examples, soft augmentation directly represents generated examples in a continuous vector space: Gao et al. (2019) propose to perform soft word substitution for machine translation; recent work has adapted the mixup method (Zhang et al., 2018), which augments the original dataset by linearly interpolating the vector representations of text and labels, to text classification (Guo et al., 2019; Sun et al., 2020), named entity recognition (Chen et al., 2020), and compositional generalization (Guo et al., 2020).

Structure-aware data augmentation. Existing work has also sought potential gain from structures associated with natural language: Xu et al. (2016) improve word relation classification by dependency path–based augmentation. Şahin and Steedman (2018) show that subtree cropping and rotation based on dependency parse trees can help part-of-speech tagging for low-resource languages, while Vania et al. (2019) demonstrate that such methods also help dependency parsing when very limited training data is available.

SUB^2 also falls into this category. The idea of same-label substructure substitution has been used to improve performance on structured prediction tasks such as semantic parsing (Jia and Liang, 2016), constituency parsing (Shi et al., 2020), dependency parsing (Dehouck and Gómez-Rodríguez, 2020), named entity recognition (Dai and Adel, 2020), meaning representation–based text generation (Kedzie and McKeown, 2020), and compositional generalization (Andreas, 2020). To the best of our knowledge, however, SUB^2 has not been systematically studied as a general data augmentation method for NLP tasks. In this work, we not only extend SUB^2 to part-of-speech tagging and structured sentiment classification, but also present a variation that allows a broader range of NLP tasks (e.g., text classification) to benefit from syntactic parse trees. We evaluate SUB^2 and several representative general data augmentation methods, which can be widely applied to various NLP tasks.

When constituency parse trees are used, there is a connection between SUB^2 and tree substitution grammars (TSGs; Schabes, 1990), where the approach can be viewed as (1) estimating a TSG using the given corpus and (2) drawing new sentences from the estimated TSG.

3 Method

We introduce the general framework we investigate in Section 3.1, and describe the variations of SUB^2 which can be applied to text classification and other NLP applications in Section 3.2.

3.1 Substructure Substitution (SUB^2)

As shown in Figure 1, given the original training set $\mathcal{D}$, SUB^2 generates new examples using same-label substructure substitution, and repeats the process until the training set reaches the desired size. The general SUB^2 procedure is presented in Algorithm 1.

Algorithm 1: SUB^2.

Input: Original dataset $\mathcal{D}$,
desired dataset size $N > |\mathcal{D}|$
Output: Augmented dataset $\mathcal{D}'$
$\mathcal{D}' \leftarrow \mathcal{D}$;
repeat
  Uniformly draw $s \in \textit{substructure}(\mathcal{D}')$
  $S \leftarrow \textit{example}(s)$
  Uniformly draw $u \in \{v \mid v \in \textit{substructure}(\mathcal{D}), \textit{label}(v) = \textit{label}(s), v \neq s\}$
  $S' \leftarrow$ replace s with v in S
  $\mathcal{D}' \leftarrow \mathcal{D}' \cup \{S'\}$
until $|\mathcal{D}'| = N$;

For part-of-speech (POS) tagging, we let text spans be substructures and use the corresponding POS tag sequences as substructure labels (Figure 1a); for constituency parsing, we use subtrees as the substructures, with constituent labels as the substructure labels (Figure 1b); for dependency parsing, we also use subtrees as substructures, and let the dependency arc labels, which link the heads of subtrees to their parents, be the substructure labels (Figure 1c).

3.2 Variations of SUB^2 for Text Classification

Text classification examples do not typically contain explicit substructures. However, we can obtain them by viewing all text spans as substructures (Figure 1d). This approach may be too unconstrained in

practice, so we consider constraining substitution based on matching several features of the spans:

- **Text classification label**: when considering this constraint, we can only substitute a span with another span that comes from text annotated with the same class label as the original one; otherwise we can choose the alternative from any example text in the training corpus.
- **Constituency**: when considering this constraint, we can only substitute a constituent with another constituent (according to a constituency parse of the text, whether they have the same constituent label or not); otherwise the considered spans do not necessarily need to be constituents.
- **Annotated text span label**: in our experiments, this constraint is valid only when the previous constraint (constituency) is considered. When considering this constraint, we can only perform substitution between text spans with the same annotated label (e.g., constituent label).[2]
- **Number of words**: when considering this constraint, we can only substitute a span with another having the same number of words; otherwise we can substitute a span with any other span.

Shi et al. (2018b) argue that binary balanced trees are better backbones for recursive neural networks (Zhu et al., 2015; Tai et al., 2015) on text classification; inspired by them, we introduce the following constraint in this work:

- **"Constituency" in binary balanced tree.** we use binary balanced trees, analogously to constituency parse trees, as the backbone for SUB^2: we (1) generate balanced trees by recursively splitting a span of n words into two consecutive groups, which consist of $\lfloor \frac{n}{2} \rfloor$ and $\lceil \frac{n}{2} \rceil$ words respectively, and (2) treat each nonterminal in the balanced tree as a substructure to perform SUB^2.

We also investigate combinations of the above constraints, where we require all the chosen constraints to be the same to perform SUB^2. For example, combining *text classification label* and *number of words* (Figure 1d) requires the original and the alternative span to have the same text label and the same number of words.

[2]There can be other text span labels such as sentiment labels of constituents (Socher et al., 2013).

4 Experiments

We introduce our experimental setups (Section 4.1), and evaluate SUB^2 and several data augmentation baselines (Section 4.2) on four tasks: part-of-speech tagging (Section 4.3), dependency parsing (Section 4.4), constituency parsing (Section 4.5), and text classification (Section 4.6).

4.1 Setup

For part-of-speech tagging and text classification, we add a two-layer perceptron on top of XLM-R (Conneau et al., 2019) embeddings, where we calculate contextualized token embeddings by a learnable weighted average across layers. We use endpoint concatenation (i.e., the concatenation of the first and last token representation) to obtain fixed-dimensional span or sentence features, and keep the pretrained model frozen during training.[3] For dependency parsing, we use the SuPar implementation of Dozat and Manning (2017).[4] For constituency parsing, we use Benepar (Kitaev and Klein, 2018).[5]

For all data augmentation methods, including the baselines (Section 4.2), we only augment the training set, and use the original development set. If not specified, we introduce 20 times more examples than the original training set when applying an augmentation method. When introducing $k\times$ new examples, we also replicate the original training set k times to ensure that the model can access sufficient examples from the original distribution.

All models are initialized with the XLM-R base model (Conneau et al., 2019) if not specified. We train models for 20 epochs in high-resource settings (i.e., high-resource part-of-speech tagging, sentiment classification trained on the full training set) or when applying data augmentation methods, and for 400 epochs in the low-resource settings without augmentation; we select the one with the highest accuracy or F_1 score on the development set. All models are optimized using Adam (Kingma and Ba, 2015), with learning rates cho-

[3]We did not observe significant improvement by finetuning the large pretrained language model, and for most cases, the performance is much worse than the current scheme we apply.

[4]https://github.com/yzhangcs/parser

[5]https://github.com/nikitakit/self-attentive-parser

sen from $\{5 \times 10^{-4}, 5 \times 10^{-5}\}$. For the hidden layer size (i.e., the hidden size of the perceptron for part-of-speech tagging and text classification, the dimensionality of span representation and scoring multi-layer perceptron for constituency parsing, and the dimensionality of token representation and scoring multi-layer perceptron for dependency parsing), we vary it between 128 and 512. We apply a 0.2 dropout ratio to the contextualized embeddings in the training stage. All other hyperparameters are the same as the default settings in the released codebases.

4.2 Baselines

We compare SUB^2 to the following baselines:

- **No augmentation** (NOAUG), where the original training and development set are used.
- **Contextualized substitution** (CTXSUB), where we apply contextualized augmentation (Kobayashi, 2018), masking out a random word token from the existing dataset and using multilingual BERT (mBERT; Devlin et al., 2019) to generate a different word.
- **Knowledge based guided synonym substitution** (SYNO), where we substitute a random word token by its synonym defined in an existing knowledge base.[6]
- **Random shuffle** (SHUF), where we randomly shuffle all the words in the original sentence, while keeping the original structured or non-structured labels. It is worth noting that for dependency parsing, we shuffle the words, while maintaining the dependency arcs between individual word tokens; for constituency parsing, we shuffle the terminal nodes, and insert them back into the tree structure. Our SHUF method for constituency parsing is arguably more noisy than that for dependency parsing.

All of the data augmentation baselines are explicit augmentations where concrete new examples are generated and used. The methods above are generally applicable to a wide range of NLP tasks.

[6]Specifically, we use the lexical PPDB-XL (Ganitkevitch and Callison-Burch, 2014; Pavlick et al., 2015) of the appropriate language when applicable.

Lang. / Aug.	SOTA NOAUG	mBERT NOAUG	XLM-R NOAUG	XLM-R SUB^2
high-resource languages				
avg.	96.9	97.1	**97.7**†	**97.7**†
bg	98.7	98.9	**99.4**	**99.4**
cs	99.0	99.0	**99.2**	**99.2**
da	97.2	97.8	**98.7**	98.5
de	94.4	94.6	**95.3**	95.1
en	96.1	96.5	**97.5**	97.3
es	96.8	96.9	**97.5**	**97.5**
eu	96.1	95.7	96.6	**96.8**
fa	97.5	96.6	**98.6**	98.5
fi	95.8	96.9	**98.3**	**98.3**
fr	96.6	96.7	**96.9**	**96.9**
he	97.4	96.9	**97.9**	97.8
hi	97.4	96.9	**97.9**	97.8
hr	96.8	97.6	97.9	**98.0**
id	94.0	93.7	**93.8**	93.7
it	98.1	98.6	**98.7**	**98.7**
nl	93.8	92.9	**94.0**	93.6
no	98.5	98.6	**99.0**	98.9
pl	97.7	98.5	98.8	**98.9**
pt	98.2	98.3	**98.6**	**98.6**
sl	98.1	98.7	**99.2**	**99.2**
sv	97.4	98.2	**98.9**	**98.9**
low-resource languages				
avg.	92.7	94.7	95.4	**96.1**†
el	98.2	98.6	**98.8**	98.7
et	92.8	94.1	95.7	**96.3**
ga	91.1	92.9	94.1	**95.8**
hu	94.0	96.8	**97.7**	97.5
ro	91.5	95.0	94.9	**95.8**
ta	88.7	90.4	91.3	**92.5**

Table 1: Part-of-speech tagging accuracy ($\times$100) on the standard test set of UD 1.2 high-resource (top) and low-resource (bottom) languages, across different pre-trained models and augmentation methods. The best numbers in each row are bolded. SOTA: previous state of the art, i.e., the best test accuracy for each language among all methods reported by Heinzerling and Strube (2019), where all numbers in the SOTA column are not necessarily produced by the same model. † denotes new state-of-the-art results.

4.3 Part-of-Speech Tagging

We conduct our experiments using the Universal Dependencies (UD; Nivre et al., 2016, 2020)[7] dataset.

First, we compare both NOAUG and SUB^2 to the previous state-of-the-art performance (Heinzerling and Strube, 2019) to ensure that our baselines are strong enough (Table 1).[8] Heinzerling and Strube (2019) take the token-wise concatenation of mBERT last-layer representations, byte-pair en-

[7]http://universaldependencies.org/

[8]We use UD v1.2 for direct comparison with existing work.

Language (Treebank)	NoAug	CtxSub	Shuf	Sub^2
avg.	92.4	87.1	86.8	**93.0**
be (hse)	96.2	90.3	92.5	**96.9**
lt (hse)	92.7	90.1	88.4	**93.1**
mr (ufal)	87.9	81.5	84.5	**89.1**
ta (ttb)	91.7	85.4	83.2	**92.3**
te (mtg)	**93.8**	88.2	85.6	93.0

Table 2: Part-of-speech tagging accuracy ($\times 100$) on the standard test set of selected UD 2.6 low-resource treebanks. The best number in each row is bolded.

coding (BPE; Gage, 1994)–based LSTM hidden states and character-LSTM hidden states as the input to the classifier, and fine-tune the pretrained mBERT during training. We find that using our framework with frozen mBERT and extra learnable layer weight parameters, we are able to obtain competitive or better results than those reported by Heinzerling and Strube (2019); the gains grow larger when using XLM-R, which is trained on larger corpora than mBERT. In addition, by augmenting the training set with Sub^2, we achieve better average accuracy on low-resource languages (paired, one-tailed t-test p-value$= 0.028$) while remaining competitive on high-resource languages (no statistically significant difference).

We further test the part-of-speech tagging accuracy on 5 selected low-resource treebanks in the UD 2.6 dataset (Table 2), following the official splits of the dataset. For four of the five treebanks, Sub^2 achieves the best performance among all methods, while also maintaining competitive performance on the Telugu treebank. In contrast, other augmentation methods (CtxSub and Shuf) are harmful compared to NoAug on all treebanks.

4.4 Dependency Parsing

We evaluate the performance of models using the standard Penn Treebank dataset (PTB; Marcus et al., 1993), converted by Stanford dependency converter v3.0,[9] following the standard splits.

We first compare the performance of Sub^2 and baselines in the low-resource setting (Table 3). All methods sometimes, though not always, improve performance over NoAug. Shuf achieves the best LAS when there is only an extremely small training set (e.g., 10 examples) available; however, when the size of the original training set becomes larger, Sub^2 begins to dominate, while CtxSub and Shuf start to sometimes hurt the performance. In addition, a larger augmented dataset does not necessarily lead to better performance, but CtxSub and Shuf often hurt performance as the augmented set size gets too large while Sub^2 does not. Although there is not a consistently best configuration, throughout our experiments, augmenting by $10\times$–$50\times$ the original dataset size produces good improvements over NoAug for both development and test sets. When tuning the augmentation factor k using small development sets, Sub^2 improves over NoAug for four out of five seed dataset sizes. CtxSub, in contrast, improves performance for only one out of five.

When training on the full WSJ training set, Sub^2 does not necessarily help improve over NoAug, but it also does not hurt performance (Table 4).[10]

$\lvert\mathcal{D}'\rvert = k \times \lvert\mathcal{D}\rvert$	$\lvert\mathcal{D}\rvert$ 10	50	100	500	1,000
$2\times$ (CtxSub)	38.3	55.1	62.9	78.1	80.1
$5\times$ (CtxSub)	35.5	55.9	62.1	81.4	81.0
$10\times$ (CtxSub)	**39.8**	55.1	61.7	81.7	80.8
$50\times$ (CtxSub)	31.2	52.3	60.9	79.3	78.0
$100\times$ (CtxSub)	32.0	53.1	58.2	77.1	75.9
$2\times$ (Shuf)	32.8	55.9	62.5	76.7	78.4
$5\times$ (Shuf)	34.4	52.7	60.5	77.5	81.6
$10\times$ (Shuf)	**39.8**	53.1	**63.7**	77.9	81.9
$50\times$ (Shuf)	34.0	52.7	60.9	79.1	79.6
$100\times$ (Shuf)	39.1	55.9	61.3	80.4	77.4
$2\times$ (Sub^2)	38.3	54.3	61.7	81.0	80.0
$5\times$ (Sub^2)	35.9	54.7	62.9	**82.5**	80.4
$10\times$ (Sub^2)	32.0	53.9	**63.7**	81.7	80.6
$50\times$ (Sub^2)	33.2	**57.0**	62.5	81.4	**82.5**
$100\times$ (Sub^2)	38.3	52.7	62.5	78.8	82.1

Table 3: Labeled attachment scores (LAS) on the standard PTB development set (PTB Section 22). We start with an original training set $\mathcal{D}$, which consists of $\lvert\mathcal{D}\rvert \in \{10, 50, 100, 500, 1000\}$ examples, and augment it $k \in \{2, 5, 10, 50, 100\}$ times. For each training set $\mathcal{D}$, the corresponding development set consists of $\max\left(10, \frac{\lvert\mathcal{D}\rvert}{10}\right)$ examples. Underlined results correspond to k values tuned to maximize development set LAS for each combination of augmentation method and $\lvert\mathcal{D}\rvert$ (if there are multiple k values with the same development LAS, we choose the smallest). The best number in each column is bolded.

[9] https://nlp.stanford.edu/software/stanford-dependencies.shtml

[10] An additional finding here is that a simple biaffine dependency parsing model (Dozat and Manning, 2017) with XLM-R initialization is able to set a new state of the art for dependency parsing with only in-domain annotation.

Model	UAS	LAS
Mrini et al. (2020)† (NOAUG)	97.4	96.3
Zhang et al. (2020b)‡ (NOAUG)	96.1	94.5
BiAffine+XLM-R (NOAUG)	96.7	95.2
BiAffine+XLM-R+SUB^2	96.6	95.2

Table 4: Unlabeled attachment score (UAS) and labeled attachment score (LAS) on the PTB dependency test set. Models are trained with the full PTB training set. †: the previously best result using any kind of annotation (e.g., constituency parse trees); ‡: the previously best result using only dependency annotations. BiAffine: the bi-affine dependency parsing model proposed by Dozat and Manning (2017).

4.5 Constituency Parsing

Shi et al. (2020) have shown that SUB^2 can significantly improve few-shot constituency parsing on the Penn Treebank dataset; in this work, we extend the few-shot parsing evaluation to other domains, using the Foreebank (FBANK; Kaljahi et al., 2015) and NXT-Switchboard (SWBD; Calhoun et al., 2010) datasets. Foreebank consists of 1,000 English and 1,000 French sentences; for either language, we randomly select 50 sentences for training, 50 for development, and 250 for testing.[11] We follow the standard splits of NXT-Switchboard, and randomly select 50 sentences from the training set and 50 from the development set for training and development respectively.

We compare data augmentation methods using the setup of few-shot parsing from scratch (Table 5). Among all settings we tested, SUB^2 achieves the best performance, while all augmentation methods we investigated improve over training only on the original dataset (NOAUG). Surprisingly, we find that the seemingly meaningless SHUF, which randomly shuffles the sentence and inserts the shuffled words back into the original parse tree structure as the nonterminals, also consistently helps few-shot parsing by a nontrivial margin.[12]

For domain adaptation (Table 6), we first train Benepar (Kitaev and Klein, 2018) on the Penn Treebank dataset, achieving an F_1 score of 95.1 on the PTB standard development set, and use the pretrained model as the initialization. While compared to few-shot parsing trained from scratch, the gain by data augmentation generally becomes smaller, SUB^2 still works the best across datasets.

Method	FBANK(en)	FBANK(fr)	SWBD
NOAUG	33.1	27.3	29.1
CTXSUB	64.8	59.9	51.1
SYNO	62.9	60.8	52.1
SHUF	55.9	48.8	37.0
SUB^2	**71.8**	**70.8**	**64.6**

Table 5: Labeled F_1 scores ($\times 100$) on the test set of each constituency treebank, in the setting of few-shot parsing. The best number in each column is bolded.

Method	FBANK(en)	FBANK(fr)	SWBD
PTB	82.3	30.8	74.3
NOAUG$\rightarrow$	83.1	70.1	77.2
CTXSUB $\rightarrow$	84.0	71.1	78.2
SYNO $\rightarrow$	84.2	71.0	77.9
SHUF $\rightarrow$	83.5	70.1	75.6
SUB^2 $\rightarrow$	**84.6**	**72.6**	**78.3**

Table 6: Labeled F_1 scores ($\times 100$) on the test set of each constituency treebank, in the setting of domain adaptation. PTB: directly testing the model trained on the Penn Treebank; $\rightarrow$: transferring a model trained on PTB to each domain. The best number in each column is bolded.

4.6 Text Classification

We take text classification as a representative of a wider range of NLP tasks, and evaluate the methods introduced in Section 3.2 and baselines on low-resource versions of two text classification datasets: SST (Socher et al., 2013) and a sentence version of the AG News dataset (Zhang et al., 2015).[13] To avoid over-fitting to the small development set and tuning on test set issues, we introduce small "development test" (devtest) sets for each task, and only evaluate on the test sets using SUB^2 variations with the best devtest performance. For settings requiring constituency parse trees, we generate them using Benepar (Kitaev and Klein, 2018) trained on the standard PTB dataset.

Across the two datasets, any data augmentation technique usually improves over NOAUG (Table 7).[14] While methods in the SUB^2 family usually lead to the best performance on both tasks, there is not a consistently best-performing combination of constraints. Surprisingly, SUB^2 without constraints

[11] We leave the other 650 sentences for future use.

[12] This trend may be explained by benefits in learning/optimization stability in this few-shot setting, but we leave a richer exploration of potential explanations for future work.

[13] We only keep the single-sentence instances among all examples in each split of the original AG News dataset, following Shi et al. (2018b).

[14] To measure variance due to random selection of data from the full sets, results on additional randomly-sampled few-shot datasets for both tasks can be found in Appendix A.

Dataset	Method	Constraints				Accuracy		
		Text Label	*Constit.*	*Span Label*	*#Words*	*Dev*	*Devtest*	*Test*
AG News-1%	NOAUG	N/A	N/A	N/A	N/A	55.2	44.8	44.8
$\lvert\mathcal{D}_{train}\rvert = 0.6$K	CTXSUB	N/A	N/A	N/A	N/A	91.0	89.6	86.0
$\lvert\mathcal{D}_{dev}\rvert = 0.06$K	SYNO	N/A	N/A	N/A	N/A	89.6	89.6	85.6
$\lvert\mathcal{D}_{devtest}\rvert = 0.06$K	SHUF	N/A	N/A	N/A	N/A	91.0	88.1	86.3
	SUB^2 (balanced tree)	✗	N/A	N/A	✓	88.1	**92.5**	86.7
	SUB^2	✓	✓	✗	✗	91.0	**92.5**	**87.0**
SST-10%	NOAUG	N/A	N/A	N/A	N/A	27.3	35.5	23.3
$\lvert\mathcal{D}_{train}\rvert = 0.8$K	CTXSUB	N/A	N/A	N/A	N/A	40.0	53.6	44.9
$\lvert\mathcal{D}_{dev}\rvert = 0.1$K	SYNO	N/A	N/A	N/A	N/A	40.0	39.1	39.0
$\lvert\mathcal{D}_{devtest}\rvert = 0.1$K	SHUF	N/A	N/A	N/A	N/A	37.3	44.5	38.9
	SUB^2 (balanced tree)	✗	N/A	N/A	✗	40.0	50.0	44.6
	SUB^2	✓	✓	sentiment	✗	40.0	**55.5**	**45.8**

Table 7: Accuracy (×100) on the low-resource sentence AG News and SST datasets, together with the corresponding constraints, where *Constit.* denotes constituency labels. We enumerate multiple constraints for data augmentation with SUB^2, and only test the obtained model with the highest devtest accuracy – results for all investigated combinations of constraints can be found in Appendix A. The best devtest accuracies and the best test accuracy for each dataset are bolded.

Method	Dev. Acc.	Test Acc.
XLM-R (NOAUG)	56.1	55.7
XLM-R (SUB^2)	**56.6**	**56.6**
Brahma (2018)	N/A	56.2

Table 8: Accuracy (×100) on the SST standard development and test set.

on the text label, which may introduce more noise than having the constraint, does not necessarily hurt the performance much.

While constituency parse tree–based SUB^2 typically achieves competitive performance among all investigated combinations of constraints, the gain over SUB^2 with balanced trees is not consistent. Our results are in line with Shi et al. (2018b).

We further use SUB^2 with constraints of (1) text label, (2) phrase, and (3) phrase sentiment label, to augment the full SST training set, since it is the best augmentation method for few-shot sentiment classification, in terms of devtest accuracy (Table 7). In addition to sentences, we also add phrases (i.e., subtrees) as training examples, following most existing work (Socher et al., 2013; Kim, 2014; Brahma, 2018, *inter alia*),[15] to boost performance. In this setting, we find that SUB^2 helps set a new state of the art on the SST dataset (Table 8).

[15]That is, unlike in Table 7, we apply the same settings as most existing work to produce numbers in Table 8.

5 Conclusion and Future Work

We investigate substructure substitution (SUB^2), a family of data augmentation methods that generates new examples by same-label substructure substitution. Such methods help achieve competitive or better performance on the tasks of part-of-speech tagging, dependency parsing, constituency parsing, and text classification in the few-shot setting, where the number of annotated examples is limited. While other data augmentation methods (e.g., CTXSUB and SHUF) sometimes improve the performance, SUB^2 is the only one that consistently improves performance for low-resource NLP across tasks and seed dataset sizes. The experimental setups used in this work can further serve as a standard benchmark for future work on NLP with limited annotations.

There are two open questions remaining to be addressed. First, it is still unclear why SHUF, which requires the model to recover the correct constituency parse tree of a sentence while only accessing shuffled words, consistently helps improve few-shot constituency parsing by a nontrivial margin. Second, while constituency parse tree–based SUB^2 sometimes achieves better performance than SUB^2 without the constituency constraint, the advantage is not large: whether explicit constituency parse trees are useful for NLP applications in the neural network era remains an open question. We leave the above questions, as well as applications of SUB^2 to more NLP tasks, for future work.

6 Acknowledgment

We thank Jiayuan Mao, Shane Settle and Bowen Shi for helpful suggestions on this work, Yu Zhang for fruitful discussion regarding dependency parsing, as well as anonymous reviewers for their valuable feedback.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566, Online. Association for Computational Linguistics.

Akari Asai and Hannaneh Hajishirzi. 2020. Logic-guided data augmentation and regularization for consistent question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5642–5650, Online. Association for Computational Linguistics.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *International Conference on Learning Representations*.

Toms Bergmanis, Katharina Kann, Hinrich Schütze, and Sharon Goldwater. 2017. Training data augmentation for low-resource morphological inflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 31–39, Vancouver. Association for Computational Linguistics.

Siddhartha Brahma. 2018. Improved sentence modeling using suffix bidirectional LSTM. *arXiv preprint arXiv:1805.07340*.

Sasha Calhoun, Jean Carletta, Jason M Brenier, Neil Mayo, Dan Jurafsky, Mark Steedman, and David Beaver. 2010. The nxt-format switchboard corpus: a rich resource for investigating the syntax, semantics, pragmatics and prosody of dialogue. *Language resources and evaluation*, 44(4):387–419.

Jiaao Chen, Zhenghui Wang, Ran Tian, Zichao Yang, and Diyi Yang. 2020. Local additivity based data augmentation for semi-supervised NER. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1241–1251, Online. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3861–3867, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Mathieu Dehouck and Carlos Gómez-Rodríguez. 2020. Data augmentation via subtree swapping for dependency parsing of low-resource languages. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3818–3830, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. DAGA: Data augmentation with a generation approach for low-resource tagging tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6045–6057, Online. Association for Computational Linguistics.

Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *International Conference on Learning Representations*.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. 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)*, pages 567–573, Vancouver, Canada. Association for Computational Linguistics.

Hagen Fürstenau and Mirella Lapata. 2009. Semi-supervised semantic role labeling. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 220–228, Athens, Greece. Association for Computational Linguistics.

Philip Gage. 1994. A new algorithm for data compression. *C Users Journal*, 12(2):23–38.

Juri Ganitkevitch and Chris Callison-Burch. 2014. The multilingual paraphrase database. In *The 9th edition*

of the Language Resources and Evaluation Conference, Reykjavik, Iceland. European Language Resources Association.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544, Florence, Italy. Association for Computational Linguistics.

Demi Guo, Yoon Kim, and Alexander Rush. 2020. Sequence-level mixed sample data augmentation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5547–5552, Online. Association for Computational Linguistics.

Hongyu Guo, Yongyi Mao, and Richong Zhang. 2019. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*.

Benjamin Heinzerling and Michael Strube. 2019. Sequence tagging with contextual and non-contextual subword representations: A multilingual evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 273–291, Florence, Italy. Association for Computational Linguistics.

Yutai Hou, Yijia Liu, Wanxiang Che, and Ting Liu. 2018. Sequence-to-sequence data augmentation for dialogue language understanding. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1234–1245, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12–22, Berlin, Germany. Association for Computational Linguistics.

Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan. 2017. Data augmentation for visual question answering. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 198–202, Santiago de Compostela, Spain. Association for Computational Linguistics.

Rasoul Kaljahi, Jennifer Foster, Johann Roturier, Corentin Ribeyre, Teresa Lynn, and Joseph Le Roux. 2015. Foreebank: Syntactic analysis of customer support forums. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1341–1347, Lisbon, Portugal. Association for Computational Linguistics.

Chris Kedzie and Kathleen McKeown. 2020. Controllable meaning representation to text generation: Linearization and data augmentation strategies. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5160–5185, Online. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. 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)*, pages 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

Dayiheng Liu, Yeyun Gong, Jie Fu, Yu Yan, Jiusheng Chen, Jiancheng Lv, Nan Duan, and Ming Zhou. 2020a. Tell me how to ask again: Question data augmentation with controllable rewriting in continuous space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5798–5810, Online. Association for Computational Linguistics.

Ruibo Liu, Guangxuan Xu, Chenyan Jia, Weicheng Ma, Lili Wang, and Soroush Vosoughi. 2020b. 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)*, pages 9031–9041, Online. Association for Computational Linguistics.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2020. Gender bias in neural natural language processing. In *Logic, Language, and Security*, pages 189–202. Springer.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2339–2352, Online. Association for Computational Linguistics.

Khalil Mrini, Franck Dernoncourt, Quan Hung Tran, Trung Bui, Walter Chang, and Ndapa Nakashole. 2020. Rethinking self-attention: Towards interpretability in neural parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 731–742, Online. Association for Computational Linguistics.

Tong Niu and Mohit Bansal. 2019. 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)*, pages 1317–1323, Hong Kong, China. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. 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)*, pages 425–430, Beijing, China. Association for Computational Linguistics.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876, Melbourne, Australia. Association for Computational Linguistics.

Husam Quteineh, Spyridon Samothrakis, and Richard Sutcliffe. 2020. Textual data augmentation for efficient active learning on tiny datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7400–7410, Online. Association for Computational Linguistics.

Gözde Gül Şahin and Mark Steedman. 2018. Data augmentation via dependency tree morphing for low-resource languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5004–5009, Brussels, Belgium. Association for Computational Linguistics.

Yves Schabes. 1990. Mathematical and computational aspects of lexicalized grammars. *PhD. thesis*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Haoyue Shi, Karen Livescu, and Kevin Gimpel. 2020. On the role of supervision in unsupervised constituency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7611–7621, Online. Association for Computational Linguistics.

Haoyue Shi, Jiayuan Mao, Tete Xiao, Yuning Jiang, and Jian Sun. 2018a. Learning visually-grounded semantics from contrastive adversarial samples. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3715–3727, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Haoyue Shi, Hao Zhou, Jiaze Chen, and Lei Li. 2018b. On tree-based neural sentence modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4631–4641, Brussels, Belgium. Association for Computational Linguistics.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99, Vancouver. Association for Computational Linguistics.

Marco Antonio Sobrevilla Cabezudo, Simon Mille, and Thiago Pardo. 2019. Back-translation as strategy to tackle the lack of corpus in natural language generation from semantic representations. In *Proceedings of the 2nd Workshop on Multilingual Surface*

Realisation (MSR 2019), pages 94–103, Hong Kong, China. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana. Association for Computational Linguistics.

Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, Philip Yu, and Lifang He. 2020. Mixup-transformer: Dynamic data augmentation for NLP tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3436–3440, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.

Clara Vania, Yova Kementchedjhieva, Anders Søgaard, and Adam Lopez. 2019. A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. 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)*, pages 1105–1116, Hong Kong, China. Association for Computational Linguistics.

Zhaohong Wan, Xiaojun Wan, and Wenguang Wang. 2020. Improving grammatical error correction with data augmentation by editing latent representation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2202–2212, Barcelona, Spain (Online). International Committee on Computational Linguistics.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563, Lisbon, Portugal. Association for Computational Linguistics.

Xinyi Wang, Hieu Pham, Zihang Dai, and Graham Neubig. 2018. SwitchOut: an efficient data augmentation algorithm for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 856–861, Brussels, Belgium. Association for Computational Linguistics.

Jason Wei and Kai Zou. 2019. 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)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Mengzhou Xia, Xiang Kong, Antonios Anastasopoulos, and Graham Neubig. 2019. Generalized data augmentation for low-resource translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5786–5796, Florence, Italy. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Ng, and Dan Jurafsky. 2018. Noising and denoising natural language: Diverse backtranslation for grammar correction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 619–628, New Orleans, Louisiana. Association for Computational Linguistics.

Yan Xu, Ran Jia, Lili Mou, Ge Li, Yunchuan Chen, Yangyang Lu, and Zhi Jin. 2016. Improved relation classification by deep recurrent neural networks with data augmentation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1461–1470, Osaka, Japan. The COLING 2016 Organizing Committee.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. Generative data augmentation for commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1008–1025, Online. Association for Computational Linguistics.

Zhilin Yang, Junjie Hu, Ruslan Salakhutdinov, and William Cohen. 2017. Semi-supervised QA with generative domain-adaptive nets. In *Proceedings of the 55th Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 1040–1050, Vancouver, Canada. Association for Computational Linguistics.

Kang Min Yoo, Hanbit Lee, Franck Dernoncourt, Trung Bui, Walter Chang, and Sang-goo Lee. 2020. Variational hierarchical dialog autoencoder for dialog state tracking data augmentation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3406–3425, Online. Association for Computational Linguistics.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Huangzhao Zhang, Hao Zhou, Ning Miao, and Lei Li. 2019. Generating fluent adversarial examples for natural languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5569, Florence, Italy. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Yi Zhang, Tao Ge, and Xu Sun. 2020a. Parallel data augmentation for formality style transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3221–3228, Online. Association for Computational Linguistics.

Yu Zhang, Zhenghua Li, and Min Zhang. 2020b. Efficient second-order TreeCRF for neural dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3295–3305, Online. Association for Computational Linguistics.

Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. 2015. Long short-term memory over recursive structures. In *International Conference on Machine Learning*, pages 1604–1612.

Ran Zmigrod, Sabrina J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1651–1661, Florence, Italy. Association for Computational Linguistics.

Appendices

A Text Classification: Another Few-Shot Dataset

For either sentence AG News or SST dataset, we create two few-shot dataset of the same size, but with different examples. We report the model performances in Tables 9 and 10. Results in Table 7 are evaluated on the same few-shot dataset as those in Table 9.

Method	Constraints				Accuracy		
	Text Label	*Phrase*	*Pharse Label*	*#Words*	*Dev*	*Devtest*	*Test*
AG News-1% ($\|\mathcal{D}_{train}\| = 0.6$K, $\|\mathcal{D}_{dev}\| = \|\mathcal{D}_{devtest}\| = 0.06$K)							
NoAug	N/A	N/A	N/A	N/A	55.2	44.8	44.8
CtxSub	N/A	N/A	N/A	N/A	91.0	**89.6**	86.0
Shuf	N/A	N/A	N/A	N/A	91.0	88.1	86.3
Syno	N/A	N/A	N/A	N/A	89.6	**89.6**	85.6
Sub^2 (balanced tree)	✓	N/A	N/A	✓	89.6	89.6	
	✓	N/A	N/A	✗	89.6	91.0	
	✗	N/A	N/A	✓	88.1	**92.5**	86.7
	✗	N/A	N/A	✗	91.0	91.0	
Sub^2	✓	✓	const.	✓	86.6	89.6	
	✓	✓	const.	✗	86.6	91.0	
	✓	✓	✗	✓	86.6	89.6	
	✓	✓	✗	✗	**91.0**	**92.5**	87.0
	✓	✗	N/A	✗	86.6	**92.5**	
	✓	✗	N/A	✓	86.6	91.0	
	✗	✓	const.	✓	86.6	86.6	
	✗	✓	const.	✗	86.6	86.6	
	✗	✓	✗	✓	91.0	88.1	
	✗	✓	✗	✗	89.6	88.1	
SST-10% ($\|\mathcal{D}_{train}\| = 0.8$K, $\|\mathcal{D}_{dev}\| = \|\mathcal{D}_{devtest}\| = 0.1$K)							
NoAug	N/A	N/A	N/A	N/A	27.3	35.5	23.3
CtxSub	N/A	N/A	N/A	N/A	40.0	**53.6**	44.9
Shuf	N/A	N/A	N/A	N/A	37.3	44.5	38.9
Syno	N/A	N/A	N/A	N/A	40.0	39.1	39.0
Sub^2 (balanced tree)	✓	N/A	N/A	✓	36.4	49.1	
	✓	N/A	N/A	✗	26.4	35.5	
	✗	N/A	N/A	✓	38.2	49.1	
	✗	N/A	N/A	✗	40.0	**50.0**	44.6
Sub^2	✓	✓	senti.	✓	39.1	53.6	
	✓	✓	senti.	✗	40.0	**55.5**	45.8
	✓	✓	const.	✓	37.3	52.7	
	✓	✓	const.	✗	40.9	50.9	
	✓	✓	✗	✓	36.4	50.9	
	✓	✓	✗	✗	39.1	47.3	
	✓	✗	N/A	✓	38.2	48.2	
	✓	✗	N/A	✗	40.9	47.3	
	✗	✓	const.	✓	39.1	50.9	
	✗	✓	const.	✗	40.0	54.5	
	✗	✓	✗	✓	38.2	54.5	
	✗	✓	✗	✗	39.1	50.0	

Table 9: Accuracy ($\times 100$) on the AG News and SST dataset (few-shot set 1). The best devtest numbers in each section are bolded.

Method	Constraints				Accuracy		
	Text Label	*Phrase*	*Pharse Label*	*#Words*	*Dev*	*Devtest*	*Test*
AG News-1% ($\|\mathcal{D}_{train}\| = 0.6$K, $\|\mathcal{D}_{dev}\| = \|\mathcal{D}_{devtest}\| = 0.06$K)							
NoAug	✓	N/A	N/A	N/A	40.3	44.8	38.0
CtxSub	N/A	N/A	N/A	N/A	88.1	**89.6**	86.1
Shuf	N/A	N/A	N/A	N/A	86.6	88.1	85.7
Syno	N/A	N/A	N/A	N/A	88.1	85.1	84.2
Sub^2 (balanced tree)	✓	N/A	N/A	✗	88.1	88.1	
	✓	N/A	N/A	✓	86.6	88.1	
	✗	N/A	N/A	✗	89.6	89.6	
	✗	N/A	N/A	✓	89.6	**91.0**	86.5
Sub^2	✓	✓	const.	✓	86.6	**88.1**	
	✓	✓	const.	✗	86.6	**88.1**	
	✓	✓	✗	✓	88.1	**88.1**	
	✓	✓	✗	✗	86.6	**88.1**	
	✓	✗	N/A	✓	86.6	**88.1**	
	✓	✗	N/A	✗	**89.6**	**88.1**	85.9
	✗	✓	const.	✓	86.6	86.6	
	✗	✓	const.	✗	83.6	**88.1**	
	✗	✓	✗	✓	83.6	86.6	
	✗	✓	✗	✗	82.1	**88.1**	
SST-10% ($\|\mathcal{D}_{train}\| = 0.8$K, $\|\mathcal{D}_{dev}\| = \|\mathcal{D}_{devtest}\| = 0.1$K)							
NoAug	✓	N/A	N/A	N/A	30.0	30.0	26.7
CtxSub	N/A	N/A	N/A	N/A	47.3	43.6	43.2
Shuf	N/A	N/A	N/A	N/A	47.3	41.8	41.5
Syno	N/A	N/A	N/A	N/A	45.5	37.3	40.0
Sub^2 (balanced tree)	✓	N/A	N/A	✓	46.4	**44.5**	44.1
	✓	N/A	N/A	✗	48.2	43.6	
	✗	N/A	N/A	✓	31.8	36.4	
	✗	N/A	N/A	✗	43.6	37.3	
Sub^2	✓	✓	senti.	✓	49.1	**45.5**	44.7
	✓	✓	senti.	✗	45.5	43.6	
	✓	✓	const.	✓	47.3	42.7	
	✓	✓	const.	✗	45.5	41.8	
	✓	✓	✗	✓	42.7	41.8	
	✓	✓	✗	✗	41.8	38.2	
	✓	✗	N/A	✓	36.4	42.7	
	✓	✗	N/A	✗	37.3	38.2	
	✗	✓	const.	✓	48.2	43.6	
	✗	✓	const.	✗	35.5	37.3	
	✗	✓	✗	✓	48.2	41.8	
	✗	✓	✗	✗	45.5	40.0	

Table 10: Accuracy ($\times 100$) on the AG News and SST dataset (few-shot set 2). The best devtest numbers in each section are bolded .