ALP: Data Augmentation using Lexicalized PCFGs for Few-Shot Text Classification

Hazel Kim[1,2], Daecheol Woo[1], Seong Joon Oh[2], Jeong-Won Cha[3], Yo-Sub Han[1]

[1] Yonsei University, Seoul, Republic of Korea
[2] NAVER AI Lab,
[3] Changwon National University, Changwon, Republic of Korea
{hazel.kim, dwoo}@yonsei.ac.kr, coallaoh@gmail.com, jcha@changwon.ac.kr, emmous@yonsei.ac.kr

Abstract

Data augmentation has been an important ingredient for boosting performances of learned models. Prior data augmentation methods for few-shot text classification have led to great performance boosts. However, they have not been designed to capture the intricate compositional structure of natural language. As a result, they fail to generate samples with plausible and diverse sentence structures. Motivated by this, we present the *data Augmentation using Lexicalized Probabilistic context-free grammars (ALP)* that generates augmented samples with diverse syntactic structures with plausible grammar. The lexicalized PCFG parse trees consider both the constituents and dependencies to produce a syntactic frame that maximizes a variety of word choices in a syntactically preservable manner without specific domain experts. Experiments on few-shot text classification tasks demonstrate that ALP enhances many state-of-the-art classification methods. As a second contribution, we delve into the train-val splitting methodologies when a data augmentation method comes into play. We argue empirically that the traditional splitting of training and validation sets is sub-optimal compared to our novel augmentation-based splitting strategies that further expand the training split with the same number of labeled data. Taken together, our contributions on the data augmentation strategies yield a strong training recipe for few-shot text classification tasks.

Introduction

Labeled data are an essential ingredient in training deep models. A major challenge in practice is the cost for collecting them. Data augmentation has provided a means to enlarge the training data without resorting to additional labeling cost (Shorten and Khoshgoftaar 2019). Training a text classifier is not an exception. Researchers have proposed different ways to augment text data to expand the labeled text data. These methods focus on diversifying the word choices while preserving the labels for the classification task. For example, Wei and Zou (2019) arbitrarily select target areas of examples for synonym swap and random insertion. Yu et al. (2018); Kumar, Choudhary, and Cho (2020) leverage pre-trained language models for model-based augmentation. Zhang, Yu, and Zhang (2020) create synthetic examples by softly combining input and output sequences.

Method	
Method	**Original**: The characters didn't seem to fit very well with the book.
EDA	the characters didnt book to fit very well with the seem. very characters seem to fit the well with the book.
BT	The characters didn't seem to fit the book that well. The characters didn't seem to fit the book very well.
SSMBA	the the don's seem to fit very well with the. the characters of the also seem to be be up with the book.
ALP (Ours)	The scripts didn't get together very comfortably with the outlook. The prospect didn't satisfy very advantageously.

Table 1: **Augmented samples.** ALP generates the most diverse, syntactically plausible, and label-preserving augmented samples. EDA: Easy Data Augmenatation, BT: Back-translation, and SSMBA: Self-Supervised Manifold based Data Augmentation.

The prior augmentation methods have successfully diversified the samples and improved classification performances with a small number of labeled data. However, we point out a crucial shortcoming shared by those methods: they do not take into account the intricate compositional structure in natural language. Replacing words and modifying structures of a sentence without linguistic rules and guidance are likely to alter the syntax and semantics. It can be seen in Table 1 that those methods (EDA, BT, and SSMBA) fail to generate samples with plausible and diverse sentence structures.

These limitations motivate us to design a grammar-based augmentation method, which generates more plausible augmented data that better respects the syntax. We present ALP: data **A**ugmentation using **L**exicalized **P**robabilistic context-free grammars for few-shot text classification. We use lexicalized PCFG (or L-PCFG) parse trees to consider both constituents and dependencies to capture two very different views of syntax in text data and produce a syntactic frame that maximizes a variety of word choices in a syntactically preservable manner without specific domain experts.

Our approach aims to reach theoretical guarantees of increasing both the amount and the diversity of a given dataset in a pretty label-preserving manner. As such, ALP is designed to produce augmented samples with diverse sentence structures, each still respecting the linguistic rules and preserving the corresponding class label. The ALP samples in Table 1 exemplify such augmented data. We demonstrate the empirical superiority of ALP augmentation in the few-shot

text classification benchmarks.

Recognizing the importance of the amount of data for few-shot learning tasks, we further contribute novel train-val splitting strategies that are relevant when data augmentation methods come into play. The train-val split is often regarded as a fixed constraint for a learning problem. However, we argue that the train-val split itself could be regarded as part of the model development pipeline. This viewpoint is echoed by researchers in meta-learning (Setlur, Li, and Smith 2020; Saunshi, Gupta, and Hu 2021; Bai et al. 2021), where they even suggest unconventional splitting methods like "train-train" that trains and validates the models on the *identical* data split. We note that, when data augmentation step is part of the game, there are even further creative possibilities to split the training and validation sets. For example, under the same number of labeled data $\mathcal{S}$ and a fixed augmentation budget, we show that training on the entire augmented labeled data aug($\mathcal{S}$) and validating on the original data $\mathcal{S}$ brings about further gains in performance across the board.

In summary, we contribute (1) a grammar-based data augmentation method that diversifies sentence structures and (2) novel train-val splitting strategies that can be combined with general data augmentation methods.

Background

In this section, we explain background research work on topics related to our core contributions: semi-supervised learning, data augmentation, and the train-val split.

Semi-Supervised Learning (SSL)

SSL leverages both labeled and unlabeled data for learning a discriminative task. Early work has embraced the viewpoint that SSL is most useful when a large amount of noisy, unlabeled data source is accessible on top of a small number of clean, labeled samples (Chen et al. 2019; Li, Socher, and Hoi 2020). As such, under the SSL setup, prior studies have focused on applying data augmentation on the unlabeled data, rather than on the labeled ones. Xie et al. (2020) and Chen, Yang, and Yang (2020) achieve state-of-the-art model performance from noisy unlabeled data using advanced data augmentation methods in text classification with limited data. Contrary to the common wisdom, we find that data augmentation on the clean, labeled data also aids the generalization. Under the setup with extremely limited data (k-shot learning), the benefits of the increased amount of data outweighs the additional noise introduced by the augmentation algorithm.

Data Augmentation

Data augmentation refers to a general training technique for machine learning where the original training data are expanded to a larger set, without resorting to external sources of further data (Shorten and Khoshgoftaar 2019). Data augmentation has proven highly effective especially for deep learning and bigger models, as they generally benefit from greater amounts of data. Data augmentation has proved useful both for labeled and unlabeled samples. For unlabeled

samples, as typically done in SSL, data augmentation is applied in the form of consistency regularization (Xie et al. 2020; Chen, Yang, and Yang 2020; Chen et al. 2021).

As mentioned above, this work focuses on the data augmentation on labeled samples. For labeled samples, the augmentation algorithm aims to preserve the semantics of the samples, while enhancing their diversity. Prior approaches on data augmentation for labeled text samples are limited in that they fail to observe linguistic rules and syntax. For example, they randomly select target areas for synonym swap and random insertion (Wei and Zou 2019) or leverage pre-trained language models for model-based augmentation (Yu et al. 2018; Kumar, Choudhary, and Cho 2020). (Zhang, Yu, and Zhang 2020) create synthetic examples by softly combining input and output sequences. While they improve the model generalization, replacing words in a sentence without linguistic rules and guidance is likely to generate samples that are less realistic and plausible. These limitations have motivated us to design our grammar-based augmentation method. We demonstrate the enhanced preservation of both the semantic and syntactic information in samples.

Train-Val Split

Machine learning focuses on the generalization beyond the particular training samples. Thus, a suitable segregation of data according to their dedicated uses is crucial in developing models and evaluating their generalization capabilities (Hastie, Tibshirani, and Friedman 2001). Practitioners typically introduce a three-way split: train, validation, and test. Train split is used for fitting model parameters, for example via gradient descent for deep models. The validation split is used for the outer optimization problem, where hyperparameters controlling the generalization performance are fitted through black-box optimization algorithms (Feurer and Hutter 2019) or heuristics (Gencoglu et al. 2019). The test split is the ultimate test ground for the model; the discussion of the test set is out of the scope.

The train-val split is often considered a given condition in machine learning dataset and literature. However, from a practical point of view, the ultimate crude material for building a model is the set of labeled data, which comes before the protocol for splitting it into the train and validation splits. In other words, the very protocol for the train-val split shall also be part of the overall pipeline for model building and be subject to scientific studies and solution-seeking. This view is shared by researchers in meta-learning (Setlur, Li, and Smith 2020; Saunshi, Gupta, and Hu 2021). (Bai et al. 2021) have even questioned the need for the train-val split and have proposed to use the entire labeled data for both training the parameters and validating the hyperparameters (the "train-train" method). In this work, we inherit this viewpoint, and consider various strategies for the train-val split. The space of possible splitting strategies is greatly expanded by the inclusion of the data augmentation stage.

Data Augmentation using L-PCFGs

Our data augmentation using lexicalized PCFGs (ALP) maximizes a variety of word choices within grammatical rules. This section introduces the ALP algorithm.
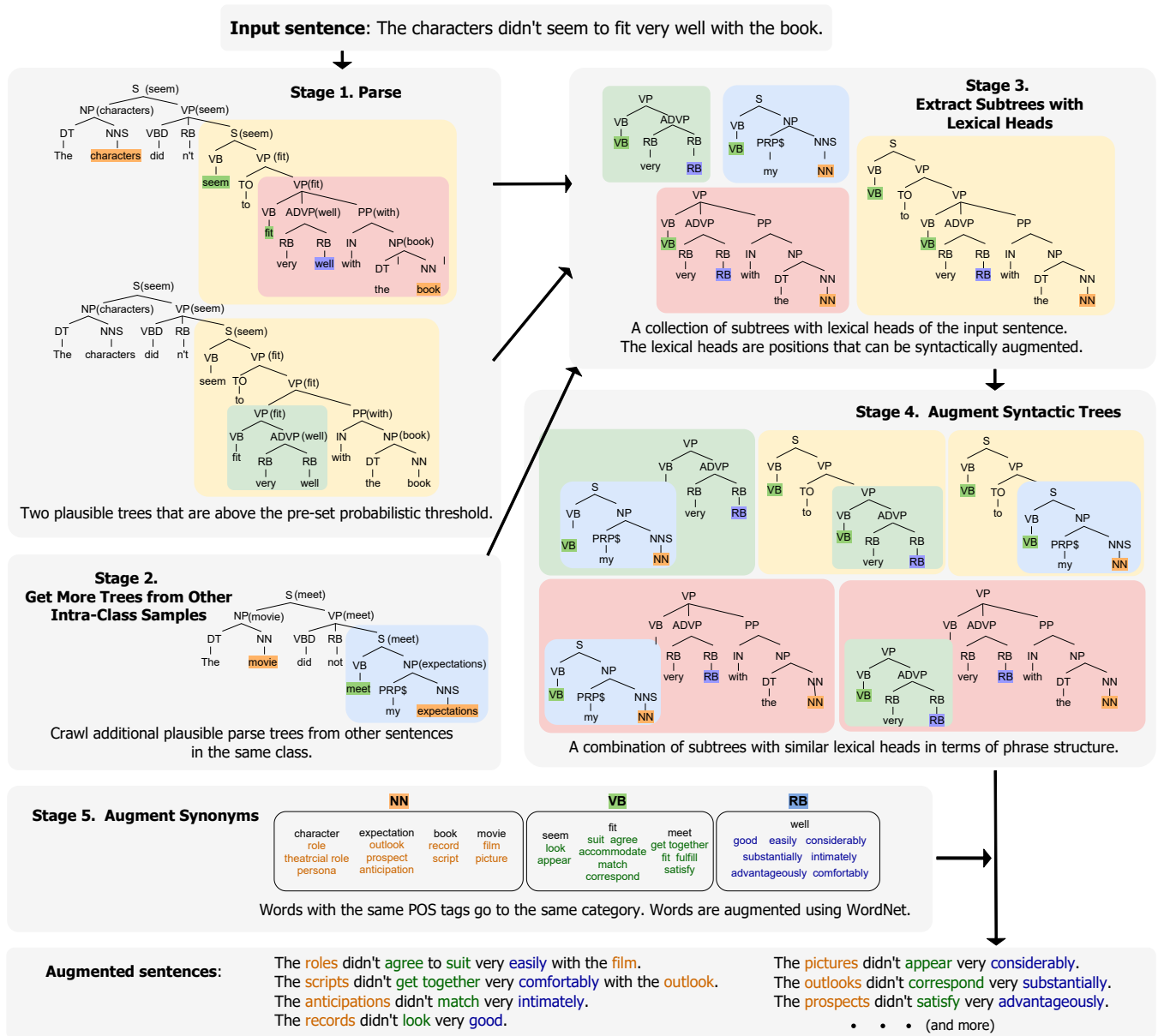
Figure 1: **Overview of ALP.** The step-by-step algorithm for generating augmented sentences using lexicalized PCFGs.

Lexicalized PCFGs

To explain the Lexicalized PCFGs, we first introduce the context-free grammar (CFG). CFG is a list of rules that define well-structured sentences in a language. Each rule has a left-hand side α that identifies a syntactic category, and a right-hand side β that defines its alternative component parts. Syntactic categories include NP for noun phrase and VP for verb phrase.

Probabilistic context-free grammars (PCFGs) have been an important probabilistic approach to syntactic analysis (Lari and Young 1990; Jelinek, Lafferty, and Mercer 1992). It assigns a probability $q(\alpha \to \beta)$ to each parse tree $\alpha \to \beta$ allowed by the underlying CFGs. The parameter $q(\alpha \to \beta)$ is the conditional probability of choosing rule $\alpha \to \beta$, given that the α is on the left-hand-side of the rule (Collins 2013). Under a particular type of ambiguity such as a preprositional-phrase (PP) attachment ambiguity, the PCFG model chooses a single parse tree between the two that have identical rules, depending on the value of $q(\text{VP} \to \text{VP PP})$ and $q(\text{NP} \to \text{NP PP})$. The probabilistic parser chooses a tree with $\text{VP} \to \text{VP PP}$ if $q(\text{VP} \to \text{VP PP}) > q(\text{NP} \to \text{NP PP})$. The probabilistic component is crucial in our application because we aim to generate a diverse set of perturbations of a sentence based on multiple plausible hypotheses.

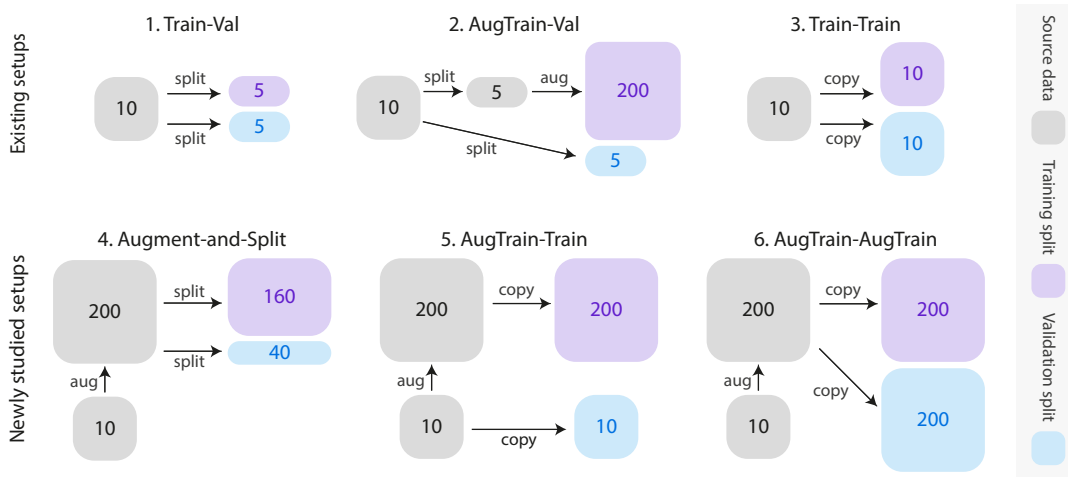Lexicalized PCFGs (L-PCFGs) extends PCFGs by in-

Figure 2: **Various train-val splitting strategies.** We match the resources used by the six splitting strategies: the number of labeled source data (10 samples per class) and the number of augmented samples (200 samples per class) if there is any. We adopt and extend the names "train-val" and "train-train" from the meta-learning literature (Bai et al. 2021).

corporating lexical information to further disambiguate the parsing decisions. For example, L-PCFGs enrich the graph S → NP VP into S(bored) → NP(movie) VP(bored). The L-PCFG model is sensitive to lexical information but becomes robust because the lexical heads with the grammar rules extend the number of parameters and smooth the estimates in the model. Lexical information serves as the additional criteria to produce parse trees that are valid in the corresponding grammar rules.

Data Augmentation using L-PCFGs

We propose to use the rule probabilities and lexical information to diversify the grammatical choices from the limited resources. We extract many plausible subtrees using probabilistic threshold and consider lexical heads as the position information to swap and augment the syntactic structure. We substitute synonymous words within the syntactic frame.

Stages 1–2. Parse with probabilistic threshold to select more trees We first extract all the valid parse trees using probabilistic threshold τ, instead of picking a single tree with the maximum probability. As shown in Figure 1, the input sentence generates two valid trees that include a PP attachment ambiguity. Unlike how regular PCFGs behave, ALP picks both trees with VP → VP PP and NP → NP PP if $q(\text{VP} \rightarrow \text{VP PP}) > \tau$ and $q(\text{NP} \rightarrow \text{NP PP}) > \tau$. We use all the plausible trees generated from sentences in the same class if they are available. We expect to maximize the candidate trees to use them in the syntactic augmentation stage.

Stage 3. Extract subtrees with lexical heads After collecting all the plausible tree rules to use, we extract subtrees using lexical heads as the position information to swap. Figure 1 shows an example of VP as the lexical head. ALP swaps sub-subtrees with other types of lexical heads such as NP or PP within the subtrees if available.

Stages 4–5. Augment and Generate While ALP extracts grammar rules from a starting input sentence to terminal

rules, its augmentation procedure starts from bottom to top. We gather words with the same POS tags such as NN or VB into one pool as shown in Stage 5 of Figure 1. We combine all the available words from different sentences in the same class to augment as many samples as possible, using the WordNet synonyms. The collected words have the freedom to be replaced with other words in the same POS-tagged pool. We then fill in augmented syntactic trees generated from Stage 4. The augmented syntactic phrases now replace subtrees extracted with the lexical information within the original sentence. This way, ALP preserves the label compatibility while augmenting data in the greatest number of ways.

Train-Val Split with Augmented Data

As explained in the Background section, the train-val split is crucial for ensuring good generalization performance of machine learning models. While the training and validation splits are often considered the given protocols for the sake of fair comparison among methods, the splitting of training and validation sets itself can be regarded as part of the model development framework (Hastie, Tibshirani, and Friedman 2001; Saunshi, Gupta, and Hu 2021). Our second contribution is based on this perspective.

In this section, we delve into different possibilities of assigning the training and validation splits of a labeled source dataset, in particular in the presence of a data augmentation procedure. The motivation for searching over multiple splitting strategies is the same as that for applying data augmentation: enlarging the labeled training data for the model.

We provide conceptual diagrams for possible train-val splitting strategies based on the common labeled source dataset in Figure 2. Experiments on those strategies will be presented in the Experiments section.

1–3. Existing splitting strategies The upper row in Figure 2 corresponds to existing train-val splitting. One may

split the source into disjoint train and validation splits (1. Train-Val) or may additionally apply a data augmentation algorithm on the training split (2. AugTrain-Val). In the meta-learning field, it has been argued that using the source dataset both for parameter tuning and model selection may enhance the final performance (3. Train-Train) (Bai et al. 2021). The intuition is that, for low-data regime like few-shot learning, the importance of enlarging the training split outweighs the importance of segregating the validation data.

4. Augment-and-Split The data augmentation step opens up new search spaces for the splitting strategy. For example, one may first augment the source dataset and then split the augmented data according to the ratio that ensures the maximal size of the training split while allowing for a "just-right" amount of validation samples. One may control the ratio between the training and validation splits (e.g. 80:20) to find the right balance.

5. AugTrain-Train To ensure the purity and representativeness of the validation split, one may opt for keeping the source data for validating models, while using the augmented version of the entire dataset as the training split. This ensures a large number of training data as well as minimal noise for the validation split. The cost to pay here is the overlap between the training and test splits, which may hinder the model selection based on the generalizability. However, again, under the low-data regime, the enlarged training split may bring greater gain than the loss incurred by the lack of ability to select models that generalize well.

6. AugTrain-AugTrain In the extreme case, one may apply the "Train-Train" strategy on the augmented source data. Both the training and validation splits *are* the augmented source data. This setup additionally enlarges the validation split to perfectly overlap with the training split. This setup is meant as a sanity check that introducing noise on the validation split via data augmentation may hinder the optimal model selection and degrade the overall performance.

Experiments

In this section, we present experimental results on our contributions. We first show the superiority of ALP among recent data augmentation baselines utilized in semi-supervised learning (SSL) for few-shot text classification. We then present various train-val splitting strategies and propose an optimal strategy that appropriately combines data augmentation with the train-val splitting.

Experimental Setup

Our experiments investigate three different data augmentation methods other than ALP. We measure their performances on top of three state-of-the-art SSL methods on four benchmark text classification tasks. We explain the details of those experiments here.

Few-Shot Text Classification Usual few-shot learning refers to the setup where k samples per class are available for training and other disjoint k samples per class are available for validation, where k is usually small (k-shot learning). This means, in total, there are $2k$ labeled samples are

Dataset	#Classes	#Train	#Test	Length
AG News	4	120K	7.6K	80
SST-2	2	6.9K	1.8K	32
IMDB	2	250K	250K	256
Yahoo! Answer	10	1400K	600K	256

Table 2: **Dataset statistics.** We report the number of samples for the original datasets. For our k-shot classification experiments, we subsample a small portion for training and validation splits; the rest is used as the unlabeled data for semi-supervised learning.

available for each class. We train models under the SSL fashion, by additionally utilizing the remaining data as the *unlabeled* source. In our experiments, we consider applying data augmentation methods on the $2k$ labeled samples. For more strategic ways to combine data augmentation with the splitting strategies for the $2k$ labeled samples, see the Section "Train-Val Split with Augmented Data" and Figure 2. Unless specified otherwise, we use the "Train-Val" and "AugTrain-Val" schemes for vanilla training and the data-augmented versions, respectively. We have conducted experiments with 5 random samplings of the labeled data, shuffling of data being presented to the models, and the weight initialization. We report the mean and standard deviation.

Datasets We conduct experiments on four benchmark text classification tasks as summarized in Table 2. SST-2 (Socher et al. 2013) and IMDB (Maas et al. 2011) are used for sentiment classification for movie reviews but with different sequence lengths per sample. AG News (Zhang, Zhao, and LeCun 2015) and Yahoo (Chang et al. 2008) are used for topic classification in regards to news articles and question and answer pairs from the Yahoo! Answers website, respectively.

Baseline Augmentation Methods We consider three data augmentation methods as our baselines. Easy Data Augmentation (EDA) (Wei and Zou 2019) is a heuristic method that randomly replaces, inserts, swaps, and deletes words. We use the official code with the recommended insertion, deletion, and swap ratios the authors provided. Unsupervised Data Augmentation (Xie et al. 2020), or back-translation (BT), is another common method that translates data to and from a pivot language to generate paraphrases. We select German as intermediate languages for back-translation using FairSeq and set 0.9 as the random sampling temperature. Self-Supervised Manifold Based Data Augmentation (SSMBA) (Ng, Cho, and Ghassemi 2020) generates pseudo-labels by using pre-trained masked language models as a denoising auto-encoder. SSMBA uses the corruption and reconstruction function to fill in the masked portion and thus augment the data. We use the default masked proportion and the pre-trained weights provided by the authors. Throughout the experiments we generate 200 samples for all augmentation methods, unless specified differently.

Base Semi-Supervised Learning (SSL) Approaches We introduce three state-of-the-art SSL approaches to explore their compatibility with data augmentation techniques. Self-

	Labeled data		Data augmentation methods				
	#train	#val	No augmentation	+EDA	+BT	+SSMBA	+ALP
AG News	5	5	77.73 ± 4.91	78.89 ± 2.64	78.66 ± 4.47	78.65 ± 1.90	**82.30** ± 3.34
	10	10	82.13 ± 3.99	80.72 ± 1.61	83.80 ± 3.48	84.68 ± 1.07	**86.18** ± 1.27
SST-2	5	5	54.38 ± 3.79	56.22 ± 2.56	55.77 ± 4.64	56.34 ± 5.42	**63.40** ± 2.33
	10	10	61.82 ± 5.85	53.96 ± 1.40	62.05 ± 5.03	59.05 ± 5.70	**69.72** ± 2.56
IMDB	5	5	54.75 ± 3.01	60.32 ± 8.38	65.33 ± 6.54	66.43 ± 9.10	**67.05** ± 10.29
	10	10	68.49 ± 7.42	69.80 ± 5.75	70.41 ± 8.96	63.36 ± 6.07	**71.29** ± 6.08
Yahoo!	5	5	47.77 ± 0.77	**55.49** ± 3.82	54.59 ± 3.68	53.17 ± 7.15	55.19 ± 3.64
	10	10	58.81 ± 3.02	63.12 ± 2.61	59.35 ± 3.24	61.50 ± 0.48	**64.16** ± 1.40

Table 3: **Comparison of data augmentation methods.** We use the Self-Training (ST) semi-supervised learning setup with k-shot samples for both training and validation, where $k \in \{5, 10\}$. The Train-Val and AugTrain-Val splits in Figure 2 have been used for No-augmentation and augmented variants, respectively.

	AG News		SST-2		IMDB		Yahoo!	
Methods	5	10	5	10	5	10	5	10
UST	79.65	83.85	57.13	62.71	63.60	73.63	55.49	63.54
+ BT	81.61	83.43	57.22	67.76	67.14	**83.21**	61.78	63.91
+ SSMBA	83.05	86.32	48.76	57.00	61.05	66.82	**62.81**	63.63
+ ALP	**84.72**	**87.41**	**73.22**	**78.01**	**71.32**	76.33	61.20	**66.89**
MixText	81.14	87.11	51.46	50.91	68.09	72.87	66.60	67.40
+ BT	82.04	70.18	51.29	51.78	62.44	74.12	66.19	66.00
+ SSMBA	83.47	69.03	51.95	52.39	54.11	61.36	65.32	67.14
+ ALP *	**83.50**	**87.72**	**52.44**	**57.06**	**84.35**	**84.16**	**67.31**	**67.81**

Table 4: **ALP with the state-of-the-art SSL methods.** k-shot samples have been used for both training and validation, where $k \in \{5, 10\}$.

Splitting schemes	#Samples		Dataset				Average
	train	val	AGNews	SST-2	IMDB	Yahoo!	
1. Train-Val	5	5	77.73	54.38	54.75	47.77	58.66
2. AugTrain-Val	200	5	82.30	63.40	64.89	55.19	66.45
3. Train-Train	10	10	80.41	57.45	60.48	50.76	62.28
4. A-and-S (50:50)	100	100	80.51	**72.33**	57.70	54.59	66.28
4. A-and-S (80:20)	160	40	83.23	71.14	71.03	59.44	72.13
4. A-and-S (90:10)	180	20	78.59	56.01	66.27	60.28	65.29
5. AugTrain-Train	200	10	**83.45**	63.64	**80.41**	62.82	**72.40**
6. AugTrain-AugTrain	200	200	82.53	64.14	62.53	59.64	67.21

Table 5: **Comparison of train-val splitting strategies.** See Figure 2 for the description of each method. We match the resources used by the six splitting methods: the number of labeled source data (10 samples per class) and the number of augmented samples (200 samples per class) if there is any. We use ALP for the data augmentation. "A-and-S" refers to the Augment-and-Split scheme.

training (ST) (Yarowsky 1995) is a classic SSL approach using the teacher-student mechanism (Yarowsky 1995). The base teacher model trained on labeled data trains the student model on unlabeled data to prevent overfitting and generalize well to unseen data. Uncertainty-aware self-training (UST) (Mukherjee and Awadallah 2020) uses stochastic dropouts and selectively samples unlabeled examples to train the model. MixText (Chen, Yang, and Yang 2020) is another SSL method using a novel data augmentation method TMix that creates virtual training samples by linearly interpolating pairs of labeled samples over their hidden-layer embeddings. MixText guesses labels for unlabeled data and leverage TMix on both labeled and unlabeled data. We use ST as the base SSL method.

Evaluating ALP

We evaluate the performance of ALP augmentation against existing methods. We then explain the performance boost in terms of the exceptionally high degrees of diversity for ALP-augmented samples.

Comparison against Other Augmentation Methods Table 3 shows the comparison among data augmentation methods when applied to the ST semi-supervision method. We observe that the prior data augmentation methods tend to enhance the performances, with a few critical exceptions. For example, for SST-2 dataset with $k = 10$, EDA drops the performance from 61.82% to 53.96%. Such aberrations occur at least once for EDA, BT, and SSMBA among the benchmarks considered. On the other hand, ALP *uniformly improves* the performance on augmented data across the board. Moreover, ALP outperforms all previous data augmentation methods by quite a margin in general. For example, on SST-2 with $k = 10$, ALP achieves 69.72%, compared to the second-best method BT with 62.05%.

Compatibility with Various SSL Approaches We verify that ALP is applicable to any deep learning model by showing its performance with other semi-supervised learning (SSL) approaches, such as UST and MixText. See Table 4 for the results. We observe that ALP generally improves the classification accuracy for both UST and MixText. We note that BT and SSMBA often fails to improved the performance for MixText; for IMDB $k = 5$, they drop the accuracy by 5.44%p and 6.73%p, respectively.

	BT	SSMBA	ALP	Original
Fidelity	79.3±4.8	77.6±4.8	81.8±4.6	85.2±3.8
SelfBLEU-{2/5}	0.75 / 0.49	0.54 / 0.32	0.33 / 0.08	0.00 / 0.00

Table 6: **Semantic fidelity and text diversity.** We measure semantic fidelity using BERT classifier and text diversity using SelfBLEU-n where $n \in \{2, 5\}$.

k	Method	No-aug	EDA	BT	SSMBA	ALP
5	AugTrain-Val	58.66	62.73	63.59	63.65	**66.45**
	AugTrain-Train	62.28	64.42	67.40	68.09	**72.40**
10	AugTrain-Val	67.81	66.90	68.90	67.15	**72.83**
	AugTrain-Train	70.97	67.31	73.15	68.91	**74.89**

Table 7: **Data augmentation with the best train-val split.** See Figure 2 for an overview of splitting methods.

Semantic Fidelity and Text Diversity of ALP We provide an explanation for the superiority of ALP against baseline methods in terms of the semantic fidelity and text diversity in augmented samples. ALP generates sentences with decent semantic fidelity, as shown in Table 6. We use a BERT-Base classifier fine-tuned on all available labeled samples to measure classification accuracies on generated data (Kumar, Choudhary, and Cho 2020). Higher scores indicate the preservation of class labels in generated data. ALP has average fidelity scores (81.8%) closest to the original (85.2%) across four different benchmarks. We evaluate text diversity for augmentation methods by measuring Self-BLEU scores (Zhu et al. 2018) that assess how an augmented sentence resembles the original one. Table 6 shows that ALP has the lowest Self-BLEU scores, which imply high diversity of the text. The diversity of sentence structure and word choices while preserving the label compatibility has an instrumental role in boosting the model performances; ALP is designed for that.

Train-Val Split with Data Augmentation

We empirically test the strategies for splitting training and validation sets from a labeled source data. As mentioned earlier, the introduction of data augmentation into the pipeline results in a multiplicity of splitting strategies in Figure 2. Which splitting scheme will yield the best gain?

Comparison of Train-Val Splitting Strategies We investigate methods for splitting training and validation sets while exploring ways of taking full advantage of available labels to optimize data augmentation performance on labeled data. To make a fair comparison, we fix the labeled source data with $2k = 10$ samples. We further fix the computational overhead due to data augmentation by fixing the number of augmented samples to 200. Figure 2 describes the setups and Table 5 shows the corresponding model performances. The Train-Val split is the standard setup for 5-shot text classification. The baseline result is 58.66% on average across the

benchmark datasets. Adding our ALP augmentation boosts the score to 66.45% (AugTrain-Val split). As an additional baseline, we test the Train-Train split, introduced by (Bai et al. 2021). As argued in the paper, we observe a mild improvement in performance (62.28%).

We now consider more creative splitting schemes. The Augment-and-Split scheme yields the average accuracy of 72.13% for the splitting ratio 8:2, greatly outperforming the previously considered splits like AugTrain-Val and Train-Train. The best average performance is reported by the AugTrain-Train split (72.40%), that uses the augmented source data for training and the original source data for validation. The other advantage of AugTrain-Train is that there is no additional hyperparameter attached, unlike the Augment-and-Split scheme.

We identify two lessons from the experiments here. First, few-shot classification generally benefits from an increased size of training data. This is so important that it even outweighs the importance of information segregation between training and validation. Second, for the validation set, the cleanliness often matters more than its bulk. For example, blindly increasing its size through augmentation drops the performance (5.19%p drop from AugTrain-Train to AugTrain-AugTrain).

Compatibility of AugTrain-Train with Various Augmentation Methods We now turn to the question: does our best splitting strategy, AugTrain-Train, also yield the best results for data augmentation methods other than ALP? Table 7 shows the results. We observe that our novel split, AugTrain-Train, uniformly improves the performance for EDA, BT, and SSMBA for $k \in \{5, 10\}$. This validates the effectiveness of AugTrain-Train beyond ALP. We further confirm that under this new split, ALP is still the best-performing augmentation method.

Conclusions

We have introduced a novel text augmentation method, ALP, that considers the syntactic structure of the augmented samples. Using grammar-based mechanisms ALP increases the diversity of the sentence structures and the word choices in sentences, while preserving the semantic content. With the exceptionally high level of diversity, ALP outperforms existing text augmentation methods on the few-shot text classification tasks on four real-world benchmarks. Given the importance of securing a large amount of labeled training data, we also explore novel train-val splitting schemes for few-shot classification task. We show that the usual disjoint training and validation splits are in fact sub-optimal and propose a novel scheme that uses the augmented source data as the training split and the un-augmented original source as validation. The two contributions are orthogonal. Together, they comprise a powerful recipe for greatly enhancing the few-shot classification scores across the board.

Acknowledgements

References

Bai, Y.; Chen, M.; Zhou, P.; Zhao, T.; Lee, J. D.; Kakade, S. M.; Wang, H.; and Xiong, C. 2021. How Important is the Train-Validation Split in Meta-Learning? In *Proceedings of the 38th International Conference on Machine Learning, ICML*, Proceedings of Machine Learning Research, 543–553. PMLR.

Chang, M.; Ratinov, L.; Roth, D.; and Srikumar, V. 2008. Importance of Semantic Representation: Dataless Classification. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI*, 830–835. AAAI Press.

Chen, J.; Tam, D.; Raffel, C.; Bansal, M.; and Yang, D. 2021. An Empirical Survey of Data Augmentation for Limited Data Learning in NLP. *CoRR*, abs/2106.07499.

Chen, J.; Yang, Z.; and Yang, D. 2020. MixText: Linguistically-Informed Interpolation of Hidden Space for Semi-Supervised Text Classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL*, 2147–2157. Association for Computational Linguistics.

Chen, P.; Liao, B.; Chen, G.; and Zhang, S. 2019. Understanding and Utilizing Deep Neural Networks Trained with Noisy Labels. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, 1062–1070. PMLR.

Collins, M. 2013. Probabilistic context-free grammars (pcfgs). *Lecture Notes*.

Feurer, M.; and Hutter, F. 2019. Hyperparameter optimization. In *Automated machine learning*, 3–33. Springer, Cham.

Gencoglu, O.; van Gils, M.; Guldogan, E.; Morikawa, C.; Süzen, M.; Gruber, M.; Leinonen, J.; and Huttunen, H. 2019. HARK Side of Deep Learning–From Grad Student Descent to Automated Machine Learning. *arXiv preprint arXiv:1904.07633*.

Hastie, T.; Tibshirani, R.; and Friedman, J. 2001. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc.

Jelinek, F.; Lafferty, J. D.; and Mercer, R. L. 1992. Basic methods of probabilistic context free grammars. In *Speech Recognition and Understanding*, 345–360. Springer.

Kumar, V.; Choudhary, A.; and Cho, E. 2020. Data Augmentation using Pre-trained Transformer Models. *CoRR*, abs/2003.02245.

Lari, K.; and Young, S. J. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer speech & language*, 4(1): 35–56.

Li, J.; Socher, R.; and Hoi, S. C. H. 2020. DivideMix: Learning with Noisy Labels as Semi-supervised Learning. In *8th International Conference on Learning Representations, ICLR*.

Maas, A. L.; Daly, R. E.; Pham, P. T.; Huang, D.; Ng, A. Y.; and Potts, C. 2011. Learning Word Vectors for Sentiment Analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies,*

Proceedings of the Conference, 142–150. The Association for Computer Linguistics.

Mukherjee, S.; and Awadallah, A. H. 2020. Uncertainty-aware Self-training for Few-shot Text Classification. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS*.

Ng, N.; Cho, K.; and Ghassemi, M. 2020. SSMBA: Self-Supervised Manifold Based Data Augmentation for Improving Out-of-Domain Robustness. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*, 1268–1283.

Saunshi, N.; Gupta, A.; and Hu, W. 2021. A Representation Learning Perspective on the Importance of Train-Validation Splitting in Meta-Learning. In *International Conference on Machine Learning*, 9333–9343. PMLR.

Setlur, A.; Li, O.; and Smith, V. 2020. Is Support Set Diversity Necessary for Meta-Learning? *arXiv preprint arXiv:2011.14048*.

Shorten, C.; and Khoshgoftaar, T. M. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1): 1–48.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A.; and Potts, C. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1631–1642.

Wei, J. W.; and Zou, K. 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*, 6381–6387. Association for Computational Linguistics.

Xie, Q.; Dai, Z.; Hovy, E. H.; Luong, T.; and Le, Q. 2020. Unsupervised Data Augmentation for Consistency Training. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems*.

Yarowsky, D. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, 189–196.

Yu, A. W.; Dohan, D.; Luong, M.; Zhao, R.; Chen, K.; Norouzi, M.; and Le, Q. V. 2018. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. In *6th International Conference on Learning Representations, ICLR 2018*.

Zhang, R.; Yu, Y.; and Zhang, C. 2020. SeqMix: Augmenting Active Sequence Labeling via Sequence Mixup. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*, 8566–8579. Association for Computational Linguistics.

Zhang, X.; Zhao, J. J.; and LeCun, Y. 2015. Character-level Convolutional Networks for Text Classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, 649–657.

Zhu, Y.; Lu, S.; Zheng, L.; Guo, J.; Zhang, W.; Wang, J.; and Yu, Y. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 1097–1100.