Expansile Validation Dataset (EVD): Towards Resolving The Train/Validation Split Tradeoff

Anonymous Authors[1]

Abstract

The standardized machine learning algorithm is built upon a train/validation/test split protocol where the testing set is usually not presented to the algorithm developer in reality. In particular, the ratio (and mechanism) of splitting out a validation set is a pivotal meta-parameter that perhaps embodies a vexing tradeoff — with more samples being placed for validation, the stabler and more robust the model evaluation will become; yet this effectively results in fewer samples being utilized for training that may lead to a performance drop. In this article, we attempt to resolve this trouble by proposing EVD. In particular, the EVD promotes a *zero- or few-shot validation* framework where its major idea is to enable a generated validation sample pool then to systematically identify the samples that may suit best for validation purposes. The EVD manifests a dynamic process of assembling a validation set that 1) is initialized by a core-set from the auxiliary dataset or the validation fold in classic validation methods like cross validation, 2) is further expanded strategically to mitigate feature distribution shift. Empirical results of the EVD on the tabular data, image, and text problems successfully justify its validity, judging by multiple metrics. We also compared EVD against some previous related works, such as the J-K fold, where EVD expresses its superiority.

1. Introduction

In recent years, the majority of existing works in the machine learning community have been focused on enhancing model performance, such as the novel learning paradigms, improved optimization algorithms, model architectures, etc.

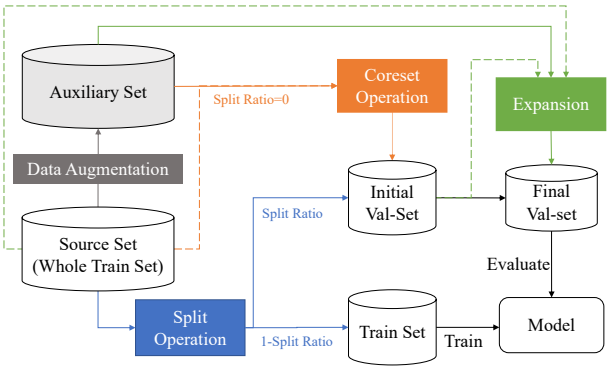


Figure 1. The framework of EVD. Given a source set and a split ratio, we aim to gather a suitable validation set. The blue and orange processes represent the two validation set initialization methods at different split ratios, respectively, and the green process represents the expansion operation. Notice that the dotted lines indicate that the sets on the starting points are used only as references.

As we look at the spectrum of the literature, an often neglected but indispensable component is the validation process. Specifically, before the final evaluation of model performance on the testing set, a validation set is utilized on model selection, hyper-parameter tunning, early-stop, etc. On the other hand, the testing set is often not presented to the algorithm developer, particularly when the testing environment is posted to the deployment. In the academic environment, this inaccessibility is reflected by a protocol where the testing set cannot be scrutinized. It can only be used once at last without further tuning or adaptation.

Due to the absence of a testing set during development, how to split a validation set is crucial to obtain a well-fledged model. Consider the development process when the algorithm developer is provided with a labeled dataset. One must decide how many data samples to put in the validation pool. Annoyingly, given limited data points, the mass ratio between the training and validation set forms a vexing tradeoff. With more data points being placed for validation, the model evaluation is likely to become stabler, and overfitting can be mitigated (e.g., through early stopping); yet it causes

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

fewer samples to be utilized for training which may lead to performance deficit and vice versa.

Indeed, there have been a few methods attempting to resolve the tradeoff. For instance, the cross-validation mechanism is proposed to reduce the variance by forming several split tuples for training and validation as per a preset ratio. Besides, Moss et al. (2018) proposed to use repeated k-fold cross-validation to reduce the variance. And some works (Joseph & Vakayil, 2021; Budka & Gabrys, 2012) reduce the variance by various representative sampling methods. However, these aforementioned methods ubiquitously set the split ratio as a fixed meta-parameter and only focus on reducing the evaluation variance while neglecting the effect of the bias between the true performance and the estimated performance.

This article aims to propose a new framework to fully get rid of the split ratio tradeoff. Briefly, the core mechanism of the proposed method, EVD, can be summarized as follows; (i) identify a cover set of the whole train set from the generated dataset; (ii) further expand to mitigate the feature distribution distance. In other words, EVD resolves the inherent split ratio tradeoff thanks to its split-free nature. The built-in mechanism to select generated validation samples ensures sufficient quantity and controlled quality. Further, as the EVD facilitates both zero- and few-shot validation scenarios, we show that it can easily be incorporated with the prior validation methods such as the cross-validation mechanism. Through extensive experiments with a wide spectrum — covering image, text, and tabular data — we prove the validity of EVD. Compared with the previous few research online, EVD expresses its significant advantages by many metrics.

To sum up, we make the following contributions through this work:

- We propose EVD, a split-free model validation framework that enhances testing performance and further relaxes the train-validation split tradeoff.
- The proposed EVD is a versatile method that can be combined seamlessly with other split-based validation mechanisms.
- Extensive experiments justify the validity of EVD in a variety of tasks. EVD alone also demonstrates much stronger performance and robustness against previous related work.

2. Related Work

2.1. Validation Dataset

Most studies on validation sets are based on the k-fold or the hold-out methods (Moss et al., 2018; Kohavi et al., 1995; Jiang & Wang, 2017; Székely & Rizzo, 2013; Jung, 2018; Tiittanen et al., 2021; Zeng & Martinez, 2000). They mainly aim to reduce variance, bias, and time cost. Specifically, variance means the variance between multiple training results with random seeds on the same model (Moss et al., 2018). And the bias means the gap between the model's evaluation results on the validation dataset and the training dataset (Zeng & Martinez, 2000).

Other works try to propose a scheme for the validation dataset generation. Joseph & Vakayil (2021) and Budka & Gabrys (2012) propose methods to sample a specific subset from the training set to generate a validation set. Perhaps the closest work to ours is attributed to Li et al. (2020b). In spite of the similarity, this method is restricted in the realm of tabular data. Further, without further adaptation, simply randomly drafting from an aux pool of generated data causes significant bias of the result, which potentially leads to unreliable model evaluation.

2.2. Data Augmentation

As mentioned in Section 3.2.1, the generated validation sample pool plays an important role in our algorithm. In the process of generating the validation set samples, we used a lot of data augmentation methods. Thus, in this section, we will introduce some methods for data augmentation for classification tasks in CV, NLP, and tabular fields, respectively.

In the field of computer vision, image geometric transformation (rotation, scale, etc.) and pixel modification (noise injection, adjusting HSV contrast and filtering, etc.) are the most common data augmentation methods (Shorten & Khoshgoftaar, 2019). Besides, masking part of the pictures is a data augmentation method that emerged in recent years (DeVries & Taylor, 2017; Zhong et al., 2020; Singh et al., 2018; Chen et al., 2020b; Li et al., 2020a). Mixup (Zhang et al., 2017) adds two images according to a certain ratio to generate a new sample, which has been widely used recently (Yun et al., 2019; Guo et al., 2019; Eaton-Rosen et al., 2018).

In the field of natural language processing, the data augmentation methods contain word-level transfer (replace, insert, swap and delete, etc.) (Wei & Zou, 2019; Feng et al., 2019), mask-based methods (Devlin et al., 2018), back translation (Sennrich et al., 2015; Li et al., 2020c), and dependency tree-based methods (Şahin & Steedman, 2019). Recently, Mixup-based methods also have been used thanks to Chen et al. (2020a) and others' work (Jindal et al., 2020; Cheng et al., 2020) following them.

For data stored in tabular form, we only introduce the SMOTE (Chawla et al., 2002) shortly. The method first selects examples, then draws a line between the examples in the feature space, and at last generates a new sample at a

point along that line.

Above all these methods, we take them to generate a validation sample pool. The specific methods are detailed in Section 3.2.1.

3. Method

3.1. Problem Setup

We start with the problem setup. Provided with a dataset, we let $\mathcal{X} \in \mathbb{R}^d$ and $\mathcal{Y} \in \mathbb{R}$ be the input and label space, respectively. We define a "source set" $\mathcal{D}^{src}$ as a compilation of the training and potential validation set, i.e., an unsplit training set that is fully labeled, as follows:

$$\mathcal{D}^{src} = \{(\mathbf{x}_i^{src}, y_i^{src})\}_{i=1}^{N}, \tag{1}$$

where N denotes the number of labeled instances in the pool, and $\mathbf{x}_i^{src} \in \mathcal{X}$, $y_i^{src} \in \mathcal{Y}$. In addition, we further define a test set as follows:

$$\mathcal{D}^{te} = \left\{\left(\mathbf{x}_i^{te}, y_i^{te}\right)\right\}_{i=1}^{M}, \tag{2}$$

where M denotes the number of labeled testing instances, and $\mathbf{x}_i^{te} \in \mathcal{X}$, $y_i^{te} \in \mathcal{Y}$. Further, following the normal machine learning research, we assume that the instances of both datasets are I.I.D drawn from an unknown underlying density distribution $\mathbb{F}(\mathcal{X}, \mathcal{Y})$, i.e.:

$$(\mathbf{x}_i, y_i) \overset{\text{iid}}{\sim} \mathbb{F} \tag{3}$$

From the source set $\mathcal{D}^{src}$, we further define a validation set, $\mathcal{D}^{val} = \mathsf{C}(\mathcal{D}^{src})$, where C is a (stochastic) collecting function that is facilitated by different seeds. For instance, C is a uniform sampling function in most of the research without further selection from the source set. This, in turn, yields a definition of a training set, $\mathcal{D}^{tr} = \mathcal{D}^{src} \setminus \mathcal{D}^{val}$, by eliminating the validation samples from the source set.

From here, certainly, one can train a mapping function f, $f(\mathcal{X}) \to \mathcal{Y}$. The validation protocol kicks by running an inference pass of f on $\mathcal{D}^{val}$ and comparing the predicted results with its gold counterpart. In short, we abbreviate this scoring process as simple as $\S(f, \mathcal{D}^{val})$ that $\S$ is a scoring function compiling the inference pass together with an evaluation metric.

Now, we may formally declare the bias and variance respectively from the perspective of validation. In particular, we assume there is an optimal function f' in the given hypothesis space $(\mathcal{X}, \mathcal{Y})$. The *bias* quantity is theoretically defined as the scored gap between the theoretical optimal and the practically yielded function (Zeng & Martinez, 2000; Budka & Gabrys, 2012):

$$\left|S(f, \mathcal{D}^{val}) - S(f', (\mathcal{X}, \mathcal{Y}))\right|. \tag{4}$$

On the other hand, a low degree of dispersion in the estimated scores based on $\mathcal{D}^{val}$ is very much ideal for machine learning problems, which reflects the stability of model evaluation. Thereby, The *variance* is defined by:

$$\frac{\sum_{i=1}^{n}\left(S(f, \mathsf{C}_i(\mathcal{D}^{src})) - \overline{S(f, \mathsf{C}(\mathcal{D}^{src}))}\right)^2}{n}, \tag{5}$$

where C_i denotes a set of collection functions under different random initialization. Usually, when the more labeled data is set for validation, i.e., gathered by C or C_i, the greater intersection extent between the chosen set becomes larger. This would effectively reduce the variance. However, this further causes a reduction of the labeled samples for training which may lead to an adverse increase of the bias quantity. In this setup for pursuing a decent validation set, we may term it as a "split tradeoff".

Essentially, it is always a plague to set the ratio between samples used for training and samples entertained for validation, drawn from the source set. We find that this line of research is very much absent. Most research has set this split ratio as a preset meta-parameter and stuck with it throughout the development. However, as important as it is, we attempt to scrutinize this problem and propose a systematically "split-free" solution.

3.2. Expansile Validation Dataset (EVD)

By proposing EVD, we entertain the possibility of relying (purely) on the generated samples to construct the validation process. Practically, this may pose a great number of advantages over the traditional split-based validation process because it saves the samples for training, which likely leads to performance gains. As a split-free method, EVD consists of the following critical components:

- A sample generation engine to construct an auxiliary dataset.
- An initialization process towards both zero- and few-shot validation scenarios. Namely, we devise the EVD to draft pivotal samples from an infinite pool from $\mathcal{D}^{aux}$, serving as a validation set at the starting point.
- To further reduce the bias, the EVD includes an iterative process aiming at a reduction of distribution deviance between the $\mathcal{D}^{val}$ and $\mathcal{D}^{src}$. Empirically, this iterative process is indeed necessary, mostly due to that $\mathcal{D}^{aux}$ has an infinitely large volume.

3.2.1. Generation of Auxiliary Dataset

The first step to achieving a split-free validation framework is to generate an auxiliary dataset. We primarily stick with the existing data augmentation methods such as

Mixup (Zhang et al., 2017), SMOTE (Chawla et al., 2002), and CutMix (Yun et al., 2019). While there is admittedly rich literature around data augmentation, this is mostly embedded into the training stage for feeding more samples to the training model instead of validation purposes. In particular, we conduct EVD on three fields:

- We use SMOTE for the tabular data.
- In the fields of computer vision, we employ CutMix, Mixup, GridMask as well as some traditional low-level image processing techniques.
- For language data, due to its discrete data, we substitute or insert words by contextualized word embeddings. In addition, sentences are easily augmented via random word corp, delete, swap, etc. And we utilize nlpaug [1] to implement the above operations.

Data generation is another achievable way to generate an auxiliary validation dataset. However, given the simplicity and computational advantages of the data augmentation methods, we stick EVD with them. We intend to leave the exploration of dataset generation methods, such as generative modeling (Goodfellow et al., 2014) and dataset distillation (Wang et al., 2018), to work further.

We use $\mathcal{D}^{aux}$ to denote the auxiliary validation set. The quantity of this set can be arbitrary large in theory, thanks to the continuous nature of the augmentation methods.

3.2.2. Validation set initialization

Once we have the generated auxiliary dataset, the next step of EVD is to initialize an initial validation set. Essentially we borrow the inspiration of the coreset (Sener & Savarese, 2017) method from the community of active learning. Inherently, the coreset is a dynamic programming problem. Given $\mathcal{D}^{src}$ of N instances and an integer $k < N$, its goal is to find a set $\mathcal{D}^{val}$ of k points from $\mathcal{D}^{aux}$ to minimize the maximum distances of any point of $\mathcal{D}^{src}$ to its closest center in $\mathcal{D}^{val}$. We adopt different distance metrics for different types of given problem/dataset: (i)-Euclidean distance for the tabular data (after one-hot vector processing for discrete columns); (ii)-Euclidean distance for image and text in the embedding space. We utilize pretrained models to cast the sample into the embedding space — a BERT model (Devlin et al., 2018) for text data and ResNet-110 (He et al., 2016) for images. Both models are pretrained.

This procedure suffices for a zero-shot validation setup, i.e., no original samples from the $\mathcal{D}^{src}$ are used. For the few-shot scenarios, the initialization is done directly as a random partition from the whole train set at a given split ratio.

[1] https://nlpaug.readthedocs.io/en/latest/

Algorithm 1 Val-Set Initialization

Input: the source dataset $\mathcal{D}^{src}$, the auxiliary dataset $\mathcal{D}^{aux}$, the split ratio r, the number of samples needed m
Initialize $\mathbb{S} = \{\}$.
if $r = 0$ **then**
  Initialize $\mathbf{s} = \{\}$.
  repeat
    $u = \arg\max_{i \in [N] \setminus \mathbf{s}} \min_{j \in \mathbb{S}} \Delta(\mathbf{x}_i, \mathbf{x}_j)$
    $v = \mathbf{findNearestNeighbor}(\mathcal{D}^{aux}, u)$
    $\mathbf{s} = \mathbf{s} \cup \{u\}$
    $\mathbb{S} = \mathbb{S} \cup \{v\}$
  until $|\mathbb{S}| = m$
else
  $\mathbb{S} = \mathbf{randomSample}(\mathcal{D}^{src}, ratio = r)$
end if
return $\mathbb{S}$

We provide the pseudo code for this process in the Algorithm 1 and refer the readers to the Coreset paper for more details and justifications.

3.2.3. Validation Set Iterative Expansion

As the $\mathcal{D}^{aux}$ is in theory infinite, we postulate that there is not necessary to stick to a static and fixed validation set. Notably, in the development of EVD we explore the possibility of forming a dynamic validation set. Specifically, we conduct the expansion based on the following steps:

- construct the representation of feature distribution;
- measure the distribution distance between $\mathcal{D}^{val}$ and $\mathcal{D}^{src}$;
- align the feature distribution via an iterative sampling strategy;

Class-level vs Feature-Level Distribution Alignment It is very much ideal to obtain a $\mathcal{D}^{val}$ that is completely consistent with the underlying distribution $F(\mathcal{X}, \mathcal{Y})$. However, since the density of $F(\mathcal{X}, \mathcal{Y})$ is (almost) impossible to achieve or estimate, there are two ways to align the generated/chosen validation set by using the source set $\mathcal{D}^{src}$ as a proxy to $F(\mathcal{X}, \mathcal{Y})$. On the one hand, one can align the class distribution. This is the most common strategy utilized in many prior works (Zeng & Martinez, 2000; Tiittanen et al., 2021). The implementation is straightforward, especially for discrete problems, such as classification, semantic segmentation, etc. Calculating the class distribution Y can maintain the class-level distribution when selecting instances from the source set. In spite of its simplicity, the class-level alignment is relatively coarse because it only focuses on the distribution of samples between classes and does not provide insight into the distribution of samples

within classes or the distribution of features within classes. This will still result in a large gap between the estimated score on the validation set and the true value(or the score on the test set). On the other hand, one can align the validation set in the feature space.

Feature distribution representation for tabular data. We characterize the feature distribution via normalized frequency distribution. Specifically for a tabular dataset $\mathcal{D}^{src} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ with L categories, where $\mathbf{x}_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. Suppose the feature space of $\mathcal{X}$ consists of numerical features and discrete features. For the discrete columns, we directly count the frequency of each feature from different entries (or categories). As for the numerical features, the frequency is counted after a binning mechanism by a given number of bins b. Note that b is set as a hyper-parameter, and it is uniformly set to 10 in implementation. Therefore, we define z_t^l as the distribution of feature t under label l

$$z_t^l = \left(z_{t,j}^l\right)_{j=1}^{b}, \;\; z_{t,j}^l = \frac{n_{t,j}^l}{\sum_{j=1}^{b} n_{t,j}^l}, \tag{6}$$

where $n_{t,j}^l$ denotes the frequency of feature t in the jth bin or the jth category. As a result, the feature distribution of $\mathcal{D}$ can be define proximally as

$$\mathbb{F}^*(\mathcal{D}) = \left(\left(z_t^l\right)_{t=1}^{d}\right)_{l=1}^{L}. \tag{7}$$

Despite its simplicity, this measurement can effectively incorporate both class-level and feature-level distributions for the given tabular dataset.

Feature distribution representation for image and text data. Unlike tabular data, we have to resort to the representation space to cope with the image and text data, assisted by pre-trained models. In the embedding space, we adopt the number of statistically-different bins (NDB) method (Richardson & Weiss, 2018) as a further descriptor. The reason for choosing this method is that it can efficiently measure the distance between the distributions of two sample groups. The NDB was implemented as a secondary-stage feature descriptor upon the embedding space. In specifics, for each class, the NDB is composed of two steps. First, we perform K-means clustering to fit on $\mathcal{S}^l$ ($\mathcal{S}^l$ is the embedded sample set in class l) at any arbitrary number of clusters K ($K \ll |\mathcal{S}|$, $|\mathcal{S}|$ is the size of embedded sample set $\mathcal{S}$). Then, Then, the second step is to transform, where we assign each sample to one of the K cells (bins) via the nearest (L2) of the K centroids. The representation of $\mathcal{S}^l$ can then be discretized into

$$\mathbb{F}_{NDB}^*(\mathcal{S}^l; K) = \left(\frac{n_k^l}{|\mathcal{S}^l|}\right)_{k=1}^{K}, \tag{8}$$

where K is the num of clusters or bins and n_k^l is size of the kth clusters in class l.

$$\begin{aligned}\mathbb{F}^*(\mathcal{D}) &= \left(\mathbb{F}^*(\mathcal{D}^l)\right)_{l=1}^{L} \\ &= \left(\mathbb{F}_{NDB}^*\left(f_e(\mathcal{D}^l); K\right)\right)_{l=1}^{L},\end{aligned} \tag{9}$$

where $\mathcal{D}$ is the input dataset which would be either $\mathcal{D}^{src}$ or $\mathcal{D}^{val}$ in our scenario, f_e is a fine-tuned feature extractor, L is the number of classes and $\mathcal{D}^l$ a subset of class l in $\mathcal{D}$.

Feature distribution distance Once we obtain the distributional descriptors for the dataset at all fields, the final missing piece is how to measure the distribution shift. The Wasserstein distance is a popular metric to serve this purpose (Arjovsky et al., 2017):

$$\mathcal{W}(u, v) = \inf_{\pi \in \Gamma(u,v)} \int_{\mathbb{R}\times\mathbb{R}} |x - y| \mathrm{d}\pi(x, y), \tag{10}$$

where $\Gamma(u, v)$ is the set of (probability) distributions on $\mathbb{R} \times \mathbb{R}$ whose marginals are u and v on the first and second factors, respectively. We simply utilize this metric for measuring a constructed validation set with the source counterpart. The function is denoted as d and defined as follows:

$$d(\mathcal{D}_l^{val}, \mathcal{D}_l^{src}) = \mathcal{W}\left(\mathbb{F}^*(\mathcal{D}_l^{val}), \mathbb{F}^*(\mathcal{D}_l^{src})\right), \tag{11}$$

where $l \in \{1, \ldots, L\}$, L is the number of classes. $\mathcal{D}_l^{val}$ and $\mathcal{D}_l^{src}$ are the subsets of class l in $\mathcal{D}^{val}$ and $\mathcal{D}^{src}$ respectively.

Iterative Expansion Procedure As we mentioned, the goal of validation set expansion of EVD is to construct a decent validation set with as least distribution shift from the source as possible. Because the volume of $\mathcal{D}^{aux}$ can be deemed infinite, we apply an iterative procedure to gradually find the best suitable samples. The full algorithm is illustrated in Algorithm 2. The procedure is applicable to both few-shot and zero-shot cases. While the difference between the two scenarios is the number of adding samples, the number for the former is more than that of the latter because a larger initial set is gathered in the latter, but the former directly retains a few-shot set.

3.3. Framework Instantiation

To this end, we finalize the instantiation of our framework, as illustrated in Figure 1.

4. Experimental Evaluation

4.1. Setup

4.1.1. Dataset

To validate our approaches, we experiment with three modalities of data: tabular, images, and natural language. All

Algorithm 2 Iterative Expansion

Input: the number of iterations n, the add_ratio p, the threshold t_{ratio}, the number of classes L, $\mathcal{D}^{src}$, $\mathcal{D}^{aux}$, $\mathcal{D}^{val}$
for $l = 1$ **to** L **do**
  $T_l = d(\mathcal{D}_l^{val}, \mathcal{D}_l^{src}) * t_{ratio}$
end for
for $i = 1$ **to** n **do**
  for $l = 1$ **to** L **do**
    $s = \textbf{randomSample}\left(\mathcal{D}_l^{aux}, size = p * |\mathcal{D}_l^{val}|\right)$
    if $d(\mathcal{D}_l^{val}, \mathcal{D}_l^{src}) - d(\mathcal{D}_l^{val} \cup s, \mathcal{D}_l^{src}) > T_l$ **then**
      $\mathcal{D}_l^{val} = \mathcal{D}_l^{val} \cup s$
      $\mathcal{D}_l^{aux} = \mathcal{D}_l^{aux} \setminus s$
    else
      continue
    end if
  end for
end for
return $\mathcal{D}^{val}$

datasets are on classification tasks. The statistics of all datasets are listed in Table 1

Tabular Data We adopt 6 publicly datasets from UCI [2] and Kaggle, which are also data sources applied in some machine learning works (Wang et al., 2020; Bi & Zhang, 2018; Moss et al., 2018). It is relatively easy to achieve stable and low bias evaluations with balanced and sufficient data via cross validation, which makes comparisons between methods less meaningful and persuasive.

Computer Vision The common and challenging long-tail distribution is chosen as our evaluation environment. And we follow previous works (Li et al., 2021; Park et al., 2021; Cui et al., 2019) to construct long-tailed versions of CIFAR10 with imbalance factor ρ = 100, named as CIFAR10-LT. Specifically, for a k-class dataset, we create a long-tailed dataset by reducing the number of examples per class according to the exponential function $n_i' = n_i \mu^i (\mu \in (0,1), i = 0, 1, ..., k)$, where n_i is the original number of examples for class i, while n_i' is the new number.

Natural Language Process We use the Reuters-21578 dataset (Dua & Graff, 2017) according to the exact setting in JKFold (Moss et al., 2018). More in detail, only corn and wheat categories are chosen.

4.1.2. Baseline

Model For tabular data, we use a decision-tree-based model xgboost (Chen & Guestrin, 2016) to perform classification tasks. While for CIFAR10-LT and REUTERS, we train ResNet-44 (He et al., 2016) and DistilBert (Sanh et al., 2019) as classifiers, respectively. Notice that we suppress randomness in the model by fixing random seeds and maintaining a consistent batch data feeding order so as to control unique variables as a dataset(train/val set).

[2] https://archive.ics.uci.edu/ml/datasets.php

Table 1. Dataset statistics. **ρ** denotes the imbalance factor, which is the ratio of the number of samples in the largest class to the smallest class. ***#C*** denotes the number of classes. ***#F*** denotes the number of feature dimensions.

Modality	Dataset	*#Data*	*#C*	ρ	*#F*
Table	*PageBlock*	5001	2	56	10
	CarEval-1	453	2	6	6
	CarEval-2	1279	2	18	6
	BankMarket	3047	2	7	13
	MushRoom	811	2	1	22
Image	*CIFAR10-LT*	14884	10	100	1024
Text	*REUTERS*	520	2	2	-

Method Three baselines are considered to compare against. First, holdout with a train-val split ratio of 8:2. Despite it being widely used, especially in deep learning owing to high efficiency, instability and deviation are its weak points due to the size of val-set being still relatively small (i.e., near a *few-shot* setup). Second, k-fold CV where a common choice 5 is taken for k according to Kuhn et al. (2013). The instability can be improved versus holdout. Third, repeated k-fold CV(named as J-K-Fold), the repeat times and k are set to 4 and 5 respectively following JKFold (Moss et al., 2018). Theoretically, it is a further enhancement of stability at the cost of time. In addition, we run baselines and EVD 100 times for each tabular dataset, while 10 times for the other datasets in consideration of time complexity issues.

4.1.3. Metrics

In order to make a comprehensive evaluation of validation methods, we propose to compare three metrics simultaneously, i.e., *variance, bias, and scores on the test set*. (i) For *variance*, we calculate the standard deviation of estimated scores (scores on val-set) overall runs. (ii)For *bias*, we use the mean of the absolute gap between scores on val-set and test-set overall runs, which is the same usage in previous work((Zeng & Martinez, 2000; Budka & Gabrys, 2012)). (iii) For *test score*, we use the mean of scores on the test set over all rounds. High test scores, small variance, and bias correspond to good validation and vice versa.

4.2. Main Results

We report our results in Table 2 respectively for tabular data and image together with text classification. We may conclude from the scores that: (i) the stability of evaluation is enhanced through EVD. The variance of EVD is

always lower than that of 5-fold and also lower than 4-5-Fold on some datasets. (ii) the estimated scores obtained from EVD are more accurate and therefore more reliable. The bias of EVD is considerably lower than baselines on most datasets, except for Reuters and Pageblock. (iii) the models obtained using EVD offer some performance gains on the test sets, indicating that our approach somewhat mitigates the variance-bias trade-off trouble.

Table 2. Tabular-data, image and textclassification results reported in variance. bias and test-f1.

DATA	METHOD	*Variance* ↓	*Bias* ↓	*Test-F1* ↑
PAGE-BLOCKS	*Holdout*	0.048	0.041	0.909
	5-Fold CV	0.010	0.037	0.909
	4-5-Fold CV	**0.0048**	**0.037**	**0.909**
	EVD(ours)	0.0090	0.060	0.909
CAR-EVAL1	*Holdout*	0.085	0.043	0.945
	5-Fold CV	0.054	0.052	0.931
	4-5-Fold CV	0.024	0.053	0.932
	EVD(ours)	**0.0086**	**0.0074**	**0.955**
CAR-EVAL2	*Holdout*	0.081	0.075	0.944
	5-Fold CV	0.033	0.076	0.944
	4-5-Fold CV	0.017	0.074	0.945
	EVD(ours)	**0.0040**	**0.025**	**0.974**
BANK-MARKET	*Holdout*	0.063	0.073	0.373
	5-Fold CV	0.017	0.079	0.375
	4-5-Fold CV	0.0087	0.079	0.373
	EVD(ours)	**0.013**	**0.020**	**0.424**
MSUH-ROOM	*Holdout*	0.011	0.0071	0.993
	5-Fold CV	0.0029	0.0080	0.994
	4-5-Fold CV	0.0015	0.0079	0.994
	EVD(ours)	**0.0013**	**0.0014**	**0.994**
CIFAR-10-LT	*Holdout*	0.028	0.028	0.658
	5-Fold CV	0.0055	0.031	0.658
	4-5-Fold CV	0.0050	0.033	0.656
	EVD(ours)	**0.0032**	**0.024**	**0.696**
REUTERS	*Holdout*	0.021	0.067	0.894
	5-Fold CV	0.0040	0.065	0.894
	4-5-Fold CV	0.0020	0.065	0.896
	EVD(ours)	**0.0021**	**0.066**	**0.909**

4.3. Parameters Tuning Case

Parameters tuning is one of the major applications of validation sets. Thereby we employ EVD based on kfold and JKFold to perform this task, where only the expansion operation is applied. Specifically, we utilize grid search to tune parameters. And we calculate performance estimation across a set of possible parameter values(or grid) and select the value that gives us the best-estimated performance. Following the same setting in JKFold(Moss et al., 2018), we tune two critical parameters(C and gamma in Sklearn) of support vector machines (SVM) on the Reuters dataset. The searching space for two parameters is {1, 5, 10, 50, 100, 500, 1000, 5000, 10000} and {0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45} respectively. As summarised in Table 3 and Fig 2, we see a substantial drop in gamma variability than C variability comparing with 5fold. And the estimated accuracy becomes stabler via EVD, which means EVD provides a more reliable distinction between the best parameters and the alternatives.

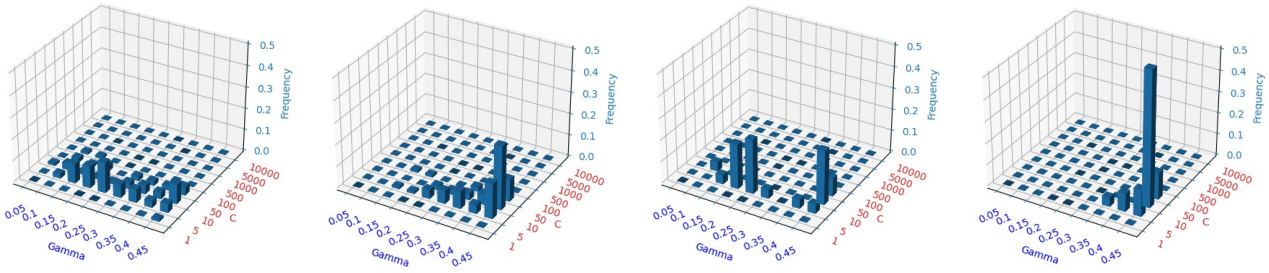
Table 3. Parameters tuning the performance of different validation methods for a support vector machine classifier on Reuters. (SD denotes standard deviation, and acc denotes accuracy.)

METHOD	*SD-C*↓	*SD-g*↓	*SD-Acc*↓
5-fold	410	0.015	4.30E-5
EX-5-fold	276	0.0060	3.24E-5
4-5-fold	155	0.013	9.83E-6
EX-4-5-fold	200	0.00058	6.41E-6

4.4. Ablation Study

In this section, we attempt to reveal the effectiveness of different components in EVD. We conduct a series of ablation studies on the following factors:

- **coreset-based initialization**. We compare coreset-based initialization with randomly selected initialization in Table 4. It is obvious that the biases are unanimously higher when the data are randomly selected from the *auxiliary dataset*, which also reflects the fact that our initialization is more representative of the whole set. Additionally, we conducted our experiments using the same training set as holdout rather than the whole training set(results are shown in Table 5). Comparing with holdout, even without the extra training data, the bias and variance of our method are still smaller in most cases when the validation set is the only different element. This also confirms that the improvement in evaluation quality comes not only from additional training data but also from the high-quality validation set itself.
- **iterative expansion**. We perform the expansion operation directly on the val-set partitioned by holdout or kfold. Table 6 summarizes the results. We have the following two observations. First and most importantly, using expansion can reduce the bias both of holdout and kfold. Second, as a minor point, we observe that it can depress the variance, yet the variance is still greater than that of jkfold. These phenomena illustrate the effects of expansion.



(a) 5-fold (b) EVD-5-fold(ours) (c) 4-5-fold (d) EVD-4-5-fold(ours)

Figure 2. Distribution of the tuned parameter values across 100 random partitions.

Table 4. Results of the ablation study part 1. See text in section 4.4 for details. The table shows the metrics that differ from the results of ours in Table 2. (*Diff* is short for *Difference*.)

METHOD	*Variance-Diff*	*Bias-Diff*	*F1-Diff*
PageBlocks	+0.012	+0.0026	+0.0001
CarEval1	+0.017	+0.013	-0.0005
CarEval2	+0.017	+0.0025	-0.013
BankMarket	+0.021	+0.017	+0.0005
MushRoom	+0.0023	+0.0016	-0.00040
CIFAR10-LT	+0.0034	+0.174	+0.010

Table 5. Results of the ablation study2. See text in section4.4 for details. The table shows the metrics that differ from the results of ours in Table 2 (*Diff* is short for *Difference*.)

METHOD	*Variance-Diff*	*Bias-Diff*	*F1-Diff*
PageBlocks	+0.034	-0.00060	+0.0001
CarEval1	-0.070	-0.031	+0.011
CarEval2	-0.058	-0.048	+0.016
BankMarket	-0.041	-0.028	+0.0094
MushRoom	-0.0078	-0.0035	+0.00020
CIFAR10-LT	-0.0090	-0.0080	+0.0090
Reuters	-0.0148	+0.013	+0.0030

5. Conclusion

In this article, we proposed a split-free framework, denoted as Expansile Validation Dataset (EVD), to resolve the train/validation tradeoff. It's a versatile method that can be combined seamlessly with other split-based validation mechanisms. Through extensive experiments on datasets in different fields, including six publicly datasets from UCI and Kaggle (tabular data), CIFAR10-LT (CV), and Reuters-21578 (NLP), we justify the validity of EVD from both performance and robustness perspectives.

Indeed, besides the excellent results yielded by EVD, we have left the theoretical understanding of this framework out of the scope of this paper. This is very much due to the absence of a theoretical basis for most of the data augmentation methods we exploited off-the-shelf — such as the family of the Mixup method. In the future, we plan to investigate more in this direction.

Table 6. Results of the ablation study 3. See text in section 4.4 for details. The table shows the metrics that differ from the results of baselines(ie., holdout and k-fold) in Table 2.(*Diff* is short for *Difference*).)

METHOD	*Variance-Diff*	*Bias-Diff*	*F1-Diff*
PageBlocks	-0.020	-0.0012	-0.0001
PageBlocks	-0.00090	-0.0013	-0.0001
CarEval1	-0.069	-0.027	+0.011
CarEval1	-0.043	-0.035	+0.023
CarEval2	-0.059	-0.049	+0.016
CarEval2	-0.020	-0.049	+0.016
BankMarket	-0.025	-0.038	+0.0086
BankMarket	-0.0018	-0.045	+0.0075
MushRoom	-0.0051	-0.0023	-0.00020
MushRoom	-0.00010	-0.0033	-0.00060
CIFAR10-LT	-0.0080	-0.0070	+0.0090
CIFAR10-LT	+0.00020	-0.0080	+0.0050
REUTERS	-0.0060	-0.063	+0.0020
REUTERS	-0.00010	-0.0050	-0.0010

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. PMLR, 2017.

Bi, J. and Zhang, C. An empirical comparison on state-of-the-art multi-class imbalance learning algorithms and a new diversified ensemble learning scheme. *Knowledge-Based Systems*, 158:81–93, 2018.

Budka, M. and Gabrys, B. Density-preserving sampling:

robust and efficient alternative to cross-validation for error estimation. *IEEE transactions on neural networks and learning systems*, 24(1):22–34, 2012.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Chen, J., Yang, Z., and Yang, D. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. *arXiv preprint arXiv:2004.12239*, 2020a.

Chen, P., Liu, S., Zhao, H., and Jia, J. Gridmask data augmentation. *arXiv preprint arXiv:2001.04086*, 2020b.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Cheng, Y., Jiang, L., Macherey, W., and Eisenstein, J. Advaug: Robust adversarial augmentation for neural machine translation. *arXiv preprint arXiv:2006.11834*, 2020.

Cui, Y., Jia, M., Lin, T.-Y., Song, Y., and Belongie, S. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9268–9277, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

DeVries, T. and Taylor, G. W. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Eaton-Rosen, Z., Bragman, F., Ourselin, S., and Cardoso, M. J. Improving data augmentation for medical image segmentation. 2018.

Feng, S. Y., Li, A. W., and Hoey, J. Keep calm and switch on! preserving sentiment and fluency in semantic text exchange. *arXiv preprint arXiv:1909.00088*, 2019.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Guo, H., Mao, Y., and Zhang, R. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jiang, G. and Wang, W. Error estimation based on variance analysis of k-fold cross-validation. *Pattern Recognition*, 69:94–106, 2017.

Jindal, A., Ranganatha, N. E., Didolkar, A., Chowdhury, A. G., Jin, D., Sawhney, R., and Shah, R. R. Speechmix-augmenting deep sound recognition using hidden space interpolations. In *INTERSPEECH*, pp. 861–865, 2020.

Joseph, V. R. and Vakayil, A. Split: An optimal method for data splitting. *Technometrics*, pp. 1–11, 2021.

Jung, Y. Multiple predicting k-fold cross-validation for model selection. *Journal of Nonparametric Statistics*, 30 (1):197–215, 2018.

Kohavi, R. et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pp. 1137–1145. Montreal, Canada, 1995.

Kuhn, M., Johnson, K., et al. *Applied predictive modeling*, volume 26. Springer, 2013.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Li, M., Zhang, X., Thrampoulidis, C., Chen, J., and Oymak, S. Autobalance: Optimized loss functions for imbalanced data. *Advances in Neural Information Processing Systems*, 34, 2021.

Li, P., Li, X., and Long, X. Fencemask: A data augmentation approach for pre-extracted image features. *arXiv preprint arXiv:2006.07877*, 2020a.

Li, W., Geng, C., and Chen, S. Leave zero out: Towards a no-cross-validation approach for model selection. *arXiv preprint arXiv:2012.13309*, 2020b.

Li, Y., Li, X., Yang, Y., and Dong, R. A diverse data augmentation strategy for low-resource neural machine translation. *Information*, 11(5):255, 2020c.

Moss, H. B., Leslie, D. S., and Rayson, P. Using jk fold cross validation to reduce variance when tuning nlp models. *arXiv preprint arXiv:1806.07139*, 2018.

Park, S., Lim, J., Jeon, Y., and Choi, J. Y. Influence-balanced loss for imbalanced visual classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 735–744, 2021.

Richardson, E. and Weiss, Y. On gans and gmms. *arXiv preprint arXiv:1805.12462*, 2018.

Şahin, G. G. and Steedman, M. Data augmentation via dependency tree morphing for low-resource languages. *arXiv preprint arXiv:1903.09460*, 2019.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Sener, O. and Savarese, S. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

Sennrich, R., Haddow, B., and Birch, A. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*, 2015.

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

Singh, K. K., Yu, H., Sarmasi, A., Pradeep, G., and Lee, Y. J. Hide-and-seek: A data augmentation technique for weakly-supervised localization and beyond. *arXiv preprint arXiv:1811.02545*, 2018.

Székely, G. J. and Rizzo, M. L. Energy statistics: A class of statistics based on distances. *Journal of statistical planning and inference*, 143(8):1249–1272, 2013.

Tiittanen, H., Holm, L., and Törönen, P. A new approach to multilabel stratified cross validation with application to large and sparse gene ontology datasets. *arXiv preprint arXiv:2109.01425*, 2021.

Wang, C., Deng, C., and Wang, S. Imbalance-xgboost: leveraging weighted and focal losses for binary label-imbalanced classification with xgboost. *Pattern Recognition Letters*, 136:190–197, 2020.

Wang, T., Zhu, J.-Y., Torralba, A., and Efros, A. A. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Wei, J. and Zou, K. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6023–6032, 2019.

Zeng, X. and Martinez, T. R. Distribution-balanced stratified cross-validation for accuracy estimation. *Journal of Experimental & Theoretical Artificial Intelligence*, 12(1): 1–12, 2000.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. Random erasing data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 13001–13008, 2020.

A. You *can* have an appendix here.

You can have as much text here as you want. The main body must be at most 8 pages long. For the final version, one more page can be added. If you want, you can use an appendix like this one, even using the one-column format.