Diagnosing Ensemble Few-Shot Classifiers

Weikai Yang, Xi Ye, Xingxing Zhang, Lanxi Xiao, Jiazhi Xia, Zhongyuan Wang, Jun Zhu, Hanspeter Pfister, and Shixia Liu, *Fellow, IEEE*

Abstract—The base learners and labeled samples (shots) in an ensemble few-shot classifier greatly affect the model performance. When the performance is not satisfactory, it is usually difficult to understand the underlying causes and make improvements. To tackle this issue, we propose a visual analysis method, FSLDiagnotor. Given a set of base learners and a collection of samples with a few shots, we consider two problems: 1) finding a subset of base learners that well predict the sample collections; and 2) replacing the low-quality shots with more representative ones to adequately represent the sample collections. We formulate both problems as sparse subset selection and develop two selection algorithms to recommend appropriate learners and shots, respectively. A matrix visualization and a scatterplot are combined to explain the recommended learners and shots in context and facilitate users in adjusting them. Based on the adjustment, the algorithm updates the recommendation results for another round of improvement. Two case studies are conducted to demonstrate that FSLDiagnotor helps build a few-shot classifier efficiently and increases the accuracy by 12% and 21%, respectively.

Index Terms—Few-shot learning, ensemble model, subset selection, matrix visualization, scatterplot

1 Introduction

The few-shot classification aims to train a classifier to recognize unseen classes with only a few labeled samples (shots) in each class, which is of great significance both academically and practically [1], [2]. For example, at the early stage of the COVID-19 epidemic, the massive labeling of the CT scans requires a long process of clinical observation with the risk to patients' lives. As such, few-shot classification is a viable choice for these scenarios. Many advances have been made to continuously improve the performance of few-shot classifiers by developing a variety of methods, such as ensemble learning, generative models, and meta-learning [2]. Because the ensemble few-shot classification can combine any few-shot classifiers (base learners) for better performance, it is the most widely used state-of-the-art method in practice. For example, three of the top five best-performing models in a CVPR challenge on few-shot learning [3] and four of the top five best-performing models in a Kaggle competition on few-shot learning [4] have used ensemble few-shot classifiers to boost performance successfully.

Previous studies have shown that the performance of the ensemble model is largely affected by the diversity and cooperation among the individual base learners [1] and the representativeness of the shots [5]. Accordingly, using all learners and shots may downgrade the performance. For example, if the performance of a learner is poor and its predictions are different from the majority, it will hurt the performance of the ensemble model. In addition, a shot wrongly representing some samples usually leads to the misclassification of these samples. Thus, it is desirable to select a subset of cooperative and diverse learners and identify a small set of representative shots, which is a long-standing challenge for the practical application of few-shot classification. Existing learning methods typically apply an ensemble model to all the given learners and shots [1], [6], which often fail to achieve the best performance. Improving the performance usually requires repeatedly selecting the learners and adjusting their weights. Without a comprehensive understanding of how the model and shots work together to reach the final predictions, this trial-and-error process is very time-consuming and expertise-demanding. Moreover, lacking the refinement of the shots, the performance improvement is limited [7], [8]. To improve the performance efficiently, users need an efficient way to analyze the performance-related log data ("*analyze first*"). The learners and shots with unusual behavior, such as the learner causing a large confidence drop or the shot with poor coverage, can be highlighted ("*show the important*"). After understanding the roles of learners/shots in the final predictions, they can then decide which ones to be added/removed for improving the performance ("*interaction and feedback*").

- *Weikai Yang and Shixia Liu are with School of Software, BNRist, Tsinghua University, Beijing 100084, China. E-mail: ywk19@mails.tsinghua.edu.cn, shixia@tsinghua.edu.cn.*
- *Xi Ye is with the University of Texas at Austin, Austin, TX 78712 USA. E-mail: xiye@cs.utexas.edu.*
- *Xingxing Zhang and Jun Zhu are with Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing 100084, China. E-mail: xxzhang2020@mail.tsinghua.edu.cn, dcszj@tsinghua.edu.cn.*
- *Lanxi Xiao is with Academy of Arts & Design, Tsinghua University, Beijing 100084, China. E-mail: xlq20@mails.tsinghua.edu.cn.*
- *Jiazhi Xia is with Central South University, Changsha, Hunan 410017, China. E-mail: xiajiazhi@csu.edu.cn.*
- *Zhongyuan Wang is with Kuaishou Technology Company Ltd., Beijing 100085, China. E-mail: wzhy@outlook.com.*
- *Hanspeter Pfister is with Harvard University, Cambridge, MA 02138 USA. E-mail: pfister@g.harvard.edu.*

Manuscript received 10 Nov. 2021; revised 8 June 2022; accepted 8 June 2022. Date of publication 13 June 2022; date of current version 1 Aug. 2022.
This work was supported in part by the National Key R&D Program of China under Grant 2020YFB2104100, in part by the National Natural Science Foundation of China under Grants U21A20469, 61936002, in part by Institute Guo Qiang, THUIBCS, and BLBCI, and in part by the Tsinghua-Kuaishou Institute of Future Media Data.
(Corresponding author: Shixia Liu.)
Recommended for acceptance by C. Wang.
Digital Object Identifier no. 10.1109/TVCG.2022.3182488

Based on the updated learners (shots), suitable shots (learners) are recommended for another round of analysis ("*analyze again*"). Such an iterative analysis process with human-in-the-loop fits well with the visual analytics mantra [9] and inspires us to develop a visual analysis tool, FSLDiagnotor, for tuning the selection of learners and shots.

The key behind FSLDiagnotor is its ability to efficiently identify and eliminate performance bottlenecks caused by the selected base learners and shots. Given a set of learners and a collection of samples with a few shots, we consider two problems: 1) finding a subset of diverse and cooperative learners that well predict the sample collections and 2) removing low-quality shots and recommending necessary new shots to adequately represent the sample collection. By studying the intrinsic characteristics of these two problems, we formulate them as sparse subset selection and develop two selection algorithms to recommend appropriate learners and shots. However, the recommendations are not always perfect and may contain one or a few low-quality learners/shots. For example, a learner that wrongly predicts some samples with high confidence can be recommended because it is mistaken as a well-performing learner for those samples. Such low-quality learners/shots are hard to be detected and corrected without human involvement. To facilitate such tasks, a matrix visualization and a scatterplot are combined to explain the prediction behavior of the recommended learners and the coverage of the shots in context. Based on the understanding of the behavior of the learners and shots, users can improve the selection of learners and enhance the shots for better performance.

We performed a quantitative evaluation to show that both the learner and shot selection algorithms can boost the performance of the few-shot classifier. We also conducted two case studies with two machine learning experts to demonstrate that our tool helps diagnose and improve the few-shot classifier more efficiently and increases the accuracy by 12% and 21%, respectively. The demo is available at http://fsldiagnotor.thuvis.org/.

The main contributions of this work include:

- The formulation of sparse subset selection that unifies the shot and learner selection into one framework.
- An enhanced matrix visualization coordinated with a scatterplot to explain how the base learners and shots contribute to the final predictions.
- A visual analysis pipeline that tightly integrates the subset selection algorithm with interactive visualization to facilitate the iterative improvement of the shots and base learners.

2 Related Work

2.1 Few-Shot Classification

The ensemble methods have been explored in the vein of few-shot classification to boost the performance [1], [6]. Dvornik *et al.* [1] encouraged the diversity and cooperation between learners for better performance. In addition to training the base learners, Qi *et al.* [6] adaptively assigned a weight to each learner for a strong few-shot classifier. While more and more sophisticated models have been developed, there is recent work pointing out the cruciality of high-quality features: using high-quality features is even more effective than employing a well-designed complex model [10]. Following such a philosophy, Dvornik *et al.* [11] learned high-quality feature extractors to extract high-quality features for unlabeled samples. Due to the importance of the diversity-cooperation strategy and the features, our work combines the two. We leveraged deep learning models, such as a pre-trained ResNet model [12], to extract the features for each sample. Then a set of learners were built based on the extracted features. This saves training time and provides the flexibility to quickly obtain the base learners. Our method also recommends a subset of base learners and enhances the quality of shots to further improve the performance.

2.2 Visual Analysis for Improving Model Performance

Existing visual analysis work for improving model performance can be classified into two categories: model-driven methods and data-driven methods [13], [14].

Model-driven methods facilitate experts to better understand the inner workings of a machine learning model and discover the reason why a training process fails to achieve an acceptable performance. For example, CNNVis [15] was developed to diagnose the potential issues of a convolutional neural network (CNN) by examining the learned features and activation of neurons. Alsallakh *et al.* [16] utilized a confusion matrix to disclose the impact of class hierarchy on the features learned at each CNN layer. Kahng [17] developed ACTIVIS to facilitate the identification of specific training issues on an industry-scale deep learning model by illustrating how neurons are activated by the instances of interest. Later efforts focus on diagnosing other types of models, such as deep generative models [18], Deep Q-Networks [19], and sequential models [20], [21]. In addition to improving a single model, some efforts focus on analyzing ensemble models [22], [23], [24], [25]. For example, Schneider *et al.* [23] developed a visual analysis tool to explore the data and model spaces of the ensemble model and improve its performance by enabling a selection of the base learners. Our method supports the improvement on both the data and model.

In the same spirit of data-centric AI [7], [8], data-driven methods aim to improve the quality of training samples at the instance and label levels. At the instance level, Chen *et al.* [26] developed OoDAnalyzer, a visual analysis tool to analyze the out-of-distribution samples in the context of the training and test samples. Yang *et al.* [27] proposed DriftVis to detect and correct the distribution changes in a data stream. Ming *et al.* [28] developed ProtoSteer to explain the prediction of an input sample by using exemplary samples that have similar scores to this sample. Model developers can improve the model performance by revising the exemplary samples. More recently, Gou *et al.* [29] proposed to generate unseen test cases to improve model robustness. At the label level, Heimerl *et al.* [30] utilized active learning to facilitate the task of interactive labeling for document classification. This idea of employing active learning to support interactive labeling has also been adopted by other visual analysis work [31], [32], [33], [34]. Most of the later research along this line has focused on detecting and correcting noisy

labels in training samples. Liu *et al.* [5] introduced LabelInspect to improve the crowdsourced annotations by utilizing the mutual reinforcement relationships between the workers' behavior and the uncertainty of the annotated results. Xiang *et al.* [35] developed a visual analysis tool to correct label errors in a large set of training samples based on user-selected trust items. More recently, Jia *et al.* [36] applied active learning to zero-shot classification. They interactively built a class attribute matrix for improving the performance of classifiers.

Although the aforementioned methods have shown the capability of improving the model performance to some extent, there are few efforts that tightly combine model-driven methods with data-driven methods to improve performance. The combination is particularly needed in few-shot learning since both the data and model greatly influence the performance. Thus, we develop FSLDiagnotor to improve both shots and learners.

3 Background

Few-shot classification aims to learn a good classifier for unseen classes with a few shots. Specifically, for the N samples from these unseen classes, only the labels of M shots (e.g., 1–5 shots per class) are provided. The shot set is denoted as $S = \{(x_j^*, y_j^*)\}_{j=1}^{M}$. Here, y_j^* is the label of shot x_j^*. It is represented by a one-hot vector where the value of the corresponding class index is 1, and the others are 0s. The goal is to build a model to predict the label distribution y of a sample x of the unseen classes based on S. The label distribution y is a C-dimensional probability vector. C is the number of classes, and the value of the i-th dimension indicates the probability of the sample belonging to the i-th class.

Ensemble few-shot classification combines a set of base learners $\{\theta_k\}_{k=1}^{K}$ for achieving better performance. Fig. 1 illustrates the process of predicting the label distribution of a sample based on three given shots and two learners. For sample x, each learner θ_k generates a label distribution y_{θ_k}. These label distributions are then averaged with weight w_k to obtain the final label distribution y:

$$y = \frac{1}{K}\sum_{k=1}^{K} w_k y_{\theta_k}. \tag{1}$$

w_k is set to 1 by default and can be adjusted in our tool. It can be seen that the prediction results of the ensemble model are determined by the base learners and shots. Thus users require a tool to help them examine the quality of base learners and shots and tune them for better performance.

4 Design of FSLDiagnotor

4.1 Requirement Analysis

We collaborated with three machine learning experts (E1, E2, E3) to design FSLDiagnotor. E1 is a postdoc researcher with an interest in data selection and few-shot learning. E2 and E3 are two Ph.D. students with a focus on few-shot learning. They are not the co-authors of this work. The following three requirements are identified based on existing literature and three 60-minute participatory design sessions with the experts.

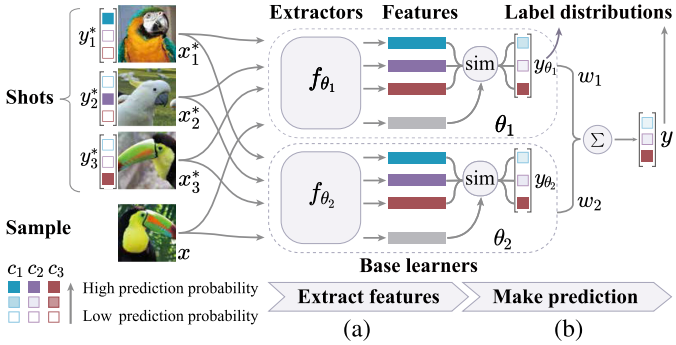


Fig. 1. The prediction process of the ensemble few-shot classifier: (a) each base learner extracts the features of the shots and samples; (b) label distributions of the samples ($y_{\theta_1}, y_{\theta_2}$) are calculated based on the similarity between the features and then averaged with weights to obtain the final label distribution y.

R1: Tuning the Selection of Learners and Their Ensemble Weights. Previous work has indicated that the diversity and cooperation among the base learners are very important for improving the performance of the ensemble model [1]. The experts also raised concerns regarding the current trial-and-error process for tuning the model when the accuracy is not acceptable. They usually need to repeatedly examine the log data to understand the diversity and cooperation among learners, and manually adjust their selection and ensemble weights. This is very time-consuming. To facilitate the tuning process, the experts expressed the need to quickly understand the prediction behavior of base learners on different levels, including the overall difference compared with the ensemble model and the detailed difference on different classes.

R2: Improving the Quality of the Shots. The representativeness of the shots is essential for few-shot classification [5]. As there are only a few labeled samples, mislabeled or confusing shots, such as the overlapped ones between two categories, decrease the model performance greatly. Removing such low-quality shots and adding necessary new ones improve the coverage of the shots and overall performance. When diagnosing an ensemble few-shot classifier, the experts need to understand the coverage of each shot and find the samples that are not well covered by the shots. In addition, the experts required a tool that can automatically recommend low-quality shots to be removed and candidate samples to be added to the shot set, so that they can only examine a small subset and then quickly decide which ones to remove/add.

R3: Being Agnostic to the Model Architectures of Learners. Existing methods for ensemble few-shot classification build the base learners based on a given model architecture [1], [6]. This is not flexible as a fixed model architecture cannot satisfy the performance requirements of different applications. Thus, the experts need the flexibility to choose an appropriate architecture for a given task. To directly employ different model architectures, such as a pre-trained ResNet model [12] or a newly developed few-shot learning model, the ensemble model should be agnostic to the model architectures that are used to build the base learners.

4.2 System Overview

Motivated by the requirements, we have developed FSLDiagnotor to interactively select high-quality base

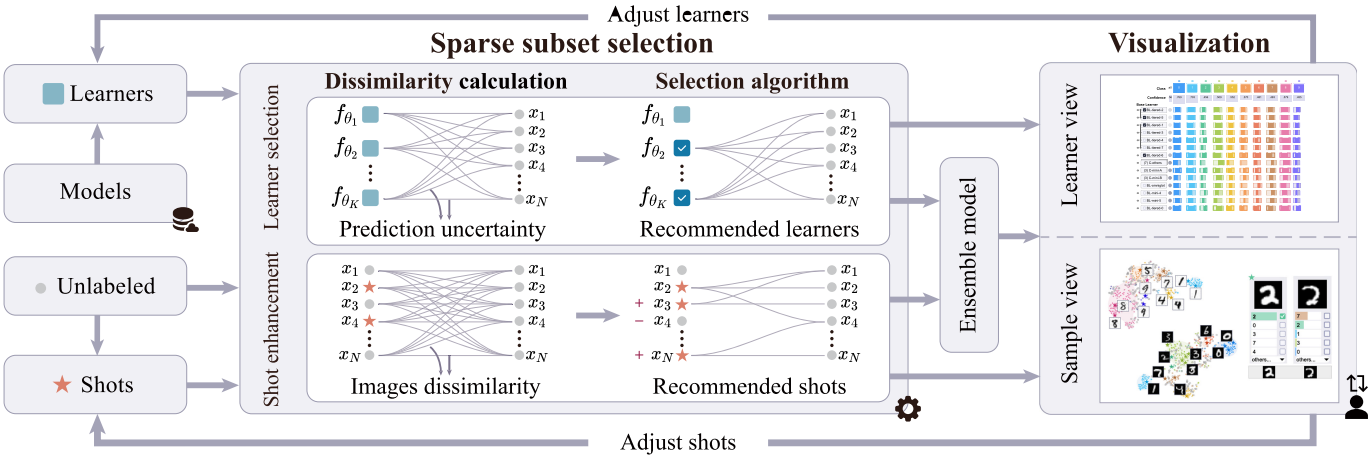


Fig. 2. FSLDiagnotor overview. Given the base learners and samples with a few shots, the sparse subset selection module recommends base learners and shots for building the ensemble few-shot classifier. The visualization module then explains how the learners and shots affect the final predictions, which facilitates users to improve them for interactively tuning the model.

learners and shots. As shown in Fig. 2, it consists of two modules: sparse subset selection and visualization. Given a set of base learners, shots, and unlabeled samples, the *sparse subset selection* module automatically recommends a subset of learners and a few shots. With these recommendations, an ensemble few-shot classifier is built. Next, the matrix visualization in the *visualization* module illustrates the performance of the learners and helps adjust their ensemble weights adaptively to improve the performance (**R1**). Users can also examine the coverage of the shots in the scatterplot and replace the low-quality shots with the high-quality ones (**R2**). The two modules work together to support an iterative tuning process until the desired performance is achieved. During the process, users can directly adjust the selection of the base learners without considering their model architectures (**R3**). This is achieved by building them directly on the features extracted by these models. As such, the ensemble model focuses only on feature-level integration. With this characteristic, users can directly use pre-trained models and newly developed few-shot models to extract features. This saves the training time and facilitates building the ensemble model flexibly.

5 Sparse Subset Selection

To build a high-quality few-shot classifier, FSLDiagnotor supports two tasks: 1) selecting a subset of diverse and cooperative base learners; 2) enhancing the representativeness of shots by replacing the low-quality ones with the high-quality ones. Because both tasks aim to find a small representative subset from a large data collection, we formulate them as distance-based sparse subset selection [37]. In this section, we first give an overview of the subset selection algorithm, then present how it can be extended to base learner selection and shot enhancement with task-related distances, and finally give the time complexity analysis. The quantitative result is shown in Section 7.1.

5.1 Algorithm Overview

Fig. 3 illustrates the basic idea of the algorithm. Given two sets $U = \{u_i\}_{i=1}^{I}$ and $V = \{v_j\}_{j=1}^{J}$ (U and V can be identical or different), the sparse subset selection algorithm aims to find a subset of U that can well represent set V. This is achieved by minimizing the following function that balances the representation quality and the size of the subset:

$$\sum_{j=1}^{J}\sum_{i=1}^{I} z_{ij} d_{ij} + \alpha \sum_{i=1}^{I} \max_j z_{ij} \quad \text{s.t. } z_{ij} \in \{0,1\},\ \forall i,j;\ \sum_{i=1}^{I} z_{ij} = 1,\ \forall j. \tag{2}$$

The first term is the cost of representing V with U (representation cost), and the second term is the sparsity term to penalize a large subset. In the first term, z_{ij} is a binary variable indicating whether v_j is represented by u_i, d_{ij} is the distance between u_i and v_j, and the constraint $\sum_{i=1}^{I} z_{ij} = 1$ guarantees that v_j is represented by only an element in U. In the second term, $\max_j z_{ij} = 1$ if u_i is selected in the subset, and $\sum_{i=1}^{I} \max_j z_{ij}$ is the size of the subset. $\alpha \geq 0$ controls the trade-off between the two terms.

The proposed formulation is NP-hard [38]. To solve it efficiently, we relax the discrete 0-1 integer $z_{ij} \in \{0,1\}$ to $z_{ij} \geq 0$ and convert the sparse subset selection into a continuous optimization problem. As in Elhamifar *et al.* [37], we adopt the alternating direction method of multipliers framework to optimize Eq. (2).

5.2 Base Learner Selection

Base learner selection aims to find a small subset of diverse and cooperative base learners to better predict the input samples (fitness). Here, U refers to the set of base learners $\{\theta_k\}_{k=1}^{K}$, and V is the set of samples $\{x_i\}_{i=1}^{N}$. As sparse subset selection encourages diversity among the selected learners,

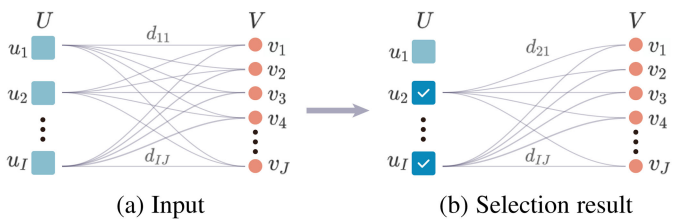


Fig. 3. An example of sparse subset selection.

we then extend it by considering fitness and cooperation. Accordingly, Eq. (2) is rewritten as:

$$\sum_{i=1}^{N}\sum_{k=1}^{K} z_{ki}d_{ki} + \alpha_1 \sum_{k=1}^{K} \lambda_k \max_i z_{ki} + \alpha_2 \sum_{1\le k<l\le K} \mu_{kl} \max_i z_{ki} \cdot \max_i z_{li}$$
$$\text{s.t.} \quad z_{ki} \ge 0,\ \forall k,i;\ \sum_{k=1}^{K} z_{ki} = 1,\ \forall i, \tag{3}$$

where the first term is the representation cost, the second term is the sparsity term that prefers the learners with higher fitness, and the third term is the cooperation term. α_1 and α_2 control the trade-off among the three terms. Following Elhamifar *et al*. [37], $\alpha_1 = \alpha_2 = 0.5\alpha_{\max}$, $\alpha_{\max}$ is the maximum distance between learners.

In the first term, to calculate the representation cost, we need to define the distance between a base learner and a sample. A straightforward way is based on the prediction accuracy. However, we cannot evaluate the accuracy without ground-truth labels. Instead, we use the prediction confidence to measure the distance because samples with high prediction confidence tend to be classified correctly [39]. The prediction confidence of learner θ_k on x_i is defined as the difference between the largest and the second-largest probabilities in the predicted label distribution y_i, which is denoted as $m_{ki} \in [0, 1]$. The distance between the learner θ_k and the sample x_i is then defined by $d_{ki} = 1 - m_{ki}$ because we prefer the base learners with larger confidence m_{ki}.

In the second term, to encourage the selection of base learners with higher fitness, we emphasize the ones that better predict the given shots. A widely used measure, likelihood, is employed to estimate the fitness value. Accordingly, we add λ_k for each learner θ_k, which is defined as its negative log-likelihood on the shots.

In the third term, to encourage the cooperation between two learners, θ_k and θ_l, we penalize the difference between their predictions. Let y_{ki} and y_{li} be the label distribution of sample x_i predicted by θ_k and θ_l, respectively. Following the previous work of Dvornik *et al*. [1], the prediction difference is measured by the symmetric KL-divergence between their predictions: $\mu_{kl} = \sum_{i=1}^{N}(\mathrm{KL}(y_{ki}||y_{li}) + \mathrm{KL}(y_{li}||y_{ki}))/(2N)$. μ_{kl} is 0 if the two learners make the same predictions.

5.3 Shot Selection

Shot selection aims to find a very small set of shots that better represents all the samples. Here, both U and V refer to the sample set $\{x_i\}_{i=1}^{N}$. Rather than treating the samples equally in the sparsity term of Eq. (2), we tend to select the low-confidence samples with higher representativeness since selecting them as shots can help the model distinguish more low-confidence samples [40]. Moreover, we try to preserve the given shots to reduce the analysis burden and labeling efforts. Accordingly, Eq. (2) is rewritten as:

$$\sum_{j=1}^{N}\sum_{i=1}^{N} z_{ij}d_{ij} + \alpha \sum_{i=1}^{N} \beta_i \gamma_i \max_j z_{ij}$$
$$\text{s.t.} \quad z_{ij} \ge 0,\ \forall i,j;\ \sum_{i=1}^{N} z_{ij} = 1,\ \forall j, \tag{4}$$

where the first term is the representation cost of the shots, and the second term is the sparsity term with preference on the previous shots. α controls the number of recommended shots. If we want to recommend N_s shots, we then set $\alpha = \alpha_{\max}/N_s$, where $\alpha_{\max}$ is the maximum distance between samples.

In the first term, the distance between samples x_i and x_j is calculated by averaging the cosine distances between their features extracted by the selected base learners.

In the second term, to encourage the selection of the low-confidence samples and given shots, we add a confidence coefficient γ_i and a stability coefficient β_i for x_i. The confidence coefficient favors the selection of low-confidence samples with higher representativeness. Accordingly, γ_i is set to its average prediction confidence of the selected learners. A sample with lower confidence results in a lower penalty in the sparsity term and then tends to be selected. The stability coefficient aims to preserve the given high-quality shots. Accordingly, β_i is set to 0.1 if x_i is a given shot. Otherwise, β_i is set to 1.

5.4 Time Complexity Analysis

The time complexity of sparse subset selection is $O(|U||V|)$ [37]. As the number of learners is not large, the running time of the learner selection is usually acceptable. However, the number of samples is relatively large, and thus, the shot selection algorithm is rather slow in computation. For example, it takes around 7 seconds to recommend shots from 1,000 samples. To tackle this issue, we first randomly sample a subset of samples and then recommend learners and shots based on the subset. The effectiveness of this sampling strategy is evaluated in Section 7.1.3.

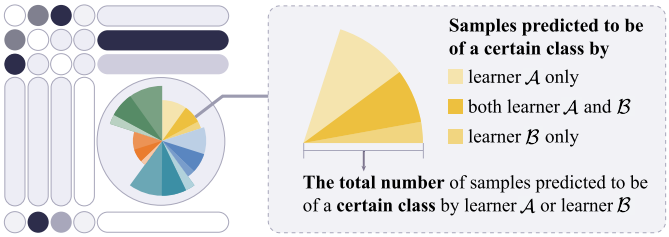
6 FSLDiagnotor Visualization

Although the sparse subset selection algorithm recommends a set of base learners and a few high-quality shots, the automatic recommendation results are not always perfect. For example, using likelihood to measure the quality of base learners is sometimes not accurate since the number of shots is very limited. In addition, an ambiguous shot wrongly representing some samples usually leads to more misclassification and thus the low representativeness of the shots. To better explain the recommendation results and facilitate the interactive tuning of the recommended learners and shots, we design a visualization-based explanatory environment. It consists of two components: 1) a learner view (Fig. 9a) to compare each base learner with the ensemble model in terms of prediction behavior (**R1**); and 2) a sample view (Fig. 9b) to present the shots and unlabeled samples in context (**R2**). The two coordinated views enable users to easily adjust the shots and the learners without considering the architectures of the learners (**R3**).

6.1 Learner View

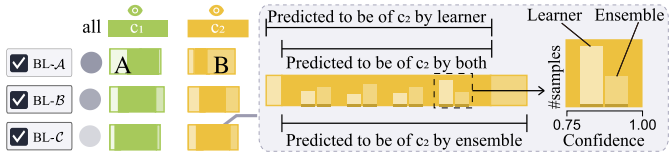
Due to the familiarity of users with the matrix visualization and its intuitiveness [41], we employ it to compare a base learner with the ensemble model and different learners (Fig. 9a). Users can tune the selection of learners or adjust their ensemble weights based on the comparative analysis.

Visual Design. Our first design focuses on the pairwise comparison between base learners, including the agreements and differences between the predictions of two learners. We design a matrix with zoomable cells (Fig. 4a) to present the pairwise comparison results where rows and



(a) Matrix with zoomable cells (b) Encoding

Fig. 4. The alternative design of the learner view. Both rows and columns represent base learners. A darker cell indicates a larger prediction difference between the two learners.



(a) Matrix with zoomable cells (b) Encoding

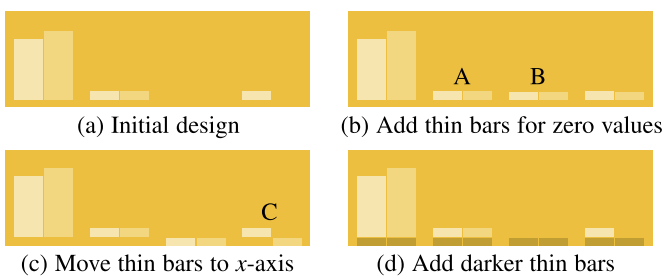
Fig. 5. The design of the learner view. Rows represent base learners, columns represent classes, and cells disclose the agreements and differences between the predictions of the base learners and ensemble model.

columns represent learners. A sequential color scheme from white to black is used to encode the total number of samples that are predicted differently by the two learners. Users can click on a cell of interest and zoom into it for the details of prediction behavior, which is depicted by a coxcomb chart. In this chart, each sector represents a class that samples are predicted to be of. A sector consists of three clockwise subsectors in the same hue (Fig. 4b), which represents the samples predicted to be of the same class by learner $\mathcal{A}$ only, by both learners $\mathcal{A}$ and $\mathcal{B}$, and by learner $\mathcal{B}$ only, respectively, where $\mathcal{A}$ is represented by the row, and $\mathcal{B}$ is represented by the column. The total number of samples that are predicted to be of this class is encoded by the radius of the sector. The experts agree that the comparison between two learners is helpful. They like the design of three sub-sectors that illustrate the agreement and difference between two learners. However, they are more interested in comparing a base learner with the ensemble model instead of comparing two learners (*issue 1*). The pairwise comparison fails to explain the role of a base learner in the ensemble predictions. Another concern is that this design does not support the comparison across different base learners on a specific class (*issue 2*), which is important for diagnosis.

To tackle these issues, we augment the matrix visualization to emphasize the comparison between the base learners and the ensemble model (*issue 1*) and enable class-level comparison (*issue 2*). In the matrix visualization (Fig. 5a), each row represents a base learner. The first column encodes the number of samples predicted differently between a base learner and the ensemble model (*issue 1*) with a sequential color scheme. The darker the cell is, the larger the difference is. The remaining columns present the comparison between a base learner and the ensemble model (*issue 1*) in terms of each class (*issue 2*). Instead of using the coxcomb chart in the first design, we employ a common visual metaphor, the stacked bar, to represent the agreement and difference between the predictions. As shown in Fig. 5b, the length of the stacked bar encodes the total number of the samples predicted to be of a certain class by the base learner and/or the ensemble model. The hue of the stacked bar encodes the class. As the experts are more familiar with the stacked bars, they can quickly identify the differences between each learner and the ensemble model under different classes. Fig. 5 A is an example where base learner "BL-$\mathcal{A}$" mostly agrees with the predictions made by the ensemble model on class "c_1." However, there are many samples that are only recognized by "BL-$\mathcal{A}$." As a result, the first bar is much longer than the third bar. This indicates that "BL-$\mathcal{A}$" *over-predicts* on class "c_1." Similarly, we find that "BL-$\mathcal{A}$" *under-predicts* on class "c_2" (Fig. 5 B).

The experts give positive feedback to the new design during our interviews. Later, two experts express the need to investigate the prediction confidence of the samples. After a thorough discussion, we use a histogram to convey the number of samples that are predicted by the learner/ensemble model with four different confidence bins (Fig. 5b). However, if one bar is not shown in a confidence bin due to the zero value, it is inconvenient for users to identify which one is not displayed (Fig. 6a). A straightforward solution is to preserve a minimum height for each bar (Fig. 6b). However, such a thin bar (Fig. 6 B) is difficult to be distinguished from other bars with very small values (Fig. 6 A). Another option is to place the thin bar on the x-axis to avoid such misunderstanding (Fig. 6c). After using it, the experts point out that it may be misunderstood as a negative value (Fig. 6 C). To tackle this issue, we add a default thin darker bar for each item on the x-axis (Fig. 6d).

Visualization Scalability. Although the matrix visualization helps users efficiently examine the predictions of learners, it suffers the scalability issue when the number of learners/classes increases. To tackle this, we cluster similar learners (or classes) using agglomerative clustering [42]. The key of the clustering method is to calculate the distance between learners (or classes). The distance between learners is measured by the symmetric KL-divergence of their predictions. The distance between classes is calculated as the Euclidean distance in the feature space. Since each class can be characterized by its shots, one common way to represent the class is by averaging the shot features (shot-based feature). However, it can be inaccurate due to the scarcity of shots. To compensate for this, we consider the word embedding of the class label (label-based feature), which is extracted by GloVe [43], a widely used word embedding model. We then obtain a more robust feature



(a) Initial design (b) Add thin bars for zero values
(c) Move thin bars to x-axis (d) Add darker thin bars

Fig. 6. Four designs for comparing the prediction confidence of samples: (a)-(c) alternative designs; (d) our design.

Fig. 7. The coverage of two shots: (a) a high-quality shot with many similar samples; (b) a low-quality shot with few similar samples.

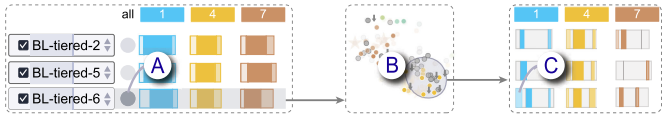


Fig. 8. "BL-tiered6" causes a confidence drop in B, and C shows some samples in B are only predicted to be of "1" by "BL-tiered6."

representation by concatenating the shot- and label-based features. Several interactions are provided to explore the clusters. For example, users can expand a cluster by double-clicking the associated rectangle and adjust the clustering result by dragging-and-dropping the rectangles. The clusters of less interest can be hidden to minimize distraction by clicking .

6.2 Sample View

Visual Design. The sample view (Fig. 9b) enables users to examine the shots in the context of samples and tune their selection. For each sample, we first concatenate the features extracted by the base learners. Next, to achieve better class separation [44], we employ t-SNE to project the samples onto 2D space and utilize a scatterplot to visualize the projections. In the scatterplot, stars and circles are used to represent shots and unlabeled samples, respectively. Samples are colored according to their classes, and those with a confidence less than 0.2 are colored *gray*. For each shot, we utilize a clutter-aware label-layout algorithm [45] to place the image content close to the shot and reduce the overlap with other scatter points. When users select the samples of interest, the image content and label distributions are displayed at the bottom of the view (Fig. 9b). The label distributions are represented by colored bars, where the color encodes the class, and the length encodes the prediction probability. Users can click the checkbox on the right side to add it as a shot or remove it from the shot set.

The sample view also illustrates the influence of the base learners and shots in the ensemble model. The influence of a learner is measured by the prediction confidence change of the ensemble model with/without the learner. If the confidence of a sample increases by 0.2 or more after adding the selected learner, the sample will be automatically marked with an upward arrow ↑. If the confidence decreases by 0.2 or more, the sample will be marked with a downward arrow ↓. We also use a gray density map as a guidance to highlight the regions where a larger increase/drop in confidence happens (Fig. 8 B). Such regions indicate the conflicted predictions between the selected learner and ensemble model and need to be further checked. The influence of a shot is characterized by its coverage, which contains its associated unlabeled samples with high similarity. The associated unlabeled samples with higher similarity are encoded by darker class colors. Fig. 7 shows the coverage of two shots. The first one is a high-quality shot of digit "1" since it influences a large number of neighboring samples that are correctly predicted with high confidence (Fig. 7a). In contrast, the second one is a low-quality shot of digit "3" because it only covers a few samples predicted with low confidence (Fig. 7b).

Visualization Scalability. The scatterplot inevitably suffers from the scalability issue when the number of samples grows [46]. To address this issue, we build a hierarchy by utilizing the random sampling strategy in a bottom-up manner [35]. Random sampling is employed because it can well preserve the overall data distribution [47]. When navigating the hierarchy, the sampled data at the current level are visualized using scatter points, and the others using a density map.

6.3 Incremental Improvement of Learners/Shots

To facilitate the diagnosis of the ensemble few-shot classifier, FSLDiagnotor provides a few interactions to assist in 1) improving the selection of base learners; 2) adjusting the ensemble weights of base learners; 3) enhancing the quality of shots; 4) mutually improving the learners and shots if either of them is adjusted. Here, recommendation-related interactions (e.g., recommend shots) and the weight adjustment are examples of semantic interactions [48], which enable smooth communication between the user and the analytical model without direct manipulation of the model.

Improving the Selection of Base Learners. FSLDiagnotor allows to remove low-quality learners and add high-quality ones. To decide which one is of low/high quality, we allow users to 1) explore the influence of the learners on the ensemble model to identify the key samples that are predicted differently by them; and then 2) examine the prediction difference between the learners and the ensemble model on these samples. For example, Fig. 8 shows that there is a larger difference between "BL-tiered6" and the ensemble model (Fig. 8 A). Users can click "BL-tiered6" to examine its influence and find that it causes a large confidence drop in a region (Fig. 8 B). After selecting samples in this region using the lasso, these samples are highlighted on the associated bars with a solid filling style . From these bars, it can be seen that some samples are only predicted to be of "1" by "BL-tiered6" (Fig. 8 C). By clicking the associated bar (Fig. 8 C), these samples are highlighted in the sample view for further examination. If the selected learner makes many wrong predictions on these samples, users can remove it.

Adjusting the Ensemble Weights of Base Learners. The ensemble weight is important for the model performance. Although automatic weight adjustment is an efficient way to achieve this, it requires some extra validation samples with labels [49]. Since these validation samples are not available in few-shot applications, FSLDiagnotor supports a semi-automatic adjustment of the ensemble weight of a learner to emphasize/de-emphasize it. For example, after examining a set of selected samples (S_1) that are predicted differently by the selected learner and the ensemble model, users can click ▲ to increase its weight if its predictions are mostly correct, or click ▼ to decrease its weight otherwise. Since the exact weight is hard to decide, our tool

automatically calculates the weight based on the prediction behavior of this learner and the ensemble model. Specifically, $\forall x_j \in S_1$, the final prediction of the ensemble model y_j should be consistent with y'_j, i.e., 1) the prediction of the learner if users increase its weight or 2) the prediction of the ensemble model without the learner if users decrease its weight. Moreover, $\forall x_j \in S_2$, where S_2 is the set of unselected samples, the final prediction y_j should be as same as possible to the previous prediction y_j^{prev}. Based on the two considerations, the weight is decided by maximizing:

$$\frac{\sum_{x_j \in S_1} \mathbb{I}(y_j = y'_j)}{|S_1|} + \frac{\sum_{x_j \in S_2} \mathbb{I}(y_j = y_j^{\text{prev}})}{|S_2|}. \quad (5)$$

The first term and the second term measure the prediction consistency on S_1 and S_2, respectively. $\mathbb{I}(\cdot)$ is the indicator function. It equals 1 if the prediction is consistent, and 0 otherwise. This optimization problem is solved by a grid search.

Steering the Selection of Shots to Enhance the Quality. FSLDiagnotor allows users to interactively enhance the quality of shots by removing the low-quality ones and adding necessary new ones in a steerable way. For example, users can identify the regions lacking shots and then label some of them. As shown in Fig. 10 B, digits "0" (blue) are mostly misclassified to be of "8" (pink) since there are no shots of digit "0" in this region. To improve the shot coverage in this region, users can manually add a few shots of "0" or click "Recommend Shot" to ask the tool to automatically recommend the candidate shots. In addition, if one class is predicted with low confidence, users can examine the associated samples to figure out the potential reason. Accordingly, users can click the bars in the matrix cell to examine the associated samples in the sample view.

Mutually Tuning the Learners and Shots. In ensemble few-shot classification, learners and shots work together for the final predictions. Generally, the change of learners influences the coverage of shots and vice versa. Thus, if the learner set or the ensemble weights are changed, the shots should also be updated to adapt to the corresponding change. To this end, users click "Recommend Shot." Then the shot selection algorithm is used to automatically recommend the shots. On the other hand, if the shots are changed, users can click "Recommend Learner" to obtain a better combination of learners by the learner selection algorithm. Such a process of mutual refinement saves users' time and efforts.

7 Evaluation

We conducted three experiments to evaluate the effectiveness of our subset selection algorithm. We also demonstrated the usability of FSLDiagnotor through two case studies. In the evaluation, we used the datasets with ground-truth labels to simulate the labeling process of users and calculate the accuracy.

7.1 Quantitative Evaluation on Subset Selection

7.1.1 Datasets and Setups

Datasets. We evaluated the learner and shot selection algorithms with four widely used datasets: *mini*-ImageNet [50], *tiered*-ImageNet [51], MNIST [52], and CIFAR-FS [53]. *Mini*-ImageNet consists of 80 seen classes and 20 unseen classes, each of which contains 600 images. *Tiered*-ImageNet contains 779,165 images of 608 classes (448 seen and 160 unseen classes). The seen classes of these two datasets were used for training base learners, while the unseen classes were used to evaluate the performance of the model. MNIST has 20,000 images of 10 unseen classes, and the images are augmented by inverting color. CIFAR-FS has 12,000 images of 20 unseen classes.

Base Learners. We used 24 base learners in the ensemble model. Sixteen of them are trained from scratch using ResNet-12 backbone [12], where 8 of them are trained on different subsets of the seen classes of the *mini*-ImageNet dataset, and the other 8 are trained on those of the *tiered*-ImageNet dataset. The remaining 8 base learners are pretrained on external datasets, e.g., natural images in ImageNet [54], handwritten characters in Omniglot [55]. We directly used the model parameters taken from publicly available implementations provided by Dvornik *et al.* [11].

Evaluation Criteria. We evaluated the performance in terms of classification accuracy, which is averaged over 100 trials.

7.1.2 Effectiveness Evaluation of Sparse Subset Selection

In this experiment, we evaluated whether the learner and shot selection algorithms can boost the few-shot classification accuracy on four datasets. Due to the limited number of shots, the randomness of few-shot classification is relatively high. To reduce the effect of such randomness, more trials are needed [10]. To perform the evaluation efficiently, we used less samples for each class by following the common practice in few-shot learning [56]. In particular, for *mini*-ImageNet and *tiered*-ImageNet, each task is a 5-class classification containing 5 randomly selected unseen classes, and each class contains 5 shots. For MNIST and CIFAR-FS, we used all the unseen classes (10 and 20, respectively) in the tasks. To simulate real-world applications, we do not guarantee that each class has the same number of shots. Instead, we randomly select 30 and 60 samples as shots (each class has 3 shots on average) from these two datasets, respectively. Each class of the four datasets contains 15 unlabeled samples. The *baseline* is obtained by using all the base learners and initial random shots in the ensemble model. Our method employs both the recommended base learners and recommended shots. For a fair comparison, the number of the recommended shots is set to be the same as that of the initial shots. The average number of the recommended learners over 100 trials is shown in Table 1. We compared our method, two ablations that only use either recommended learners (Rec. Learners) or shots (Rec. Shots), the state-of-the-art method, TIM [56], and the baseline.

As shown in Table 1, using either recommended learners or shots alone can boost the performance on all datasets, and combining them together can further improve the performance. By comparing the recommended learners/shots with the initial ones, we found that the low-quality learners/shots, such as a learner that has poor performance and predicts differently from the majority, were removed. Some

TABLE 1
Classification Accuracy on Four Datasets

Model	*mini*	*tiered*	MNIST	CIFAR-FS
Baseline	0.873	0.849	0.476	0.447
TIM [56]	0.874	0.898	-	-
Rec. Shots	0.877	0.862	0.611	0.517
Rec. Learners	0.880 (3.9)	0.868 (3.6)	0.481 (4.3)	0.480 (5.2)
Our method	**0.896** (3.9)	**0.908** (3.6)	**0.615** (4.3)	**0.541** (5.2)

The average numbers of recommended learners are given in parentheses.

high-quality learners/shots, such as a shot that well represents the unlabeled samples but does not appear in initial shots, were added. This is the main reason why the developed subset selection algorithms can boost the performance.

7.1.3 Balance Between Effectiveness and Efficiency

In our implementation, the random sampling strategy is employed to reduce the time cost for tasks with tens of thousands of samples or more. Here, we conducted this experiment to 1) investigate whether this sampling strategy can reduce the time cost while achieving comparable performance to that of running the algorithm on all the samples, and 2) determine the smallest sampling ratio needed to meet this requirement. We adopted different sampling ratios (1%, 2%, 3%, 4%, 5%, 6%, 7%, 10%, 100%) for recommending learners/shots, and calculated the accuracy on all the samples except shots.

Table 2 shows that the accuracy increases with the number of samples when the sampling ratio is lower than 5%. However, when the sampling ratio is greater than 5%, the pace of the increase begins to slow down. Based on this observation, we drew the conclusion that using a small subset of samples can achieve comparable accuracy to that of using the full samples. Furthermore, the sampling ratio of 5% is a good balance between efficiency and accuracy.

7.1.4 Analysis on Diversity and Cooperation

The goal of this experiment is to evaluate the diversity and cooperation between learners. We use the Jaccard Index to measure the diversity between learners, which is widely used to measure the difference between two sets [57]. Let S_{θ_k} and S_{θ_l} be the set of high-confidence samples (> 0.2) predicted by two learners θ_k and θ_l, respectively. The diversity is defined as $|S_{\theta_k} \cap S_{\theta_l}|/|S_{\theta_k} \cup S_{\theta_l}|$. A smaller value indicates that the two learners are more diverse. We use the symmetric KL-divergence to measure the cooperation between learners, which is introduced in Section 5.2. A smaller value indicates that the two learners are more cooperative. The diversity/cooperation of a set of learners is defined as the average of all pairwise diversity/cooperation between two learners. Table 3 shows that on all the datasets, our method recommends a set of more diverse and cooperative learners.

TABLE 2
The Accuracy Using Different Sampling Ratios (SR)

SR	1%	2%	3%	4%	5%	6%	7%	10%	100%
mini	0.838	0.865	0.874	0.882	0.886	0.889	0.891	0.891	0.893
tiered	0.851	0.875	0.887	0.895	0.899	0.901	0.901	0.902	0.907
MNIST	0.591	0.592	0.597	0.601	0.598	0.599	0.603	0.602	0.609
CIFAR-FS	0.526	0.531	0.542	0.548	0.550	0.551	0.547	0.554	0.554

TABLE 3
Comparison of the Diversity and Cooperation Between All Learners and Recommended Learners

	Diversity			Cooperation		
	All	Rec.	Diff	All	Rec.	Diff
mini	0.124	0.063	49.2%	0.959	0.204	78.7%
tiered	0.138	0.082	40.6%	0.942	0.336	64.3%
MNIST	0.449	0.208	53.7%	1.045	0.531	49.2%
CIFAR-FS	0.321	0.264	17.8%	2.311	0.961	58.4%

The smaller values indicate that the recommended learners are more diverse and cooperative.

7.2 Case Studies

In the case studies, we used the same 24 base learners employed in the quantitative evaluation. To demonstrate the generalization of our approach to new tasks, we used the MNIST and CIFAR-FS datasets because there are no base learners pre-trained on them. Based on the experiment results in Section 7.1, we select a trial with higher accuracy for each dataset. The experts started from the setting of recommending learners because it does not need any human involvement. When performing the case studies, we followed the pair analytics protocol [58], where the expert guided the exploration, and we interacted with the tool. This protocol helps the experts focus more on the analysis of the model.

7.2.1 MNIST Dataset

In this case study, we collaborated with expert E1 to understand and diagnose a model built on the MNIST Dataset [52]. She is interested in knowing how FSLDiagnotor supports the selection of base learners and the enhancement of the shots, thus improving the accuracy of the model. The experiment in Section 7.1.3 indicates that a sampling ratio of 5% can better balance performance and efficiency. Thus, E1 sampled $5\% \times 20,000 = 1,000$ samples.

Overview. E1 first observed that four base learners were recommended by FSLDiagnotor (Fig. 9a). She then examined the selected base learners and noticed that "BL-tiered6" made many different predictions from the ensemble model (Fig. 9 A). This needed further investigation to figure out the reason. In the sample view, she observed that the samples were separated into two groups, with the upper ones being samples of black digits with a white background (e.g., Fig. 9 C, D), and the lower ones being samples of white digits with a black background (e.g., Fig. 9 E, F). Most of the regions were covered by the given shots well (e.g., Fig. 9 C, E). However, there were a few regions not covered by the shots (Fig. 9 D, F), where some samples (in gray) were predicted with low confidence. Using the 4 recommended base learners (**R1**), the accuracy was **0.513**. The accuracy was calculated offline before and after the corresponding operations to verify the effectiveness of the improvement with our tool.

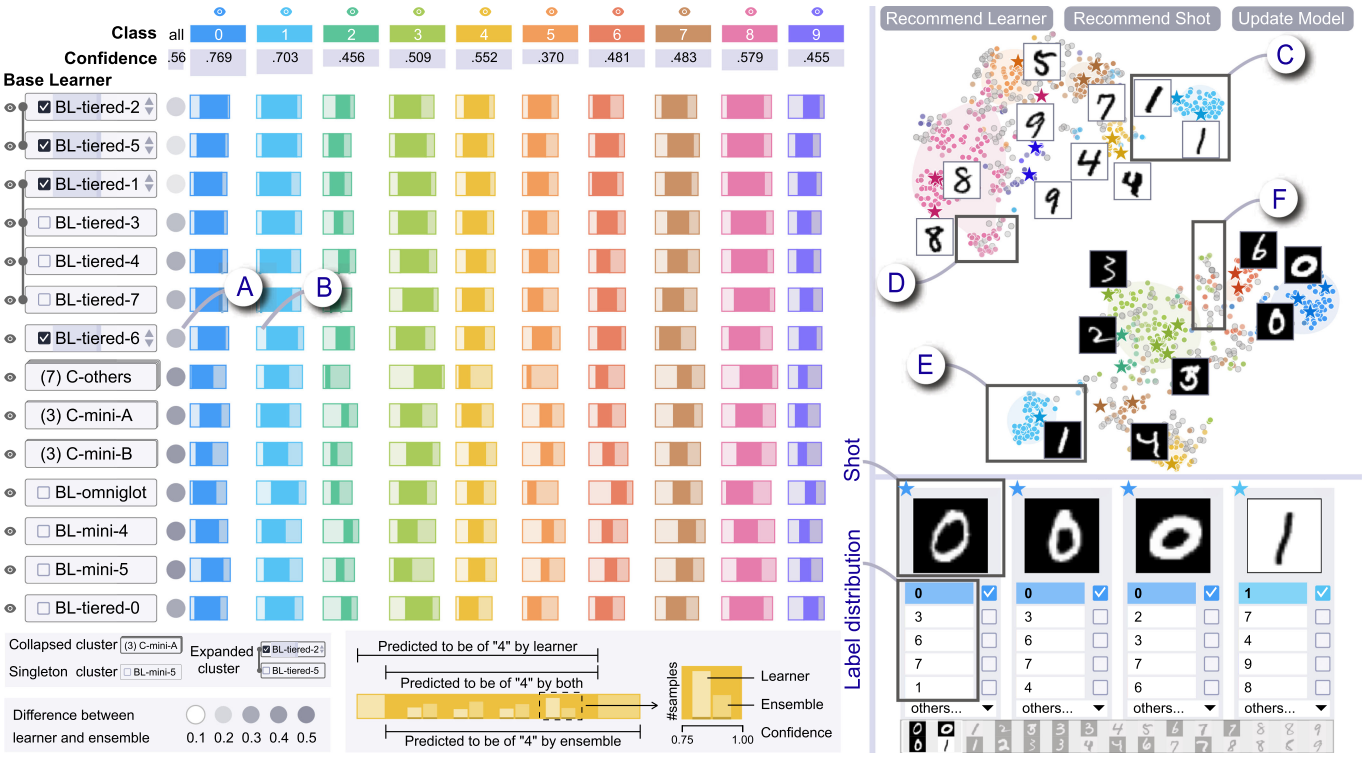


Fig. 9. FSLDiagnotor: (a) *learner view* compares base learners (rows) with the ensemble model, including the overall difference (circles in the first column) and detailed difference (stacked bars in the other columns); (b) *sample view* visualizes the shots and unlabeled samples in context. The image content and label distributions of the samples of interest are displayed below.

Learner-Based Improvement. E1 started the analysis from the base learners. E1 first examined the selected base learners for potential improvement. Since "BL-tiered6" made more different predictions from the ensemble model, she clicked on this learner to examine on which samples it made different predictions. These samples were highlighted in the sample view. Three gray density area also appeared in the sample view, indicating a larger drop in prediction confidence (Fig. 10 A, B, C). She decided to examine these three regions one by one.

E1 began the analysis with *region A*, where most samples were black digits "3" and "5." She noticed that the ensemble model misclassified most samples of "3" to be of "5" or "8," and some samples of "5" to be of "8." Checking the shots near this region, she found that there was no shot of digit "3" and only one shot of digit "5." She decided to add more shots by selecting the samples in this region and clicking "Recommend shot." Samples of "3" and "5" were recommended, which met her expectation. She added one shot for "3" and one shot for "5" (**R2**). Then E1 switched to *region B*. To her surprise, the region contained the samples of black digit "0," but both the learners and the ensemble model misclassified them to be of "8" (Fig. 10 D). The reason was that there were no shots of black digit "0" (Fig. 11a), so the nearest shots of a black digit "8" influenced the predictions of these samples. These misclassifications can be corrected by adding more shots of black digit "0." Since the samples of "0" in this region looked quite similar, she directly labeled one of them as a shot (**R2**). E1 further examined **region C**, where most samples were white digits "4." The learner view showed that most of the base learners, as well as the ensemble model, made the correct predictions (Fig. 10 F). However, "BL-tiered6" misclassified some of them to be of "1" (Fig. 10 E). She noticed that "BL-tiered6" *over-predicts* on "1" compared with other base learners (Fig. 9 B). She clicked the bar and found that many samples of white digits "7" were also predicted to be of "1" by "BL-tiered6." She then concluded that "BL-tiered6" was confused about how to classify the white digits "1," "4," and "7," which caused the drop in the prediction confidence (Fig. 10 C). Due to the poor performance of "BL-tiered6" in *region C*, she decided to remove it (**R1**). After these adjustments, the model was

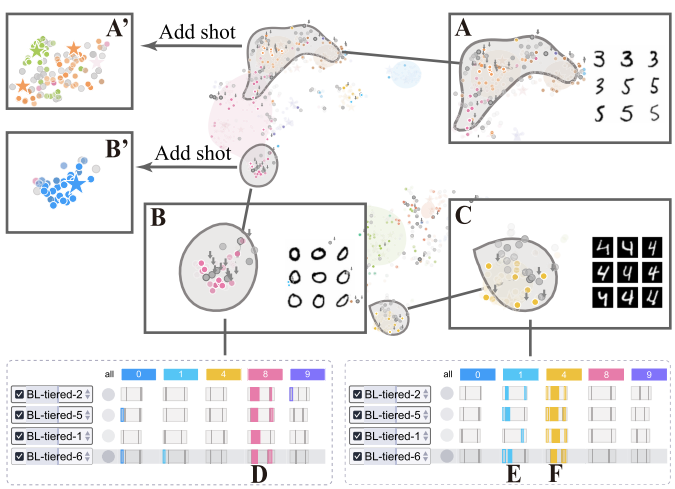


Fig. 10. Analyzing "BL-tiered6." With it in the ensemble model, three regions (A, B, C) have a larger drop in prediction confidence.

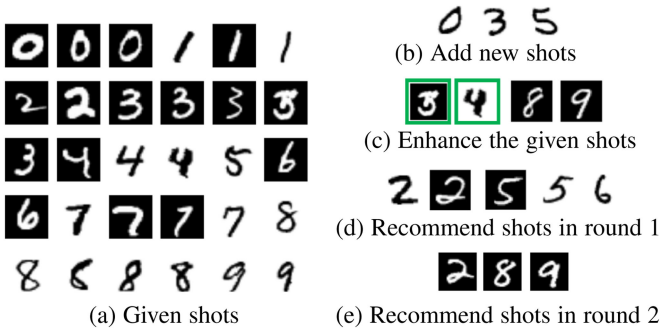


Fig. 11. Enhancing the quality of the given shots (a) by going through steps (b)-(e). The samples with green borders are recommended to be removed.

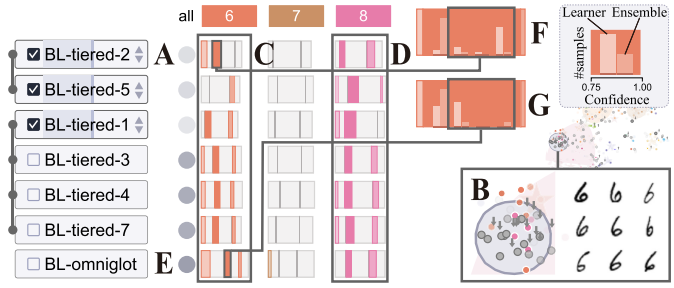


Fig. 12. "BL-tiered2" lowers the prediction confidence in region B. These samples are of "6" but mis-predicted to be of "8" by the ensemble model. In contrast, "BL-tiered2" and "BL-omniglot" make more correct predictions on them.

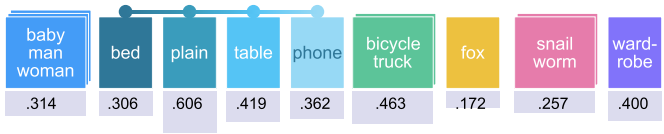


Fig. 13. The initial class clusters and their prediction confidence.

updated. The accuracy was improved from 0.513 to **0.582**. E1 was satisfied that 1) the added shots well covered those two regions (Fig. 10 A′, B′); 2) removing "BL-tiered6" increased the confidence of the samples in *region C* from 0.453 to 0.525.

Shot-Based Improvement. To adapt to the change of the base learners, she used our tool to automatically detect low-quality shots and recommend high-quality ones. Inspired by the query strategy in active learning [59], E1 decided to add a few shots (3-5 shots) in each recommendation. The recommended shots to be added/removed were displayed at the bottom of the sample view. She found two low-quality shots with poor coverage (e.g., Fig. 7b) were detected and removed them (the samples in Fig. 11c with a green border) by clicking the checkbox. For the shots to be added, E1 found that a white digit "8" and two white digits "9" were recommended, which did not appear in the given shots (Fig. 11a). To supplement the shot set that does not contain any white "8" and "9," she selected one from each class, respectively (samples in Fig. 11c without border). E1 updated the model with the new shot set, increasing the accuracy from 0.582 to **0.622**. E1 repeated the recommendation operation again and selected five more shots (Fig. 11d), the accuracy was improved to **0.664**.

Mutually Tuning Between the Base Learners and Shots. To further improve the performance, she switched back to the learner view to see if there were any changes after updating the shots. After examining the three selected learners one by one, she found that "BL-tiered2" (Fig. 12 A) lowered the prediction confidence of some samples in a small cluster. This cluster contained some samples with black digits "6" (Fig. 12 B). While "BL-tiered2" and "BL-omniglot" classified them correctly (Fig. 12 C), the ensemble model misclassified some of them to be of "8" (Fig. 12 D). The distribution of confidence showed that these two learners were more confident than the ensemble model (Fig. 12 F, G). As "BL-tiered2" was already selected in the ensemble model, E1 clicked ▲ to increase the weight of "BL-tiered2." She also added "BL-omniglot" to the ensemble model. She commented that Omniglot [55] was a dataset containing different handwritten characters and was similar to MNIST. She considered that a learner trained on this dataset would be beneficial to the current task. In addition, it increased the diversity among the learners as its predictions differed much from the ensemble model (Fig. 12 E). After increasing the weight of "BL-tiered2" and adding "BL-omniglot" into the ensemble model (**R1**), the accuracy was improved from 0.664 to **0.680**. E1 then added three more shots (Fig. 11e) to adapt to the learner change (**R2**), and the accuracy increased to **0.707**.

Summary. E1 removed 2 low-quality shots and added 13 shots in total. The final accuracy was **0.707**. To achieve comparable performance, the random selection strategy requires 68 more shots and the automatic shot selection algorithm requires 28 more shots. E1 was satisfied with the ability of FSLDiagnotor in helping her identify misclassified regions and verify the recommended learners and shots for such a simple classification task.

7.2.2 CIFAR-FS Dataset

This case study demonstrates the capability of our tool in boosting performance on a natural image dataset, CIFAR-FS [53]. In this case study, we collaborated with E2. As the task involved more classes and contained only 12,000 samples, E2 increased the sampling ratio to 10% and obtained 1,200 samples.

Overview. To improve readability, 20 classes were grouped into 10 clusters. Most clusters looked reasonable. For example, "baby," "man," "woman" were in the same cluster, and "bicycle" and "truck" were in another (Fig. 13). However, E2 found that "plain," "bed," "table," "phone" formed a cluster while "wardrobe" formed another one. The sample view (Fig. 14a) showed that "bed" (Fig. 14 A), "table" (Fig. 14 A), and "wardrobe" (Fig. 14 B) were closed to each other, and "plain" (Fig. 14 C) was away from them. So he dragged "wardrobe" into the cluster and moved "plain" out as another cluster. Seven base learners were recommended, including five ones trained on *tiered*-ImageNet and two ones trained on *mini*-ImageNet. With the recommended learners, the accuracy of the ensemble was **0.497**.

Diagnosing the Clusters With Poor Performance. He first examined the cluster with the lowest confidence (0.172), which only contained the class "fox" (Fig. 13). Twelve samples were predicted to be of "fox." However, some of them were images with leopards (Fig. 14 B2, B3). In this class,

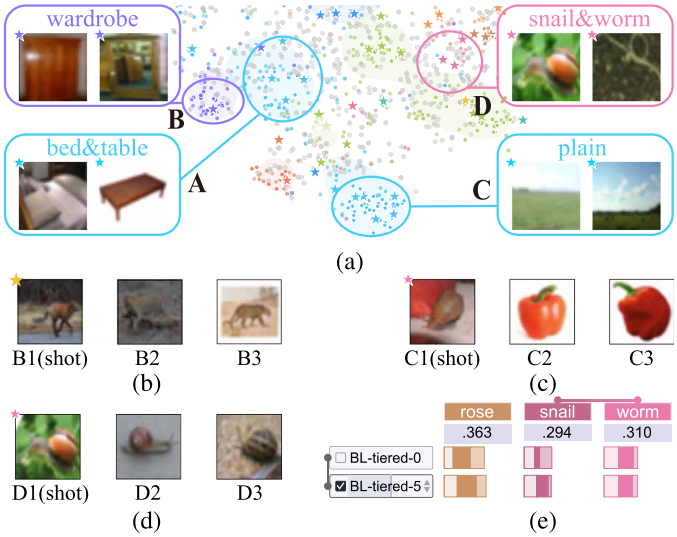


Fig. 14. Analysis of the CIFAR-FS dataset: (a) the sample view; (b) lack of shots for "fox;" (c) an outlier shot of "snail;" (d) poor diversity of the shots of "snail;" (e) the prediction behavior of two learners for "snail."

there was only one shot (Fig. 14 B1). Both the learners and ensemble model have low confidence on the predictions. E2 commented that one shot was insufficient to distinguish "fox" from "leopard." So he added 4 shots for "fox" and 2 shots for "leopard" (**R2**) and then updated the model. The confidence increased to 0.288, and the accuracy reached **0.503**.

Next, he moved to cluster "snail&worm" with a lower confidence of 0.257. After zooming in the cluster, he found that the confidence of "snail" was only 0.228, and many samples of "pepper" (Fig. 14 C2, C3) were predicted to be of "snail." E2 examined the sample view and found a shot of "snail" that contained a red object (Fig. 14 C1). He speculated that this shot disturbed the classification. After removing it (**R2**), the confidence reached 0.294. Then he examined the learner view and noticed the shorter length of the stacked bar charts for "snail" (Fig. 14e). This indicated that only a few samples were predicted to be of "snail." To figure out why, he examined the base learners and found that "BL-tiered5" over-predicted on "snail." The over-predicted samples were snails on non-green backgrounds (Fig. 14 D2, D3) instead of the green background in the shots (Fig. 14 D1). He labeled three such samples to augment the diversity of the shots of "snail" (**R2**). Moreover, he found that some samples of "snail" and "worm" were mixed and hard to be classified (Fig. 14 D). He used FSLDiagnotor to recommend two more shots for each of these two classes and updated the model (**R2**). The confidence increased to 0.378, and the accuracy was **0.530**.

He continued to diagnose cluster "baby&man&woman" (confidence: 0.314) in a similar way, and the accuracy reached **0.541** after labeling 2 shots for "man" and 2 shots for "woman" (**R2**). After removing the outlier shots in the largest cluster "bed&table&phone&wardrobe" and adding six shots for "bed" and "table" (**R2**), the accuracy reached **0.561**.

Improving Base Learners and Shots. The aforementioned diagnosis added/removed some shots. To adapt to these changes, E2 used FSLDiagnotor to recommend the learners and removed "BL-mini7" and "BL-tiered5" (**R1**). The accuracy remained to be **0.561**. To adapt to the change of the base learners, 14 more shots were also recommended (**R2**), and the accuracy increased to **0.594**.

Summary. E2 successfully improved the accuracy from 0.474 to 0.594 with only 37 extra shots. To achieve comparable performance, the random selection strategy requires 115 extra shots, and the automatic shot selection algorithm requires 75 extra shots. He was satisfied that FSLDiagnotor helped find a variety of quality issues of shots more efficiently. "I do not realize that the shot like Fig. 14 C1 hurts performance until I see those misclassified samples." He further pointed out that it was usually difficult to provide representative shots exhaustively. The exploratory environment of the tool helps find the missing shots.

8 Expert Feedback and Discussion

To evaluate the usefulness of FSLDiagnotor, we conducted six semi-structured interviews with the three collaborated experts (E1, E2, E3) and three newly invited ones (E4, E5, E6). The three new experts are Ph.D. students who have worked in the field of machine learning for 5, 3, and 2 years, respectively. In each interview, we spent 5 minutes introducing the design of our tool. Then the experts played with the tool to get familiar with it. For example, they tried to improve the performance by adjusting the selection of learners and/or enhancing the quality of shots. Finally, we presented our case studies and gathered their feedback. Each interview lasted approximately 45-65 minutes. All experts were generally positive about the usability of FSLDiagnotor. They also pointed out a few limitations, which shed light on future work.

8.1 Usability

Facilitating the Performance Improvement. Encouragingly, our experts agreed that FSLDiagnotor was useful for improving model performance. E1 liked the shot quality enhancement module. "Generally, some initial shots are probably of low-quality. I would like to remove the low-quality ones and annotate a few more shots for better performance. The tool recommends high-quality shot candidates for labeling, which reduces my workload." E5 was impressed by the promising accuracy improvement from 0.513 to 0.582 with only three shots added in the MNIST case. In addition, the experts indicated that FSLDiagnotor not only provided an effective way to address the scenarios where only a few shots were available, but also an efficient mechanism to label a set of diverse shots that can better represent the unlabeled samples.

Being Easy to Use and Reducing Analysis Efforts. The experts agreed that the visual design was familiar and easy to understand. E2 commented, "The stacked bar chart is very intuitive and clearly explains the prediction agreement and difference between the learner and ensemble model." E4 believed that our tool could be used by practitioners easily, "They are familiar with bar charts and scatterplots, so according to my experience, 15-30 minutes should be enough for them to get familiar with this tool." E3 shared his experience of improving the performance, "An effective way to improve the performance is to label more shots in

the regions that contain many samples with low confidence. Such regions are highlighted with gray density and easy to identify. With the recommendation function, I only examine the recommended samples from these regions and label the appropriate ones." The experts also commented that although extra analysis of the learners and samples was needed, the efforts were small because of the visual guidance and semantic interactions. Thus, their overall analysis efforts were reduced.

8.2 Limitations

Generalization. In addition to classification tasks, the experts also expressed the need to apply our tool to handle object detection and segmentation. After discussion, we found that the only change was induced by the IoU (Intersection over Union) measure employed in these tasks, which represents the area ratio of the intersection to the union of two shapes. Unlike the binary variable used in the classification task to indicate whether a sample belongs to a class or not, the IoU score is a value between 0 and 1. An interesting problem worth studying is how to effectively convey the distribution of the IoU scores in the learner view. In addition, the experts expressed their need to analyze non-ensemble few-shot models, such as generative models and meta-learning [2]. The subset selection algorithm and the sample view could be directly used to enhance the quality of shots and make adjustments. However, the learner view needs to be re-designed to adapt to the analysis of a single model. We leave this as future work.

Algorithm Scalability. The shot recommendation is frequently performed to improve the performance in the analysis process. The experts usually select a region for the detailed examination, which contains at most thousands of samples. For such cases, the subset selection algorithm can recommend shots in real-time. However, when first providing the overview in the pre-processing stage, the recommendations are from the whole dataset. It may still introduce the scalability issue into this offline process when the dataset consists of tens of thousands of samples or more. For example, it takes around 20 hours to recommend shots from 100,000 samples. It is worth studying how to reduce the pre-processing time. For example, we can study how to use progressive visual analytics techniques [60], [61] to recommend necessary shots progressively.

8.3 Lesson Learned

Using Simple and Familiar Visualization. During the interviews, the experts appreciated the simple and familiar visual designs used in our tool. A simple and intuitive visualization requires less time to learn and allows them to focus more on their analysis tasks. For example, E2 commented, "The learner view can be regarded as a variant of the confusion matrix, with which I am very familiar. Thus, I can go directly to analyze the root cause of low performance, which saves my time and efforts." The experts also pointed out that the visualization could be used in other tasks. For example, all the experts commented that they would like to use this tool to analyze a generic ensemble model, which they commonly used in various tasks. The experts also indicated that the learner view can be directly used to compare datasets from different perspectives.

Employing Steerable Visualization. During the development of FSLDiagnotor, we find that steerable visualization is an effective method to address the scalability issue when handling large-scale data. The core of steerable visualization is to steer the computational efforts to the regions of interest [62]. In FSLDiagnotor, since recommending shots may take a long time, users first identify the regions that lack shots and then steer more computational efforts to recommend shots in those regions. Such steerable shot selection supports the exploration tasks where only a small subset of samples are of interest, such as finding diverse and well-performing learners from a large collection to build an ensemble model.

Providing Semantic Interactions. In many sensemaking processes, users need to adjust the adopted analytical model to form hypotheses and derive conclusions. Most existing interaction techniques rely on users' expertise to adjust analytical models, such as modifying parameters and adding constraints. This requires users to be familiar with the working mechanism of the analytical model and thus limits the usage of the developed visual analysis tool/method. With semantic interactions, users can easily steer the model without expertise in it. Traditional interactions are well studied and several taxonomies are built [63], [64]. However, semantic interaction research is quite new and more work is needed to form a taxonomy. In FSLDiagnotor, we provide a few concrete examples of semantic interactions. We hope these examples can help inspire more research in this direction and build a solid taxonomy for semantic interactions.

9 Conclusion

We have presented a visual analysis tool, FSLDiagnotor, to assist in visually diagnosing an ensemble few-shot classifier for better performance. FSLDiagnotor integrates the sparse subset selection method with an enhanced matrix visualization and a scatterplot to understand the inner workings of the base learners and the coverage of the shots. With such a comprehensive understanding, users can build a better ensemble few-shot learning model by interactively and efficiently improving the selection of base learners and shots. A quantitative evaluation demonstrates the effectiveness of the developed subset selection method in selecting appropriate base learners and enhancing the quality of the shots. Two case studies are conducted to demonstrate the usefulness of our tool in diagnosing the few-shot classifier and improving its performance.

References

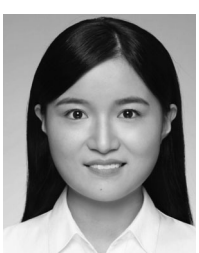
[1] N. Dvornik, C. Schmid, and J. Mairal, "Diversity with cooperation: Ensemble methods for few-shot classification," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 3723–3731.
[2] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surv.*, vol. 53, no. 3, pp. 1–34, 2020.
[3] "Cross-domain few-shot learning (CD-FSL) challenge," 2019, Accessed: Jun. 5, 2022. [Online]. Available: https://www.learning-with-limited-labels.com
[4] "Humpback whale identification," 2019, Accessed: Jun. 5, 2022. [Online]. Available: https://www.kaggle.com/c/humpback-whale-identification/overview

[5] S. Liu, C. Chen, Y. Lu, F. Ouyang, and B. Wang, "An interactive method to improve crowdsourced annotations," *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 1, pp. 235–245, Jan. 2019.
[6] M. Qi, J. Qin, X. Zhen, D. Huang, Y. Yang, and J. Luo, "Few-shot ensemble learning for video classification with slowfast memory networks," in *Proc. ACM Int. Conf. Multimedia*, 2020, pp. 3007–3015.
[7] C. Renggli, L. Rimanic, N. M. Gurel, B. Karlas, W. Wu, and C. Zhang, "A data quality-driven view of MLOps," *IEEE Data Eng. Bull.*, vol. 44, no. 1, pp. 11–23, Jan. 2021.
[8] A. Ng, "MLOps: From model-centric to data-centric AI," 2021. [Online]. Available: https://www.deeplearning.ai/wp-content/uploads/2021/06/MLOps-From-Model-centric-to-Data-centric-AI.pdf
[9] D. A. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler, "Visual analytics: Scope and challenges," in *Visual Data Mining*. Berlin, Germany: Springer, 2008, pp. 76–90.
[10] Y. Tian, Y. Wang, D. Krishnan, J. B. Tenenbaum, and P. Isola, "Rethinking few-shot image classification: A good embedding is all you need?," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 266–282.
[11] N. Dvornik, C. Schmid, and J. Mairal, "Selecting relevant features from a multi-domain representation for few-shot classification," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 769–786.
[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[13] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau, "Visual analytics in deep learning: An interrogative survey for the next frontiers," *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 8, pp. 2674–2693, Aug. 2019.
[14] J. Yuan, C. Chen, W. Yang, M. Liu, J. Xia, and S. Liu, "A survey of visual analytics techniques for machine learning," *Comput. Vis. Media*, vol. 7, no. 1, pp. 3–36, 2021.
[15] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu, "Towards better analysis of deep convolutional neural networks," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 1, pp. 91–100, Jan. 2017.
[16] A. Bilal, A. Jourabloo, M. Ye, X. Liu, and L. Ren, "Do convolutional neural networks learn class hierarchy?," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 152–162, Jan. 2018.
[17] M. Kahng, P. Y. Andrews, A. Kalro, and D. H. Chau, "ActiVis: Visual exploration of industry-scale deep neural network models," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 88–97, Jan. 2018.
[18] M. Liu, J. Shi, K. Cao, J. Zhu, and S. Liu, "Analyzing the training processes of deep generative models," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 77–87, Jan. 2018.
[19] J. Wang, L. Gou, H.-W. Shen, and H. Yang, "DQNViz: A visual analytics approach to understand deep Q-networks," *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 1, pp. 288–298, Jan. 2019.
[20] Y. Ming *et al.*, "Understanding hidden memories of recurrent neural networks," in *Proc. IEEE Conf. Vis. Analytics Sci. Technol.*, 2017, pp. 13–24.
[21] H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. M. Rush, "Seq2seq-Vis: A visual debugging tool for sequence-to-sequence models," *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 1, pp. 353–363, Jan. 2019.
[22] S. Liu, J. Xiao, J. Liu, X. Wang, J. Wu, and J. Zhu, "Visual diagnosis of tree boosting methods," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 163–173, Jan. 2018.
[23] B. Schneider, D. Jackle, F. Stoffel, A. Diehl, J. Fuchs, and D. Keim, "Integrating data and model space in ensemble learning by visual analytics," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 483–496, Jul. 2021.
[24] X. Zhao, Y. Wu, D. L. Lee, and W. Cui, "iForest: Interpreting random forests via visual analytics," *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 1, pp. 407–416, Jan. 2019.
[25] M. P. Neto and F. V. Paulovich, "Explainable matrix - Visualization for global and local interpretability of random forest classification ensembles," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 1427–1437, Feb. 2021.
[26] C. Chen *et al.*, "OoDAnalyzer: Interactive analysis of out-of-distribution samples," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 7, pp. 3335–3349, Jul. 2021.
[27] W. Yang *et al.*, "Diagnosing concept drift with visual analytics," in *Proc. IEEE Conf. Vis. Analytics Sci. Technol.*, 2020, pp. 12–23.
[28] Y. Ming, P. Xu, F. Cheng, H. Qu, and L. Ren, "ProtoSteer: Steering deep sequence model with prototypes," *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 1, pp. 238–248, Jan. 2020.
[29] L. Gou *et al.*, "VATLD: A visual analytics system to assess, understand and improve traffic light detection," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 261–271, Feb. 2021.
[30] F. Heimerl, S. Koch, H. Bosch, and T. Ertl, "Visual classifier training for text document retrieval," *IEEE Trans. Vis. Comput. Graphics*, vol. 18, no. 12, pp. 2839–2848, Dec. 2012.
[31] M. Behrisch, F. Korkmaz, L. Shao, and T. Schreck, "Feedback-driven interactive exploration of large multidimensional data supported by visual classifier," in *Proc. IEEE Conf. Vis. Analytics Sci. Technol.*, 2014, pp. 43–52.
[32] P. Bruneau and B. Otjacques, "An interactive, example-based, visual clustering system," in *Proc. Int. Conf. Inf. Vis.*, 2013, pp. 168–173.
[33] B. Höferlin, R. Netzel, M. Höferlin, D. Weiskopf, and G. Heidemann, "Inter-active learning of ad-hoc classifiers for video visual analytics," in *Proc. IEEE Conf. Vis. Analytics Sci. Technol.*, 2012, pp. 23–32.
[34] J. G. S. Paiva, W. R. Schwartz, H. Pedrini, and R. Minghim, "An approach to supporting incremental visual data classification," *IEEE Trans. Vis. Comput. Graphics*, vol. 21, no. 1, pp. 4–17, Jan. 2015.
[35] S. Xiang, X. Ye, J. Xia, J. Wu, Y. Chen, and S. Liu, "Interactive correction of mislabeled training data," in *Proc. IEEE Conf. Vis. Analytics Sci. Technol.*, 2019, pp. 57–68.
[36] S. Jia, Z. Li, N. Chen, and J. Zhang, "Towards visual explainable active learning for zero-shot classification," *IEEE Trans. Vis. Comput. Graphics*, vol. 28, no. 1, pp. 791–801, Jan. 2022.
[37] E. Elhamifar, G. Sapiro, and S. S. Sastry, "Dissimilarity-based sparse subset selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 11, pp. 2182–2197, Nov. 2015.
[38] A. Schrijver, *Theory of Linear and Integer Programming*. Hoboken, NJ, USA: Wiley, 1998.
[39] X. Zhu and A. B. Goldberg, *Introduction to Semi-Supervised Learning*. San Rafael, CA, USA: Morgan & Claypool, 2009.
[40] Y. Yang, Z. Ma, F. Nie, X. Chang, and A. G. Hauptmann, "Multi-class active learning by uncertainty sampling with diversity maximization," *Int. J. Comput. Vis.*, vol. 113, no. 2, pp. 113–127, 2015.
[41] K. Dinkla, M. A. Westenberg, and J. J. van Wijk, "Compressed adjacency matrices: Untangling gene regulatory networks," *IEEE Trans. Vis. Comput. Graphics*, vol. 18, no. 12, pp. 2457–2466, Dec. 2012.
[42] P. H. Sneath *et al.*, *Numerical Taxonomy: The Principles and Practice of Numerical Classification*. San Francisco, CA, USA: W.H. Freeman, 1973.
[43] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1532–1543.
[44] S. Liu, D. Maljovec, B. Wang, P.-T. Bremer, and V. Pascucci, "Visualizing high-dimensional data: Advances in the past decade," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 3, pp. 1249–1268, Mar. 2017.
[45] Y. Meng, H. Zhang, M. Liu, and S. Liu, "Clutter-aware label layout," in *Proc. IEEE Pacific Vis. Symp.*, 2015, pp. 207–214.
[46] A. Mayorga and M. Gleicher, "Splatterplots: Overcoming overdraw in scatter plots," *IEEE Trans. Vis. Comput. Graphics*, vol. 19, no. 9, pp. 1526–1538, Sep. 2013.
[47] J. Yuan, S. Xiang, J. Xia, L. Yu, and S. Liu, "Evaluation of sampling methods for scatterplots," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 1720–1730, Feb. 2021.
[48] A. Endert, "Semantic interaction for visual analytics: Inferring analytical reasoning for model steering," *Synth. Lectures Visualization*, vol. 4, no. 2, pp. 1–99, 2016.
[49] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL, USA: CRC Press, 2012.
[50] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3637–3645.
[51] M. Ren *et al.*, "Meta-learning for semi-supervised few-shot classification," in *Proc. Int. Conf. Learn. Representations*, 2018.
[52] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[53] L. Bertinetto, J. F. Henriques, P. Torr, and A. Vedaldi, "Meta-learning with differentiable closed-form solvers," in *Proc. Int. Conf. Learn. Representations*, 2019.
[54] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.
[55] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[56] M. Boudiaf, I. Ziko, J. Rony, J. Dolz, P. Piantanida, and I. Ben Ayed, "Information maximization for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 2445–2457.
[57] N. J. v. Eck and L. Waltman, "How to normalize co-occurrence data? An analysis of some well-known similarity measures," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 60, no. 8, pp. 1635–1651, 2009.
[58] R. Arias-Hernandez, L. T. Kaastra, T. M. Green, and B. Fisher, "Pair analytics: Capturing reasoning processes in collaborative visual analytics," in *Proc. Int. Conf. Syst. Sci.*, 2011, pp. 1–10.
[59] B. Settles, "Active learning literature survey," Madison, WI, USA: Univ. Wisconsin-Madison, Tech. Rep. 1648, 2009.
[60] J.-D. Fekete and R. Primet, "Progressive analytics: A computation paradigm for exploratory data analysis," 2016, *arXiv:1607.05162*.
[61] J.-D. Fekete, "ProgressiVis: A Toolkit for steerable progressive analytics and visualization," in *Proc. 1st Workshop Data Syst. Interactive Anal.*, 2015, Art. no. 5.
[62] M. Williams and T. Munzner, "Steerable, progressive multidimensional scaling," in *Proc. IEEE Symp. Inf. Vis.*, 2004, pp. 57–64.
[63] J. S. Yi, Y. ah Kang, J. Stasko, and J. A. Jacko, "Toward a deeper understanding of the role of interaction in information visualization," *IEEE Trans. Vis. Comput. Graphics*, vol. 13, no. 6, pp. 1224–1231, Nov./Dec. 2007.
[64] J. Heer and B. Shneiderman, "Interactive dynamics for visual analysis," *Commun. ACM*, vol. 55, no. 4, pp. 45–54, 2012.

Weikai Yang received the BS degree from Tsinghua University. He is currently working toward the PhD degree with Tsinghua University. His research interests include visual text analytics and interactive machine learning.

Xi Ye received the BS degree from Tsinghua University. He is currently working toward the PhD degree with the University of Texas at Austin. His research interests include focuses on building interpretable and robust models for complex NLP tasks.



Xingxing Zhang received the BE and PhD degrees in signal and information processing from Beijing Jiaotong University, in 2015 and 2020, respectively. She is a postdoc with the Department of Computer Science, Tsinghua University. Her research interests include data selection, zero/few-shot learning, and adversarial learning. She received the excellent Ph.D. thesis award from the Chinese Institute of Electronics in 2020.

Lanxi Xiao received the BA degree in art & technology (information design) from Tsinghua University, in 2020. She is currently working toward the master's degree with Tsinghua University. Her research interests include information, interaction, and innovation design, as well as Human-AI collaboration design.

Jiazhi Xia is a professor with the School of Computer Science and Engineering, Central South University. His research interest includes data visualization, visual analytics, and computer graphics. Recently, he has performed research in high-dimensional data visualization, graph visualization, and visual analytics for machine learning and published more than 10 IEEE VIS papers. He has served as the survey paper co-chair of ChinaVis in 2019-2020, paper co-chair of ChinaVis in 2021.

Zhongyuan Wang received the BS, MS, and PhD degrees from the Renmin University of China. He is the director with the AI team and a vice president with Kuaishou. His research interests include knowledge graph, natural language processing, information retrieval, and deep learning. Before Kuaishou, he worked for Meituan, Facebook, and Microsoft Research separately and led the research and development of artificial intelligence techniques. He received the 2018 MIT TR Innovators Under 35 China.

Jun Zhu received the BS and PhD degrees from the Department of Computer Science and Technology, Tsinghua University. He is a professor with Tsinghua University. His research interests include primarily on developing statistical machine learning methods to understand scientific and engineering data arising from various fields. He was an adjunct faculty of Carnegie Mellon University. He is an associate editor-in-chief of *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Hanspeter Pfister received the PhD degree from the State University of New York at Stony Brook, and the MS degree in electrical engineering from ETH Zurich. He is an Wang professor with Harvard University. His research interest includes biomedical image analysis and visualization, image and video analysis, and visual analytics in data science. He worked as an associate director and senior research scientist with Mitsubishi Electric Research Laboratories. He is the recipient of the 2010 IEEE Visualization Technical Achievement award.

Shixia Liu (Fellow, IEEE) received the BS and MS degrees from the Harbin Institute of Technology, the PhD degree from Tsinghua University. She is a professor with Tsinghua University. Her research interests include visual text analytics, visual social analytics, interactive machine learning, and text mining. She worked as a research staff member with IBM China Research Lab and a lead researcher with Microsoft Research Asia. She is an associate editor-in-chief of *IEEE Transactions on Visualization and Computer Graphics*.