Applied Intelligence
https://doi.org/10.1007/s10489-021-02515-y

Active learning using a self-correcting neural network (ALSCN)

Velibor Ilić [1] · Jovan Tadić [2]

Accepted: 6 May 2021

Abstract
Data labeling represents a major obstacle in the development of new models because the performance of machine learning models directly depends on the quality of the datasets used to train these models and labeling requires substantial manual effort. Labeling the entire dataset is not always necessary, and not every item from the image dataset contributes equally to the training process. Active learning or guided labeling is one of the attempts to automate and speed up labeling as much as possible. In this study we present a novel active learning algorithm (ALSCN) that contains two networks, convolutional neural network and self-correcting neural network (SCN). The convolutional network is trained using only manually labeled data, and after training that network it predicts labels for unlabeled items. The SCN network is trained with all available items, some of those items are manually labeled and remaining items are automatically labeled with previous network. After training SCN network, it predicts new labels for all available items, and the new labels are compared with the labels used for training. Items in which differences have been identified are selected for manual labeling and then added to dataset of previously manually labeled items. After that, the convolutional network is trained with extended dataset and previously described steps are repeated. Our experiments show that the network trained using items selected by the proposed method exceeds the performance of a network trained with the same number of items randomly selected from the set of available items. Items from the complete datasets are selected in several iterations, and used for training the models. The accuracy of the models trained with selected items matched or exceeded the accuracy of models trained with the entire dataset, which shows the extent of reduction in the required manual labeling effort. The efficiency of presented algorithm is tested on three datasets (MNIST, Fashion MNIST and CIFAR-10). The final results show that manual labeling is required for only 6.11% (3667/60,000), 23.92% (14,353/60,000) and 59.4% (29,704/50,000) items, in case of MNIST, Fashion MNIST and CIFAR-10 dataset, respectively.

Keywords Active learning · Machine learning · Convolutional neural networks (CNN) · Dataset labeling

1 Introduction

Neural networks were discovered a long time ago, but their wider practical applications began in the last decade, as a result of the increase in the amount of available data and computing power. Most of machine learning (ML) techniques require large amounts of data, especially in the case of supervised learning. Accurate data labeling is a basic prerequisite for many ML projects. Currently, ML is applied in the areas of video/image analysis, Internet of things (IOT), industrial control systems, self-driving vehicles, Internet, etc., where it is possible to collect a large amount of data in a short period of time. A more challenging problem is to provide the labeling of such data, if necessary [1]. Data labeling represents a crucial prerequisite in the development of new models because the performance of ML models directly depends on the quality of the input datasets used to train these models.

One of the most popular types of neural networks today is the convolutional neural network (CNN). They can be applied to a wide variety of tasks such as computer vision, object detection, image classification, pattern recognition, and scene segmentation [2] [3]. In training neural networks for classification tasks, it is usually necessary to provide a relatively large dataset that contains labeled data [4]. Labeling the input dataset for neural networks is a time-consuming, tedious and often expensive process, especially if the labeling requires expertise in a particular domain.

✉ Velibor Ilić
ilicv@EUnet.rs

Jovan Tadić
JTadic@lbl.gov

[1] RT-RK Automotive, Novi Sad 21000, Serbia

[2] Climate and Ecosystem Sciences Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

Published online: 01 June 2021

Furthermore, labeling the entire dataset is not always necessary, and not every item from the image dataset contributes equally to the training process [5]. For example, if the distribution of samples per classes is imbalanced, the priority in labeling is commonly given to classes with a smaller number of examples, so that all classes are more equally represented. A network trained for classification is usually not equally successful in identifying all classes of items. In theory, the network caninitially be trained with a smaller dataset, and after checking the efficiency of the prediction on test dataset, the initial dataset can be extended with supplemental items from problematic classes. After extension of the training dataset, the network can be re-trained. Addition of the examples of classes that are already successfully recognized would not significantly increase the accuracy of the predictions [5] [6].

Ideally, the datasets used for training classification networks are expected to contain only items that are unambiguous and accurately labeled. Unfortunately, in practice it is not uncommon to find a fraction of problematic items (e.g. noise) in the training datasets [7]. Considering that items are usually manually labeled, errors can occur due to subjective assessment and perception-related causes. Also, some datasets are created by extracting images from social networks, where individual objects on such images are not always correctly labeled at the source. The presence of noise in the training datasets can significantly reduce the accuracy of classification even with the best classifiers, and the noise at labels is considered more harmful thanthe noise in inputs attributes [8, 9].

If the initial dataset contains incorrectly labeled items, in order to improve the quality of the dataset the ambiguous items should be identified, and if possible, reviewed and corrected by an expert. Simply removing ambiguous items from a dataset increases the risk of losing very important examples, which is especially problematic for small datasets. Verifications, relabeling and cleaning such datasets from errors could require a lot of effort [8]. The problem of missing labels in datasets can be viewed as an extreme case of noisy label data because models trained with smaller subset of labeled items can be used to predict labels for unlabeled items [10, 11]. Self-error-correcting convolutional neural networks (SEC-CNN) are a special class of neural networks that are capable of dealing with the noisy labels problem. SEC networks can simultaneously correct improbable labels and optimize the model by predicting new labels [10].

Active learning or **guided labeling** is one of the attempts to automatize and speed up labeling as much as possible. It is a type of semi-supervised learning where a small part of labeled items and significant amount of unlabeled data are used in the process of training models [12–14]. The model is iteratively trained, and the algorithm has the ability to select the data from which it learns, so it can choose the most useful items from the set of unlabeled data [15]. Also, the algorithm has the ability to actively request human assistance when it is necessary to label selected items, see Fig. 1 [8, 16]. This approach does not use the complete available dataset; it is especially useful in situations where a lot of unlabeled data are available while manual labeling is expensive [17].

There are three main approaches to how an active learning algorithm can request human assistance [18]:

- The **pool-based sampling scenario**: in this approach the learner is initially trained with a subset of available items that have been manually labeled, and remaining instances from the entire data pool will be used in evaluating how accurately the model can classify the data. Then, the most informative instances will be selected and sent to an annotator for labeling.
- The **stream-based selective sampling scenario**: in this approach each unlabeled item is examined one-by-one using initially trained model and the learner decides whether to automatically assign a label or to ask for the assistance from a human annotator for each item separately. In this approach the model can be trained after the examination of each item.
- The **membership query synthesis scenario**: in this approach the learner can generate training instances; this can be useful if the dataset is small.

In an ideal case, in each iteration the active learning algorithm can select one item from a set of unlabeled items that represents the best candidate for manual labeling; this is not appropriate for models based on CNNs because the extension of the training set with just one item usually does not make a significant difference and usually it is computationally expensive to train a model after each iteration [2].

The main idea in this study is that the active learning algorithm can achieve a better accuracy if it has the freedom to choose the items that will be used in training. Active learning is based on the assumption that items in the dataset do not have equal informative value, so adding new items that are similar to items already existing in the dataset will not lead to higher accuracy [6].

The approach used to determine which items from a set of available data will be selected for labeling is called a query

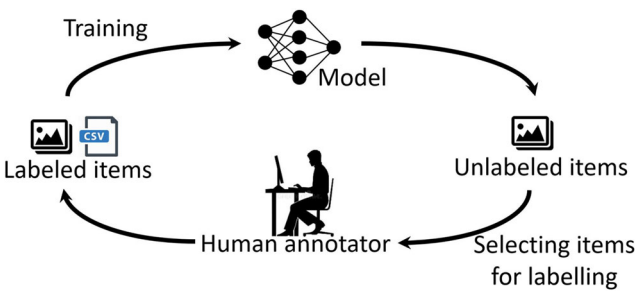

Fig. 1 Active learning

strategy [18]. Strategies that are used for the selection of unlabeled items can be categorized into following groups:

- uncertainty sampling is a strategy for identifying unlabeled items that are near a decision boundary [19, 20], the following strategies are subcategories of uncertainty sampling:
  - least confidence strategy, one item with smallest confidence in prediction will be selected for manual labeling in the next learning iteration [21],
  - margin sampling, algorithm selects items for manual labeling where highest similarity in predictions between different classes is noticed [21],
  - entropy-based, priority for manual labeling is given to items with the highest variance in their predictions, this approach attempts to sequentially minimize the expected entropy (uncertainty) of the labels for the unlabeled items [11],
- query-by-committee (QBC), in this approach several models are trained with available labeled items, each of those models are used to predict labels for unlabeled items, the predicted labels are compared and the items where differences appeared are selected for manual labeling [22],
- heuristic based algorithm, a strategy where sample selection is based on a heuristic objective function, for example highest entropy or geometric distance to decision boundaries [23–26],
- expected model change, aims to select the instance that would cause the greatest change in the current model if we knew its label [18],
- expected error reduction, approach aims to measure how likely it is to reduce generalization error [18],
- variance reduction, this approach aims to reduce generalization error by minimizing output variance indirectly [18],
- density-weighted methods, this approach aims to estimate future errors and output variances by weighting items, the main idea is that informative instances should not only be those which are uncertain, but also those which are "representative" of the underlying distribution [18].

The active learning using a self-correcting neural network (ALSCN) method presented in this study exhibits certain similarities with query-by-committee (QBC) and uncertainty sampling.

In QBC approach several models are trained with labeled items, each of these models predicting labels for unlabeled items, predictions are compared, and differences are selected for manual labeling. In the next iteration, the training dataset is extended with new manually labeled items and previous steps are repeated. In the ALSCN strategy presented in this study, the model of self-correcting neural network is trained with all available items, a combination of manually and automatically labeled items, and this network is used to predict new labels for the same items that have been used for training; comparing labels used for training with predicted labels represents an attempt to identify incorrectly labeled items. Items exhibiting such differences are selected for manual labeling in next training iterations.

There is also a similarity with the method presented in the study [10]. They are trying to automatically identify and correct mislabeled items at the training set using self-error-correcting convolutional neural network (SEC CNN). In our approach (ALSCN) for items that are identified as incorrectly labeled, new labels will not be assigned automatically. For such items the active learner framework will request confirmation from a human expert whether the identified item is really incorrectly labeled or perhaps it is correctly labeled but represents a marginal case.

2 Datasets

The efficiency of active learning using a self-correcting neural network (ALSCN) method is evaluated at three publicly available and well-known datasets:

- MNIST dataset that contains images of handwritten digits. Dataset contains 70,000 28×28 grayscale images in 10 different classes, 60,000 for training images and 10,000 for testing. Each image is associated with a label representing 10 classes (0–9).
- MNIST Fashion dataset that contains images of clothing. Dataset contains 70,000 28×28 grayscale images in 10 different classes, 60,000 for training images and 10,000 for testing. Each image is associated with a label from 10 classes: t-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot.
- The CIFAR-10 dataset contains images of animals and man-made objects. The CIFAR-10 dataset contains 60,000 32×32 color images in 10 different classes, 50,000 for training images and 10,000 for testing. Each of those images are associated with a label from 10 classes: airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks.

Figure 2 shows example images from datasets MNIST, Fashion MNIST, and CIFAR-10:

3 Network configuration

The ALSCN active learning method described in this research is based on two convolutional networks that have different

Fig. 2 Example images from datasets MNIST, Fashion MNIST, and CIFAR-10

roles. The first convolutional network is trained using only manually labeled data. The second network represents a self-correcting neural network (SCN); that network is trained using all available items from the training data (combination of manually and automatically labeled items). After the training, that network predicts new labels for all available items from the training set. Both of these networks have the same configuration but their role in this framework is different, and they are trained using different datasets. In the experiments described in this research, convolutional networks with a similar configuration are used for all three datasets (MNIST, Fashion MNIST and CIFAR-10), the only difference is in the dimensions of the input matrices; the networks used for MNIST and Fashion MNIST datasets take a 28×28×1 matrix as input, while the network used for CIFAR-10 dataset expects 32×32×3 matrix, due to the color images. There are no other differences at internal layers. The first convolutional layer contains 16 3×3 kernels with ReLu activation, a max pooling layer with a pooling size of 2×2, a dropout probability of 0.25, next convolutional layer contains 32 3×3 kernels with ReLu activation, a max pooling layer with a pooling size of 2×2, a dropout probability of 0.25, convolutional layer contains 64 3×3 kernels with ReLu activation, a max pooling layer with a pooling size of 2×2, a dropout probability of 0.25. These convolutional layers are followed by fully connected layers: fully connected layer with 128 output neurons, dropout layer with dropout probability of 0.25, fully connected layer with 64 output neurons, dropout layer with dropout probability of 0.25, fully connected layer with 32 output neurons, dropout layer with a dropout probability of 0.25, and output layer containing 10 neurons, (Table 1 and Fig. 3). At all layers, except the last one the ReLU activation function is used. Adam is used as an optimizer. The duration of training was limited at 350 epochs, with two additional parameters: *ReduceLROnPlateau* with patience 10 and *EarlyStopping* with patience 25. Parameter *ReduceLROnPlateau* reduces learning rate if there was no improvement of the accuracy on the validation dataset for 10 epochs. *EarlyStopping* interrupts training if there is no improvement of the accuracy on the validation dataset for 25 epochs.

Configuration of the network that we used in this paper is presented in Table 1 and Fig. 3, but the ALSCN algorithm can work with any network configuration that makes sense to apply to such a dataset. The selection of parameters for the networks used in the ALSCN method does not differ from the selection of parameters for the networks to be used for training, as if we had such a dataset with labeled data. More details on this will be presented in chapter 8 Variations of the ALSCN algorithm and Fig. 13.

4 Description of iterative training process

Prior to training, the initial dataset was divided into training and test subsets, following the common practice. The following algorithm was applied only on items from the training subset and it could reduce the number of items requiring manual labeling.

The proposed algorithm contains two main parts, (a) initialization and (b) iterative part. In Figs. 4 and 5 processes are tagged with labels (**p1**, **p2**, **p3**, **p4**, and **p5**), datasets are tagged with labels (**d1**, **d2**, **d3**, **d4**, and **d5**), and some common errors are tagged with labels (**e1**, **e2**).

In the initialization part, we start with the dataset without labeled items (**d1**). A small group of items from the initial dataset is randomly selected for manual labeling (**p1**), and the results are saved as initial dataset (**d2**).

Table 1 Configuration of convolutional networks

Layer type	Size	Activation
Input for MNIST and Fashion MNIST	28×28×1	
Input for CIFAR-10	32×32×3	
Convolutional layer	16×3×3	ReLu
Max pooling layer	2×2	
Dropout	0.25	
Convolutional layer	32×3×3	ReLu
Max pooling layer	2×2	
Dropout	0.25	
Convolutional layer	64×3×3	ReLu
Max pooling layer	2×2	
Dropout	0.25	
Fully connected	128	ReLu
Dropout	0.2	
Fully connected	64	ReLu
Dropout	0.25	
Fully connected	32	ReLu
Dropout	0.25	
Fully connected	10	Sigmoid
Outputs	10	

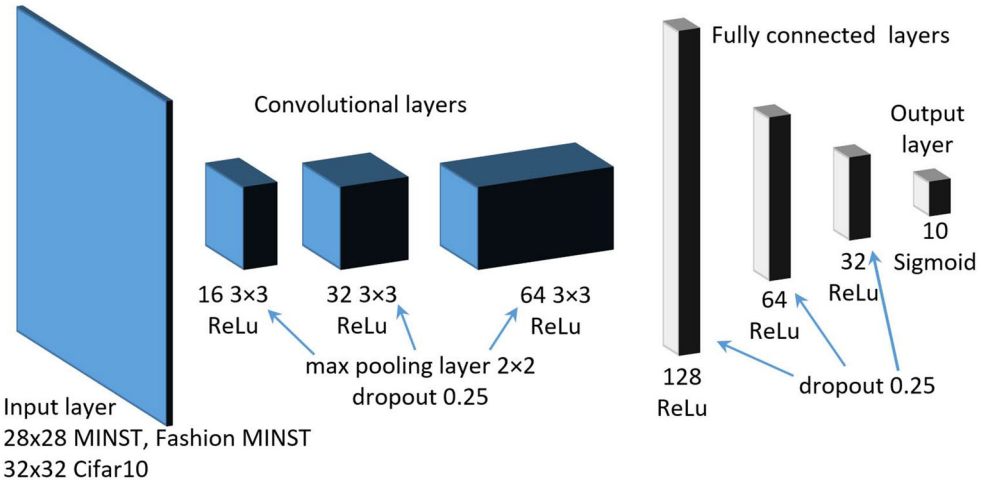

Fig. 3 Structure of convolutional networks

The iterative part begins with training the neural network (**p2**) with items from dataset (**d2**). After training, the network (**p2**) is used to predict labels for remaining unlabeled items (automatically labeling items). This step creates the first version of the dataset where all available items are labeled (**d3**). Self-correcting network (**p3**) is trained with dataset (**d3**) that contains manually and automatically labeled items. After the training is completed this self-correcting network (**p3**) is used to predict new labels for all items that are used for training that network (manually and automatically labeled items), and this step creates dataset (**d4**). By doing simple comparison between datasets (**d3**) and (**d4**) we can identify four groups of items:

- Group (**c1**) contains items that are manually labeled (**d3**) and their predicted value at (**d4**) match each other, and those items represent the confirmation that network is successfully trained to predict labels as it is expected,
- Group (**c2**) contains items that are automatically labeled (**d3**) and their predicted value at (**d4**) match each other, those items are considered just conditionally correct because they were not evaluated by a human; this group usually contains a small percentage of items that are incorrectly labeled, and in each subsequent iteration the number of incorrectly labeled items decreases,
- Errors in the prediction in manually labeled items (**e1**) and,
- Errors that appear in automatically labeled items (**e2**).

In the proposed approach, the errors that appear in manually labeled items (**e1**) can be simply replaced with correct values from the dataset with manually labeled items (**d2**), especially because we used publicly available and well-known datasets for which we can assume that the items in these datasets are correctly labeled. However, in a real-life situation, if an error appears in the group of items that are previously manually labeled (**e1**) it could point to incorrectly labeled initial items. Those items could be selected for double checking by a human expert. Even more interesting are the errors in

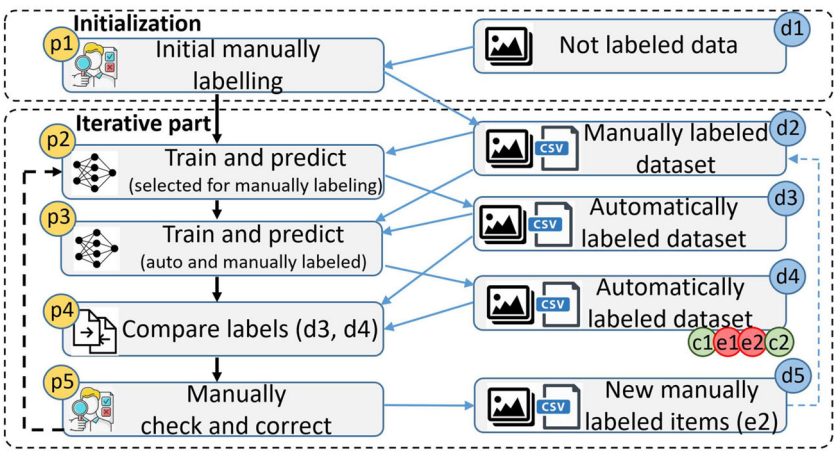

Fig. 4 Overview of the ALSCN algorithm for iterative training

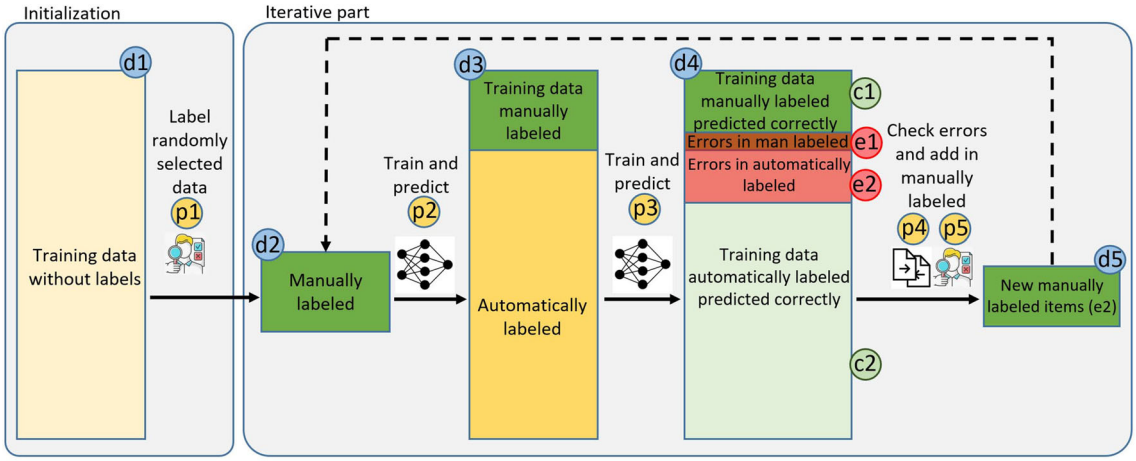

Fig. 5 Overview of the ALSCN algorithm for iterative training

automatically labeled items (**e2**) because those errors have not been previously checked by a human, so all items from that group should be manually checked and corrected if necessary. All items that have been manually checked in the previous step should be appended to the dataset that contains manually labeled items (**d2**). That represents the last step in the iterative part of this algorithm and previously described steps are repeated in a loop.

We can count errors in automatically labeled items (**e2**) that need to be corrected, as a condition for the completion of such a loop; if the number of such errors is less than a threshold or if satisfactory results have been achieved, we can consider that sufficient items from the training set are selected for manual labeling. Furthermore, the networks at (**p2**) and (**p3**) can load previously saved network parameters, which reduces training time and increases accuracy. Networks trained in the previous iteration can already accurately predict the labels for most items from the new dataset in current iteration, so instead of a training network from scratch it only needs to learn the differences related to the previous iteration. This is especially evident in later iterations when only a few items are detected per iteration. Related to previous iteration: dataset (**d2**) is extended with new manually labeled items and in dataset (**d3**) some of automatically labeled items are replaced with new manually labeled items.

The main idea behind this approach is to detect as many errors as possible in a group of automatically labeled items (**e2**) in each iteration, and then ask human assistance to resolve the issue. If the dataset used for network training contains incorrectly labeled items, these items are usually different from other items that belong to such a class, and for a neural network it is usually more difficult to learn mislabeled items.

In the next chapter we presented results of this method applied on three different datasets.

5 Experiments on MNIST, FASHION MNIST and CIFAR-10 datasets

In the following experiments selected to demonstrate the efficiency of the proposed ALSCN method, convolutional network is trained using manually labeled items and self-correcting network is trained using combinations of manually and automatically labeled items (initially unlabeled items). Publicly available labeled datasets (MNIST, Fashion MNIST, and CIFAR-10 respectively) were split into two subsets; a smaller subset of it is used to train models, and the remaining images are treated as unlabeled. The goal of the experiment is to utilize active learning method in an attempt to use as little labeled data as possible maintaining similar prediction accuracy as in the case of using complete dataset.

MNIST dataset contains 70,000 labeled items, 10,000 of those items are test data and other 60,000 can be used for training network. In the following results, we consider that items from the training dataset are not labeled. A script that simulates human labeling by reading correct values from the original dataset has been deployed. Initially, the script reads labels for 1000 randomly selected items and those values are saved in dataset (**d2**) that is used for training the network (**p2**). After training is complete, the network can predict labels for items from the training subset that are not labeled. The self-correcting network (**p3**) is trained with the entire dataset (60,000 items, combination of manually and automatically labeled items (**d3**)), and after completing the training that network predicts new labels for 1000 items that are manually labeled and 59,000 automatically labeled items. Those new predicted labels are written in a new dataset (**d4**). We can identify items having equal values (**c1, c2**) and items where predictions are different (**e1, e2**) by comparing datasets (**d3**) and (**d4**). Group (**c1**) contains items that are manually labeled, and its predicted values are equal at datasets (**d3**) and (**d4**). Group (**e1**) contains items that are manually labeled in (**d3**)

Fig. 6 Example of images selected for manual labeling

and their predicted values at (**d4**) do not match each other. In this case it is clear that these values at (**d4**) are incorrectly predicted and can be simply replaced by values from the dataset (**d3**) that was manually labeled. In a real-life situation, the items from group (**e1**) may indicate incorrectly labeled items, which could be double checked by a human expert.

The remaining two groups refer to the items that are automatically labeled; in the group of items (**e2**) where the predicted values from datasets (**d3**) and (**d4**) do not match, at least one of these two values is not correct, but since both values are automatically assigned, it is not possible to determine which one is correct, so these items should be checked by a human (or in this case by a script that replaces these values with the correct one from the original dataset). The last group from Figs. 4 and 5, marked as (**c2**) contains items with equal values in (**d3**) and (**d4**), and these items can be considered as conditionally correctly labeled because these items at (**d3**) are automatically labeled. Some items belonging to group to group (**c2**) are incorrectly labeled even if predicted values for those items match in (**d3**) and (**d4**). In order to eliminate errors in this data group, it is necessary to perform several iterations of this algorithm. These items that are manually checked/validated are removed from the unlabeled dataset and appended to dataset (**d2**) that contains previously manually labeled items. The examples of items selected for manual labeling from group (**e2**) from datasets MNIST, Fashion MNIST, and CIFAR-10, are presented in the Fig. 6, which shows one such item from each class. The model usually incorrectly predicts labels if an item resembles items from another category. For example, if we look through items from the MNIST dataset, the expected value for the first item is 1, but this item also looks like no. 7. Similar situation is with items 4 (9), 5 (3) and 9 (5), these items being reminiscent of numbers written in parentheses. In other cases when the items were not correctly predicted, they were not clear or they do not look like other items from the same class, item with numbers no. 3 and no. 8 in the MNIST dataset. Another example that is difficult to predict correctly is from the Fashion MNIST dataset, where some images contain several objects.

To make the assessment of the effectiveness of this method easier, a network with identical architecture is also trained with the same number of randomly selected items as a network trained with manually labeled items. Prediction accuracy in the following images represents the number of correctly classified images in relation to the number of tested images. In Fig. 7 we present the accuracy of the predictions according to the number of items used for training within three different datasets (MNIST, Fashion MNIST, and CIFAR-10):

- **red line** represents prediction accuracy of CNN network (**p2**), that is trained with manually labeled items,
- **green line** represents prediction accuracy of the self-correcting network (SCN) (**p3**), that is trained on manually labeled items and automatically labeled items (number of manually labeled items are shown at X axis),
- **blue line** represents average prediction accuracy of five networks trained with the same number of randomly selected items.

The results achieved using three datasets (MNIST, Fashion MNIST and CIFAR-10) is presented in Table 2. The columns show the results of CNN predictions that have been trained in different ways. In the row “CNN, (manually labeled)" we present the results of CNN network that are trained using only

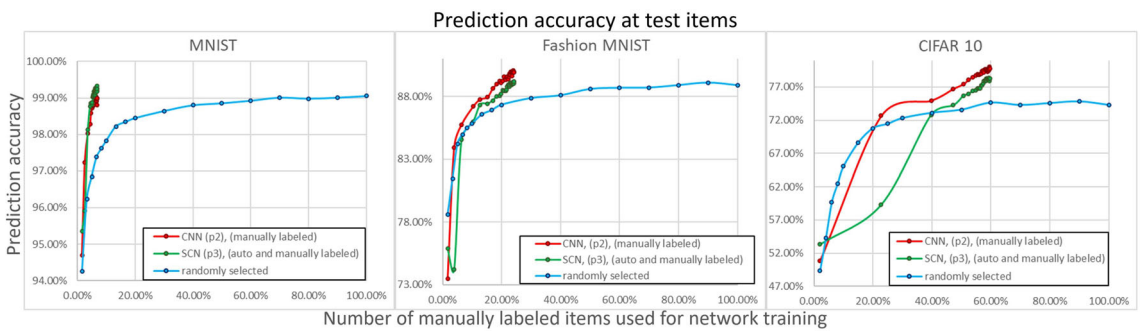

Fig. 7 Accuracy of predictions of items according to the number of items used for training neural network

Table 2 Accuracy and number of errors at test (MNIST, Fashion MNIST and CIFAR-10)

	MNIST				Fashion MNIST				CIFAR-10			
	Iter	Items	Accuracy	Errors	Iter	Items	Accuracy	Errors	Iter	Items	Accuracy	Errors
CNN, (manually labeled)	**17**	**3667**	99.06%	94	**28**	**14353**	**90.07%**	**993**	26	297 42	**80.01%**	**1999**
SCN, (automatically & manually)	31	4068	**99.33%**	**67**	31	14538	89.20%	1080	**25**	**29704**	78.41%	2159
All labeled	1	60000	99.05%	94.8	1	60000	88.89%	1110.2	1	50000	74.29%	2570.6

Note. For all three datasets, the columns of the table contain the following values: the number of subjects used for training, the prediction accuracy of the subjects from the test dataset and the number of incorrect predictions (errors) on 10,000 test items

items that are manually labeled. In the row "SCN, (automatically & manually)" we present results of the self-correcting network that are trained with all available items from training set (60,000 for MNIST, Fashion MNIST, and 50,000 for CIFAR-10); part of those items is manually labeled and remaining items are automatically labeled. In the row "all available items" we present results of network that are trained with all items from the dataset that are available for training (the best results in table are bolded).

Using the described algorithm, items from complete datasets were selected in several iterations, and used for training network, and this procedure was repeated until similar or better accuracy was achieved in relation to the results when a complete dataset was used for training. Based on the obtained results from MNIST dataset 6.11% (3667/60,000) items were selected, from Fashion MNIST 23.92% (14,353/60,000) items, and CIFAR-10 59.4% (29,704/50,000) items.

On all three graphs in Fig. 8, it can be noticed that after the first iteration the highest prediction accuracy is achieved with a neural network that is trained using manually labeled items. In the subsequent iterations, the prediction accuracy of manually labeled objects is lower than in the first iteration. Considering that in each subsequent iteration we select items that the network did not successfully recognize, and those items are manually labeled by a human expert, and added to the dataset of manually labeled items. In this way, after a few iterations, we identify the most problematic cases from the training set. It can be noticed that the accuracy in the predictions of items that are automatically labeled is higher compared to the items from the test set, and especially comparing to manually labeled items. In all three figures the accuracy on the automatically labeled items reached almost 100% after a few sessions, even on CIFAR-10 which is way more complex than the other two datasets.

The subplots in Fig. 9 show the number of the mismatched items, identified after comparing datasets (**d3**) and (**d4**), dataset (**d3**) used for a training network (**p3**), and dataset (**d4**) that represent predictions of network (**p3**). Once the differences are determined, a human expert reviews such items and determines which of these two values is incorrect. Those manually labeled items are written in dataset (**d5**) and appended to a dataset (**d2**) containing manually labeled items. The ALSCN algorithm identifies most of the problematic items in the first few iterations, and later the number of such items decreases.

In items that are manually reviewed, the percentage of detected incorrectly labeled items in (**d3**) are presented in Fig. 10. The number of items that the ALSCN algorithm selects for manual labeling decreases over iterations (Fig. 9); also the percentage of items that are incorrectly labeled in dataset (**d3**) decreases (Fig. 10).

All the experiments are conducted on a common desktop PC with Intel Core i9-9900 K 3.60Ghz CPU and a Nvidia GeForce RTX 2070 GPU. Experiments with MNIST dataset

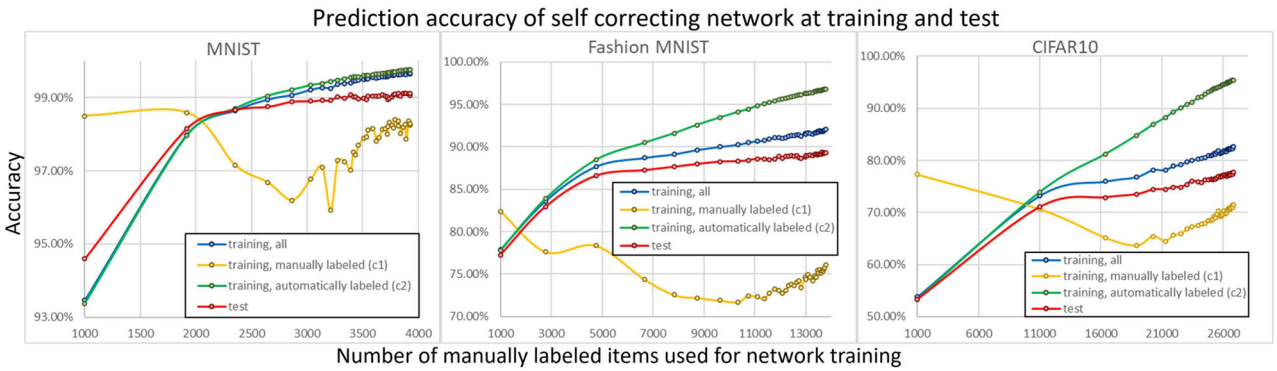

Fig. 8 Comparing accuracy of items per number of items used for training neural network. Note. The blue line represents the prediction accuracy for all items from the training set, the red line represents the prediction accuracy for items from the test subset. The training subset can be divided into two categories: manually labeled items and automatically labeled items. The prediction accuracy for manually labeled items is represented by a yellow line, and the prediction accuracy for automatically labeled items is represented by a green line

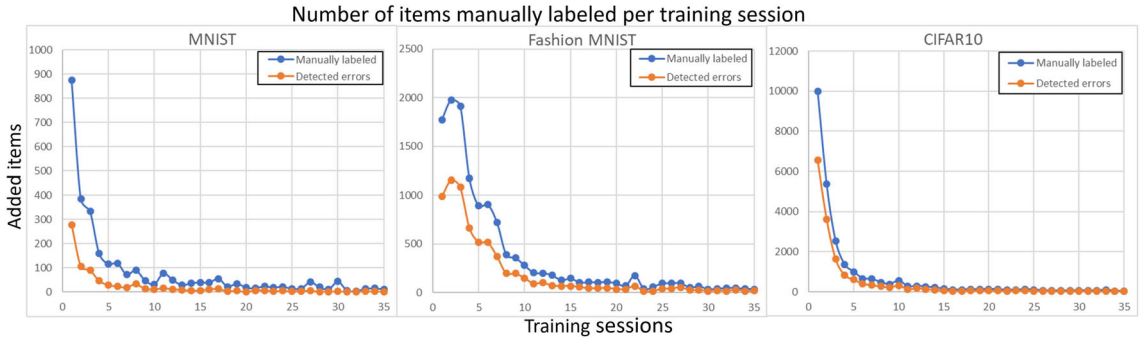

Fig. 9 Number of items selected for manual labeling per training session. Note. Blue line "Manually labeled" represent the number of items where a difference has been identified between (**d3**) and (**d4**). The number of automatically labeled items having the incorrect value at (**d3**) are presented with orange line

and 35 iterations take more than 6.5 h (approximately 5-10 min per iteration), for Fashion MNIST and 38 iterations it takes 9.5 h (approximately 10-25 min per iteration) and for CIFAR-10 and 35 iterations it takes 45 h (approximately 1–1.5 h per iteration) for the completion. Experiments with CIFAR-10 are slower than MNIST and Fashion MNIST because image augmentation was used with CIFAR-10 dataset, i.e. for each available image from the dataset, an additional four images were created by shifting the original image by a couple of pixels left, right, up, or down.

6 Applying the ALSCN algorithm for detecting errors in manually labeled items

The ALSCN algorithm is primarily used to reduce the number of items that need to be manually labeled, but it can also be used to identify errors in manually labeled data. In a real-life it often happens that error appears in items that are manually labeled. To identify these errors, we can check labels predicted with networks (**p2**) and (**p3**) for manually labeled items. All items where differences are identified in the group of manually labeled items (**e1**), and items that are automatically labeled (**e2**) should be sent to another round of manual labeling.

Previous experiments used publicly available and well-known datasets, so we can assume that the items in these datasets are correctly labeled. So, for purpose of demonstration of detecting errors in manually labeled items following experiment is conducted in which in first step where group of items are selected for manual labeling ((**p1**) at Figs. 4 and 5), for 20% of those items are intentionally assigned wrong labels (200/1000). Table 3 shows the number of detected items with wrong labels at manually labeled items. The ALSCN algorithm detects most mislabeled items in the first few iterations.

7 A comparison of the presented ALSCN method with other active learning methods

Active learning method called Query-by-committee (QBC) [22] can be described by following steps:

1. Manual labeling of the initial training set (in this case 1000 items, same as in presented method),
2. Train several neural networks with manually labeled items, (in this case three networks with identical structure as networks presented in this research, see Table 1),
3. Predict labels for unlabeled items using trained neural networks,
4. Compare predicted labels,
5. Manual labeling of the items with discrepant predictions,

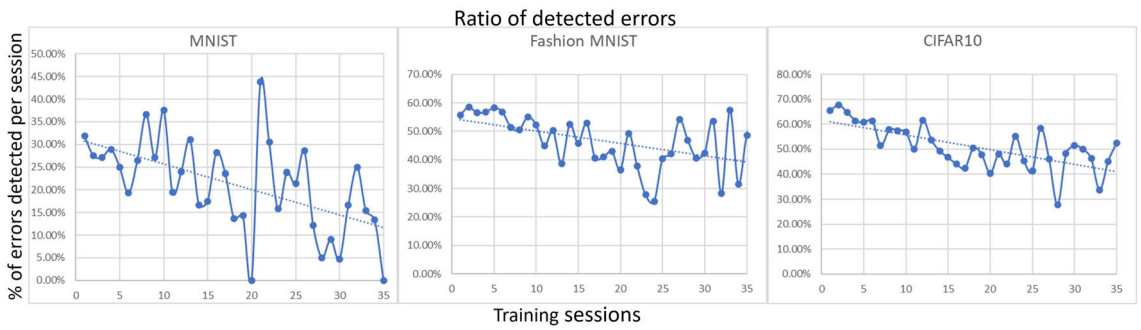

Fig. 10 Ratio of detected errors in items selected for manual labeling per training session

Table. 3 Number of detected incorrectly labeled items

		MNIST			Fashion MNIST			CIFAR-10		
Iter.	Network	New det.	Total	% Detected	New det.	Total	% Detected	New det.	Total	% Detected
1	P2	194	194	97.0%	185	185	92.5%	0	0	0.0%
1	P3	2	196	98.0%	9	194	97.0%	110	110	55.0%
2	P2	2	198	99.0%	0	198	99.0%	16	126	63.0%
2	P3	0	198	99.0%	1	198	99.0%	5	131	65.5%
3	P2	1	199	99.5%	0	199	99.5%	1	132	66.0%
3	P3	0	199	99.5%	0	199	99.5%	0	132	66.0%
4	P2	0	199	99.5%	0	199	99.5%	2	134	67.0%
4	P3	0	199	99.5%	0	199	99.5%	0	134	67.0%

Note. Initial dataset contains 1000 items, 200 of these items are intentionally incorrectly labeled. The table contains the following columns: (Iter.) iteration, name of network, (New det.) number of incorrectly labeled items detected that after comparing items before and after training networks, (Total) total number of incorrectly labeled items, and (%Detected) percent of detected incorrectly labeled items

6. Add manually labeled items from step 5 to previously labeled items,
7. Repeat sequence, starting from 2.

Active learning methods based on uncertainty sampling (margin sampling and entropy sampling) can be described by following steps:

1. Manual labeling of the initial training set (in this case 1000 items, same as in presented method),
2. Train neural networks with manually labeled items, (networks architecture is same as in Table 1),
3. Predict labels for unlabeled items using trained neural networks,
4. Calculate confidence for predicted labels (using formulas for margin confidence (1) or entropy (2))
5. Sort items by confidence,
6. A number of items with highest uncertainty are selected for manual labeling (250 items per iteration for MNIST, 1000 items for MNIST fashion, and 1000 for CIFAR-10),
7. Manual labeling of the selected items,
8. Add manually labeled items from step 7 to previously labeled items,
9. Repeat sequence, starting from 2.

Margin confidence *mc(x)* for item x is calculated as differences in probabilities $P_\theta(y|\,x)$ that most likely label y_m and next likely y_n.

$$mc(x) = P_\theta(y_m|x) - P_\theta(y_n|x) \tag{1}$$

Entropy *H(x)* for item x is calculated as sum of probabilities for $P_\theta(y_i|\,x)$ for each label y_i of variable x.

$$H(x) = \sum P_\theta(y_i|x) \cdot log(P_\theta(y_i|x)) \tag{2}$$

To evaluate the efficiency of the ALSCN active learning method, additional scripts based on QBC, margin sampling, and entropy sampling were developed. These scripts use the same network architecture as it is presented in Table 1. All methods start with the initial set of 1000 labeled items. In Fig. 11 and Table 4 we present the accuracy of the prediction per number of manually labeled items. The ALSCN method is more efficient than QBC because it requires fewer items to be manually labeled to achieve the same or higher accuracy, compared to QBC method.

There is a similarity between the ALSCN and QBC methods as both methods try to find differences in the

Table 4 Comparison of prediction accuracy between ALSCN and other active learning methods

	MNIST			Fashion MNIST			CIFAR-10		
	Iter	Items	Accuracy	Iter	Items	Accuracy	Iter	Items	Accuracy
ALSCN	17	**3667**	99.06%	28	**14353**	90.07%	26	**29742**	**80.01%**
QBC	**10**	6500	99.13%	**17**	19197	88.95%	**11**	36496	78.79%
Margin	28	7750	**99.17%**	25	25000	**90.22%**	33	3300	79.15%
Entropy	25	7000	99.14%	20	20000	89.76%	37	37000	78.68%
All available data	1	60000	99.05%	1	60000	88.89%	1	50000	74.29%

predicted data. The QBC method determines the candidates for manual labeling by comparing the predictions of several neural networks. These networks are trained on the same dataset, and they are used to determine predictions for all available data. The ALSCN method compares the data used for training with the predicted labels for the same items used for training. The ALSCN method uses two networks, one is trained only with manually labeled items, other "self-correcting network" is trained with all available items from dataset (some of those items are manually labeled and for remaining items labels are predicted using the previous network).

In active learning methods, such as margin sampling and entropy sampling, items are sorted by selected functions, so it is always known which items have a priority for manual labeling. However, estimating how many items should be selected for manual labeling in each iteration it is not a clear task. The opposite is the ALSCN method that focuses on detecting the difference between datasets (**d3**) and (**d4**). In this approach it is known how many items should be selected for manual labeling. However, since in this approach the items are not ranked, any item where a difference in the prediction has been identified is equally important and limiting the number of items that will be verified in one iteration is not the best practice as all of them should be verified. If we omit the opportunity to verify some items in one iteration, there is no guaranties that this item will be detected in the next iteration.

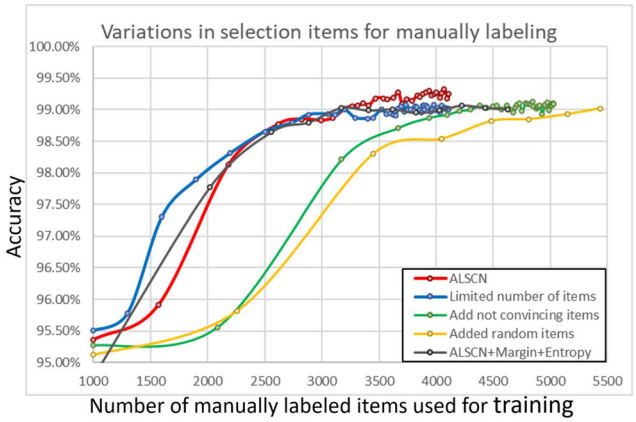

Fig. 12 Variations of the ALSCN algorithm

8 Variations of the ALSCN algorithm

During the development of the ALSCN active learning method, we experimented with a few other approaches that were found to be less effective. The results of those experiments are presented in Fig. 12. Briefly, we focused on:

- Limitation of the number of items selected for manual labeling per session (blue line); without such a limitation, this algorithm in the first iterations identifies a larger number of items that need to by manually labeled; in latter iterations the number of such items decreases rapidly. The assumption was that if a network would be trained more often, the predictions could be more accurate and that would lead to a reduced number of items that need to be manually labeled. Experiments show that better results are achieved without such limitations.
- Adding "unconvincing" items (green line), (the term "unconvincing" item in this context refers to those items in which the predictions are not wrong, but predicted values are very close to the limit and it could be easily classified differently). Experiments showed that the number of predicted labels is not large and that inclusion of those items to the list of items that should be manually labeled did not significantly improve results.
- Beside the items that are selected by the algorithm, certain number of items is randomly selected for manual labeling (yellow line). The assumption was that in this way some additional items could increase accuracy of the trained model and perhaps it could assist in discovering some incorrectly labeled items, hard to be detected by a trained model. Experiments show that such randomly selected items do not significantly contribute to accuracy increase, especially because in the pool of automatically labeled items the number of errors decreases much faster comparing to the error rate in test subset (see Fig. 8).

Fig. 11 Comparison of prediction accuracy between the ALSCN and other active learning methods

- Two optimizers were tried: AdaDelta and Adam. Adam optimizer resulted in more accurate results.
- Gray line represents combination of methods ALSCN, margin and entropy sampling. The ALSCN method selects items for manual labeling as it is previously described, in addition to these items, another 100 items are selected using margin and entropy sampling methods, and each of these two methods selects 50 items.
- Red line represents the prediction accuracy of the ALSCN method presented in this research.

The ALSCN algorithm can work with any network configuration, but selected configuration could affect the number of items that will be selected for manual labeling per one iteration, also number of iterations. The results of the ALSCN algorithm at MNIST dataset with different network configurations are presented in Fig. 13:

- Network1: inputs 28×28×1, (16,32,32)×3×3 convolutional layers, (64,64,32) fully connected and 10 output,
- Network2: (model used in this paper) inputs 28×28×1, (16,32,64)×3×3 convolutional layers, (128,64,32) fully connected and 10 output,
- Network3: inputs 28×28×1, (16,32,128)×3×3 convolutional layers, (256,64,32) fully connected and 10 output,
- Network4: inputs 28×28×1, (32,64,128)×3×3 convolutional layers, (256,128,64) fully connected and 10 output.

The ALSCN algorithm, with a more complex network, selects fewer items for manual labeling at first iterations compared to experiments where it uses networks with fewer parameters. If we compare the performance of four different networks trained on datasets that contain fewer than 2000 items selected by the ALSCN algorithm (from the MNIST dataset), the more complex networks achieve better accuracy. In addition, the more complex networks need more iterations to select these items. However, if we compare their performance after 3000 selected items, the accuracy of the predicted results becomes similar.

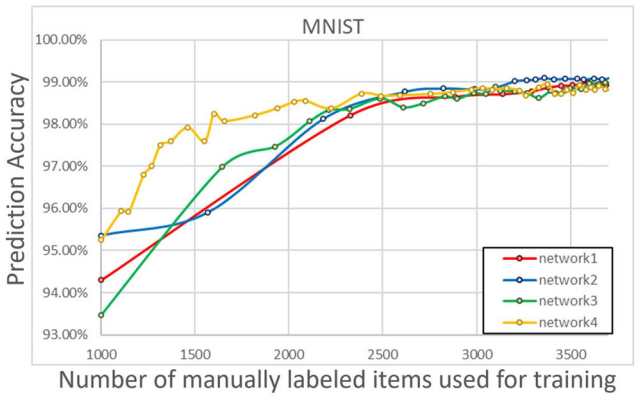

Fig. 13 The ALSCN method with different network configurations

9 Conclusion

The main idea behind active learning methods is that items from the training set do not have equal contributions to a classifier. The ALSCN active learning framework presented in this study enables us to iteratively select the best candidates for manual labeling (the most informative instances) from a pool of unlabeled items. This approach shows best results where there is a large amount of unlabeled data available, and it is especially useful if manual labeling is expensive or time-consuming. Another significant advantage of active learning is that in this approach, the dataset is allowed to be significantly expanded in size by adding unlabeled data, without substantially increasing the effort of human labelers in labeling these additional items.

The ALSCN framework can efficiently find difficult examples in a set of unlabeled (automatically labeled) items in all three datasets, and items remaining in a set of automatically labeled items represent good candidates for items that do not need to be manually labeled. Also, this approach has a lot of potential in finding errors in previously labeled data.

Our experiments show that the network trained using items selected by the ALSCN method exceeds the performance of a network trained with the same number of items randomly selected from the set of available items. An interesting observation is that the performance of a network trained with selected items can achieve even better accuracy than a network trained with all items from datasets, and this was shown on all three datasets.

Our ALSCN solution belongs to the group of pool-based sampling scenarios active learning and our query strategy exhibits similarities to query-by-committee (QBC) and uncertainty sampling. The approach is located somewhere between unsupervised and supervised learning. Supervised learning can be very efficient for detecting false or inaccurate prior knowledge, where solutions delivered by unsupervised methods depend on the encoded prior knowledge up to a large extent. The first network in this framework is trained following the supervised learning approach, and CNN network is trained by a small amount of manually labeled items. The second network is trained by an unsupervised learning approach; “self-correcting network" is trained with much larger dataset that contains all available items that are automatically labeled, and after training that network predicts new labels for the same items that are used in training. With simple

comparison of labels from these two datasets the items that need to be manually labeled can be identified.

In this paper we presented examples where the ALSCN algorithm (containing two convolutional neural networks) is applied on three datasets for the purpose of classification. The future work could explore the efficiency of the proposed approach with different types of networks and different domains, e.g. object detection, semantic segmentation, NLP, time series data, etc.

References

1. Wang D (2013) Active labeling in deep learning and its application to emotion prediction." PhD diss., University of Missouri–Columbia
2. Sener O, Savarese S (2017) Active learning for convolutional neural networks: a core-set approach, International Conference on Learning Representations (ICLR)
3. Kim T, Lee K, Ham S, Park B, Lee S, Hong D, Kim GB, Kyung YS, Kim CS, Kim N (2020) Active learning for accuracy enhancement of semantic segmentation with CNN-corrected label curations: Evaluation on kidney segmentation in abdominal CT, Scientific reports, (Nature Publisher Group), 10(1), pp.1–7
4. Fortuny EJ, Martens D, Provost F (2013) Predictive modeling with big data: is bigger really better? Big Data 1(4):215–226
5. Gao M, Zhang Z, Yu G, ArÄ±k SÃ, Davis LS, Pfister T (2020) Consistency-based semi-supervised active learning: Towards minimizing labeling cost, In European Conference on Computer Vision (pp. 510–526). Springer, Cham
6. Zhu X, Vondrick C, Ramanan D, Fowlkes CC (2012) Do We Need More Training Data or Better Models for Object Detection?, In BMVC (Vol. 3, No. 5)
7. Neutatz F, Mahdavi M, Abedjan Z (2019) Ed2: a case for active learning in error detection, In Proceedings of the 28th ACM international conference on information and knowledge management (pp. 2249–2252)
8. Bekker AJ, Goldberger J (2016) Training deep neural-networks based on unreliable labels, In 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 2682–2686). IEEE
9. Nettleton D, Orriols-Puig A, Fornells A (2010) A study of the effect of different types of noise on the precision of supervised learning techniques, Artificial intelligence review
10. Liu X, Li S, Kan M, Shan S, Chen X (2017) Self-error-correcting convolutional neural network for learning with noisy labels, In 2017 12th IEEE international conference on Automatic Face & Gesture Recognition (FG 2017) (pp. 111–117). IEEE
11. Grandvalet Y, Bengio Y (2005) Semi-supervised learning by entropy minimization, In CAP (pp. 529–536)
12. Fazakis N, Vasileios GK, Christos KA, Stamatis K, Sotiris K (2019) Combination of Active Learning and Semi-Supervised Learning under a Self-Training Scheme, Entropy 21, no. 10, 988
13. Chen T, Kornblith S, Swersky K, Norouzi M, Hinton G (2020) Big self-supervised models are strong semi-supervised learners, advances in neural information processing systems, (NeurIPS 2020)
14. Goudjil M, Koudil M, Bedda M, Ghoggali N (2018) A novel active learning method using SVM for text classification. Int J Autom Comput 15(3):290–298
15. Chen T, Kornblith S, Norouzi M, Hinton G (2020) A simple framework for contrastive learning of visual representations, proceedings of the 37th international conference on machine learning. PMLR 119:1597–1607
16. Mnih V, Hinton GE (2012) Learning to label aerial images from noisy data, In Proceedings of the 29th international conference on machine learning (ICML-12) (pp. 567–574)
17. Seeger M (2001) Learning with labeled and unlabeled data, Institute for Adaptive and Neural Computation, University of Edinburg
18. Settles B (2009) Active learning literature survey, University of Wisconsin-Madison Department of Computer Sciences
19. Yang Y, Ma Z, Nie F, Chang X, Hauptmann AG (2015) Multi-class active learning by uncertainty sampling with diversity maximization. Int J Comput Vis 113(2):113–127
20. Sharma M, Bilgic M (2017) Evidence-based uncertainty sampling for active learning. Data Min Knowl Disc 31(1):164–202
21. Zhou J, Sun S (2014) Improved margin sampling for active learning, In Chinese Conference on Pattern Recognition. Springer, Berlin, pp 120–129
22. Grimova N, Macas M (2019) Query-By-Committee Framework Used for Semi-Automatic Sleep Stages Classification, In Multidisciplinary Digital Publishing Institute Proceedings (Vol. 31, No. 1, p. 80)
23. Wang K, Zhang D, Li Y, Zhang R, Lin L (2016) Cost-effective active learning for deep image classification. IEEE Trans Circuits Syst Video Technol 27(12):2591–2600
24. Wei K, Iyer R, Bilmes J (2015) Submodularity in data subset selection and active learning, In International conference on machine learning (pp. 1954–1963)
25. Joshiy AJ, Porikli F, Papanikolopoulos N (2010) Multi-class batch-mode active learning for image classification, In 2010 IEEE international conference on robotics and automation (pp. 1873–1878). IEEE
26. Jakramate B, Kab'an A (2012) Label-noise robust logistic regression and its applications, in Machine Learning and Knowledge Discovery in Databases, pp. 143–158

Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH (“Springer Nature”).
Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users (“Users”), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use (“Terms”). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.
These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.
We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.
While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.
These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.
To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.
Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.
If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com