CTypiClust: Confidence-Aware Typical Clustering for Budget-Agnostic Active Learning with Confidence Calibration

Takuya Okano[a], Yohei Minekawa[b] and Miki Hayakawa[c]
Hitachi High-Tech Corporation, Japan

Keywords: Deep Learning, Active Learning, Confidence Calibration.

Abstract: Active Learning (AL) has been widely studied to reduce annotation costs in deep learning. In AL, the appropriate method varies depending on the number of annotatable data (budget). In low-budget settings, it is appropriate to prioritize sampling typical data, while in high-budget settings, it is better to prioritize sampling data with high uncertainty. This study proposes Confidence-aware Typical Clustering (CTypiClust), an AL method that performs well regardless of the budget. CTypiClust dynamically switches between typical data sampling and low-confidence data sampling based on confidence. Additionally, to mitigate the overconfidence problem in low-budget settings, we propose a new confidence calibration method Cluster-Enhanced Confidence (CEC). By applying CEC to CTypiClust, we suppress the occurrence of overconfidence in low-budget settings. To evaluate the effectiveness of the proposed method, we conducted experiments using multiple benchmark datasets, and confirmed that CTypiClust consistently shows high performance regardless of the budget.

1 INTRODUCTION

Reducing annotation costs is one of the critical challenges in deep learning. To enhance the performance of deep learning models, a large amount of data is required, but annotating all the data is very costly. This problem is particularly severe in fields requiring expertise, such as manufacturing and healthcare, where accurate labeling demands enormous costs and time.

Active Learning (AL) has been widely studied as a method to minimize annotation costs (Ren et al., 2021). In AL, a fixed budget of data is sampled from a large pool of unlabeled data, based on its usefulness for improving model performance. This process is repeated to optimize the model with minimal labeled data. Traditional methods include AL methods using uncertainty (low confidence) (Roth and Small, 2006; Gal et al., 2017; Pop and Fulop, 2018), methods considering data diversity (Sener and Savarese, 2018; Zhdanov, 2019), and methods considering both uncertainty and diversity (Sinha et al., 2019). These methods assumed relatively high-budget settings, but recent advances in self-supervised learning (Jaiswal et al., 2021) have led to the development of AL methods in cold-start settings with no labeled data (Chen et al., 2023; Yi et al., 2022), and research on AL in low-budget settings (Hacohen et al., 2022).

However, (Hacohen et al., 2022) shows that the optimal method varies depending on the budget. In low-budget settings, it is necessary to learn from limited data, so prioritizing typical data makes it easier to capture the overall characteristics of the dataset, leading to faster model accuracy improvement. On the other hand, in high-budget settings, the main features of the dataset can be learned from a large amount of data, so learning data with high uncertainty near the decision boundary is effective for improving accuracy (Hacohen et al., 2022).

Therefore, in this study, we propose Confidence-aware Typical Clustering (CTypiClust), which performs highly regardless of the budget. This method extends Typical Clustering(TypiClust)(Hacohen et al., 2022), a method for low-budget settings, by dynamically switching between sampling typical data and low-confidence data based on confidence, making it effective in high-budget settings as well. Additionally, to address the issue of overconfidence in low-budget settings, we propose a new confidence calibration method Cluster-Enhanced Confidence(CEC) and apply it to CTypiClust. To

[a] https://orcid.org/0009-0006-3749-3883
[b] https://orcid.org/0009-0006-8980-4356
[c] https://orcid.org/0009-0004-3547-8896

Okano, T., Minekawa, Y. and Hayakawa, M.
CTypiClust: Confidence-Aware Typical Clustering for Budget-Agnostic Active Learning with Confidence Calibration.
DOI: 10.5220/0013139400003912
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2025) - Volume 2: VISAPP*, pages 340-347
ISBN: 978-989-758-728-3; ISSN: 2184-4321
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

confirm that CTypiClust performs well regardless of the budget, we evaluate its effectiveness using CIFAR-10, CIFAR-100, and STL-10.

The contributions of this paper are as follows:

1. We propose an AL method CTypiClust that consistently demonstrates high performance regardless of budget constraints.
2. A confidence calibration method CEC is proposed, which effectively mitigates overconfidence even in low-budget settings.
3. The effectiveness of CTypiClust and CEC is demonstrated through experiments using multiple benchmark datasets.

2 RELATED WORK

2.1 Active Learning

High-Budget Active Learning. Many AL methods typically assume a high-budget setting, where methods that sample data with high uncertainty (Roth and Small, 2006; Gal et al., 2017; Pop and Fulop, 2018), methods that consider diversity (Sener and Savarese, 2018; Zhdanov, 2019), and methods that consider both uncertainty and diversity have been proposed (Kirsch et al., 2019; Sinha et al., 2019).

Among these, methods that sample data with high uncertainty have been widely proposed (Roth and Small, 2006; Gal et al., 2017; Pop and Fulop, 2018). (Roth and Small, 2006) proposed Margin, which prioritizes sampling data with a small difference between the highest and second-highest predicted probabilities, considering such data as having high uncertainty. (Gal et al., 2017) proposed DBAL, which utilizes a Bayesian approach.

(Sener and Savarese, 2018) proposed CoreSet, an AL method that prioritizes diversity. CoreSet achieves diversity-aware sampling by sampling representative data based on the core-set approach. (Zhdanov, 2019) proposed a mini-batch active learning method that incorporates data diversity using K-means clustering to enhance the efficiency of label selection in large-scale datasets.

Approaches that consider both uncertainty and diversity have also been proposed (Kirsch et al., 2019; Sinha et al., 2019). (Kirsch et al., 2019) proposed BALD, which balances uncertainty and diversity by sampling to maximize mutual information among data points within a batch, in addition to a Bayesian approach. (Sinha et al., 2019) introduced VAAL, a method that focuses on both diversity and uncertainty using a variational autoencoder.

Cold and Low-Budget Active Learning. In settings like cold-start and low-budget, where there is little or no labeled data, methods effective in high-budget settings perform worse than random sampling (Hacohen et al., 2022). As an effective method in cold-start settings, (Yi et al., 2022) introduced a method that samples data with high loss in pretext tasks, considering it to have high learning efficiency. (Chen et al., 2023) proposed a method based on contrastive learning that samples data that is difficult to distinguish as typical data. In cold-start and low-budget settings, (Hacohen et al., 2022) proposed TypiClust. TypiClust prioritizes sampling data with high density in the feature space of unlabeled data as typical data.

Typicality-prioritized AL methods like TypiClust perform poorly in high-budget settings. Therefore, in this study, we propose CTypiClust, which performs highly regardless of the budget by combining methods based on uncertainty. Uncertainty is generally calculated from the model's confidence, but in low-budget settings, there is a problem of overconfidence, where the model becomes excessively confident.

2.2 Confidence Calibration

Deep learning models are known to exhibit overconfidence, where the predicted confidence significantly exceeds the actual accuracy. This issue is particularly prevalent in low-budget settings with very limited training data. To address this problem, numerous calibration methods have been proposed to align predicted probabilities with actual accuracies (Wang, 2024).

Calibration methods can be broadly categorized into post-hoc methods and regularization methods. Post-hoc methods perform calibration using a large amount of data after the model has been trained. For example, temperature scaling (Platt, 2000; Mozafari et al., 2019) optimizes the temperature parameter of the softmax function using validation data. On the other hand, regularization methods add penalties to the model's loss function to suppress overconfidence (Guo et al., 2017; Pereyra et al., 2017).

Many calibration methods assume settings with a large amount of labeled data or aim to improve the model itself or the loss function. However, they do not consider special settings like low-budget, where labeled data is extremely limited.

In this study, we propose a new calibration method CEC to mitigate overconfidence in low-budget settings and apply it to CTypiClust. CEC corrects the model's confidence based on the clustering results of intermediate layer features.

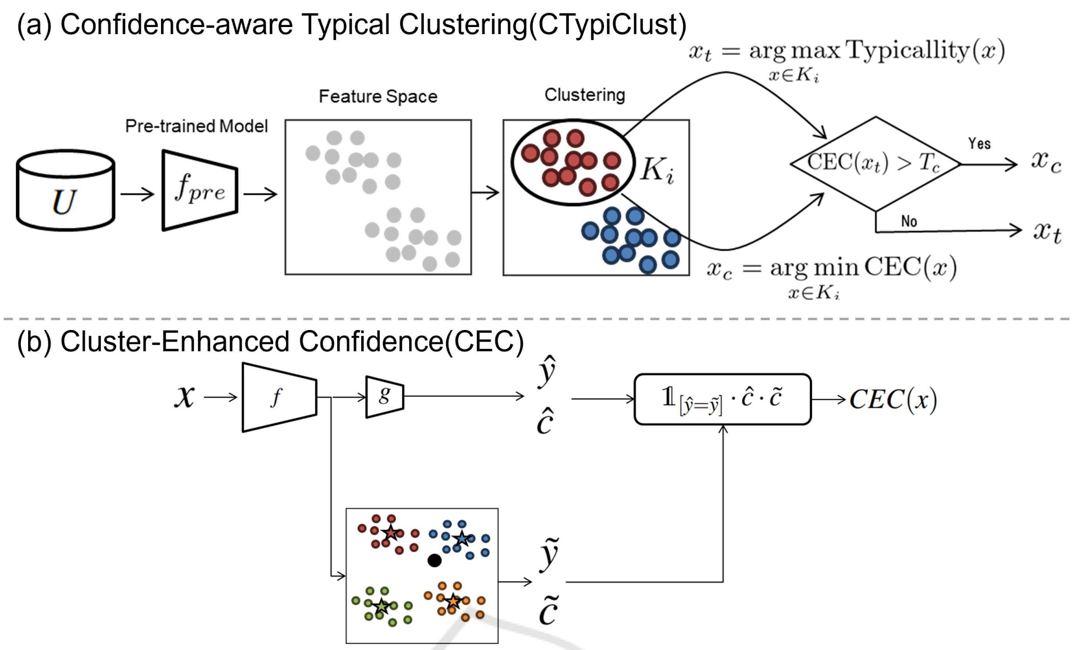

(a) Confidence-aware Typical Clustering(CTypiClust)
U
Pre-trained Model
f_{pre}
Feature Space
Clustering
K_i
$x_t = \arg\max_{x \in K_i} \text{Typicallity}(x)$
$x_c = \arg\min_{x \in K_i} \text{CEC}(x)$
$\text{CEC}(x_t) > T_c$
Yes
No
x_c
x_t
(b) Cluster-Enhanced Confidence(CEC)
x
f
g
$\hat{y}$
$\hat{c}$
$\mathbb{1}_{[\hat{y}=\tilde{y}]} \cdot \hat{c} \cdot \tilde{c}$
$CEC(x)$
$\tilde{y}$
$\tilde{c}$

Figure 1: Illustration of CTypiClust and CEC. (a) CTypiClust, similarly to TypiClust, obtains features from the unlabeled dataset U and performs clustering. From each cluster K_i, it retrieves the data with the highest typicality x_t and the data with the lowest CEC x_c. It decides whether to sample x_t or x_c based on the CEC(x_t) of the data with the highest typicality. (b) In CEC, the input data x is fed into the model to obtain the features $f(x)$ (black circle) and the output of the classification model $g(y \mid f(x))$. From $g(y \mid f(x))$, the pseudo-label $\hat{y}$ and the confidence $\hat{c}$ are calculated. The features of the unlabeled data are clustered, and the cluster to which x belongs is assigned the pseudo-label $\tilde{y}$. The confidence $\tilde{c}$ is calculated based on the relative distance to the center μ_i (star) of each cluster. Finally, CEC(x) is calculated from the two pseudo-labels and the confidence.

3 METHOD

In this section, we introduce Confidence-aware Typical Clustering (CTypiClust). The detailed methodology of CTypiClust is explained in Section 3.2. Additionally, we propose a new confidence calibration method called Cluster-Enhanced Confidence (CEC), which is used in CTypiClust and is discussed in Section 3.3. The methods are illustrated in Figure 1.

3.1 Notation

Let X be the set of all input data, and each data point $x \in X$ is included in the unlabeled dataset $U \subseteq X$. Although the data in the unlabeled dataset U are not labeled, there exists a set of class labels $Y = \{1, 2, \ldots, |Y|\}$ corresponding to the data. Each data point x has a corresponding label $y \in Y$. The model used in this study is divided into a feature extractor $f(\cdot)$ and a classifier $g(\cdot)$. First, the feature extractor f extracts features $f(x)$ from the input x. The classifier g takes the features $f(x)$ as input and outputs the probability distribution $g(y \mid f(x))$ for the label y.

3.2 Confidence-Aware Typical Clustering

We propose CTypiClust, an extension of TypiClust (Hacohen et al., 2022) that considers confidence. While TypiClust samples data with high typicality as is, CTypiClust determines whether to sample data with high typicality or low-confidence data based on the confidence of the data with high typicality. If the confidence of the data with high typicality is high, CTypiClust assumes that the learning efficiency of typical data is low and samples low-confidence data. As a result, in immature stages like low-budget settings, typicality-prioritized sampling is expected, while in mature stages like high-budget settings, low-confidence-prioritized sampling is expected. Additionally, CTypiClust uses CEC as the confidence measure to mitigate overconfidence in low-budget settings.

The specific steps of CTypiClust are explained below. CTypiClust consists of four steps. Steps 1 and 2 are the same as in TypiClust.

Step1: Pre-train the model f using the unlabeled data U with Self-Supervised Learning methods (e.g., Sim-

CLR (Chen et al., 2020)).

Step2: Input the unlabeled data U into the model f_{pre} trained in Step 1 to obtain features. Perform clustering on the obtained features using a method such as K-means.

Step3: Extract data with high typicality $x_t = \arg\max_{x \in K_i}\{\text{Typicality}(x)\}$ and data with low CEC $x_c = \arg\min_{x \in K_i}\{\text{CEC}(x)\}$ from each cluster K_i. The calculation method of CEC is explained in Section 3.3. Typicality(x) is calculated as the local density in the feature space of x, as in (Hacohen et al., 2022). Specifically, it is defined by the following equation:

$$\text{Typicality}(x) = \left(\frac{1}{\mathcal{K}} \sum_{x_i \in \mathcal{K}\text{-NN}(x)} \|x - x_i\|_2 \right)^{-1}.$$

Here, $\mathcal{K}$ is the number of data points in the $\mathcal{K}$-nearest neighbors ($\mathcal{K}$-NN) of x, and x_i is one of the data points in the neighborhood. $\|x - x_i\|_2$ is the Euclidean distance between the data point x and its neighboring point x_i.

Step4: If the CEC(x_t) of the data with high typicality x_t is below the threshold T_c, sample x_t as is. If CEC(x_t) is higher than the threshold T_c, sample the data x_c with the lowest CEC in the same cluster.

The algorithm of CTypiClust is shown in Algorithm 1.

```
Data: Unlabeled pool $U$, Budget $B$
Result: Queries
Embedding ← Representation_Learning($U$);
Clust ← Clustering_algorithm(Embedding, $B$);
Queries ← $\emptyset$;
for $i = 1$ to $B$ do
    $x_t \leftarrow \arg\max_{x \in K_i}\{\text{Typicality}(x)\}$ ;
    $x_c \leftarrow \arg\min_{x \in K_i}\{\text{CEC}(x)\}$ ;
    if $CEC(x_t) > T_c$ then
        Add $x_c$ to Queries ;
    else
        Add $x_t$ to Queries ;
    end
end
```

Algorithm 1: CTypiClust.

3.3 Cluster-Enhanced Confidence

In low-budget settings, where the amount of training data is limited, the model tends to overfit and become overconfident. To mitigate overconfidence, we propose a confidence calibration method CEC, which corrects the confidence of the classification model's output using the clustering results of intermediate layer features.

The pseudo-label obtained from the classifier g is $\hat{y} = \arg\max_y g(y \mid f(x))$, and the confidence is $\hat{c} = \max_y g(y \mid f(x))$. Additionally, the features of the unlabeled data U are clustered, and the center of each cluster K_i is $\mu_i = \frac{1}{|K_i|}\sum_{x \in K_i} f(x)$. The pseudo-label $\tilde{y}$ is the label obtained from clustering. In K-means, $\tilde{y} = \arg\min_i D(x, \mu_i)$. D represents any distance function (e.g., Euclidean distance, Cosine similarity). The correspondence between the model's pseudo-label $\hat{y}$ and the clustering pseudo-label $\tilde{y}$ is based on the frequency of label occurrence within each cluster. Specifically, after each data x is assigned a pseudo-label $\tilde{y}$ by the clustering method, the model's pseudo-label $\hat{y}$ that appears most frequently within each cluster K_i is assigned as the representative label of that cluster. The confidence $\tilde{c}$ is calculated based on the relative distance to the center of each cluster, similar to Prototypical Networks (Snell et al., 2017).

$$\tilde{c} = \max_k \frac{\exp(-D(f(x), \mu_k))}{\sum_i \exp(-D(f(x), \mu_i))}.$$

By using these values, CEC(x) is defined as follows:

$$\text{CEC}(x) = \mathbb{1}_{[\hat{y}=\tilde{y}]} \cdot \hat{c} \cdot \tilde{c}. \quad (1)$$

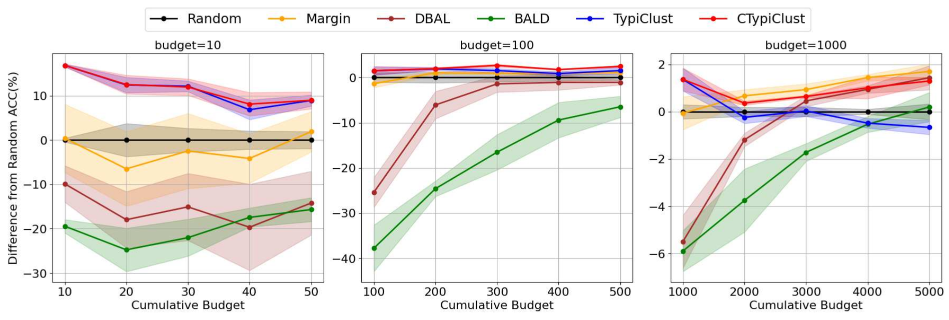
Here, $\mathbb{1}_{[\hat{y}=\tilde{y}]}$ is an indicator function that returns 1 if the two pseudo-labels match and 0 if they do not. The number of clusters $|K|$ is set to be equal to the number of classes $|Y|$. This function ensures that if the labels do not match, CEC becomes 0, and unless both confidences are high, the confidence will not be high. The algorithm of CEC is shown in Algorithm 2.

```
Data: Data $x$, Clust $K$, Models $f, g$
Result: CEC($x$)
for $i = 1$ to $|K|$ do
    $\mu_i \leftarrow \frac{1}{|K_i|}\sum_{x \in K_i} f(x)$ ;
end
$\hat{y} \leftarrow \arg\max_y g(y \mid f(x))$;
$\tilde{y} \leftarrow \arg\min_i D(x, \mu_i)$ ;
$\hat{c} \leftarrow \max_y g(y \mid f(x))$;
$\tilde{c} \leftarrow \max_k \frac{\exp(-D(f(x),\mu_k))}{\sum_i \exp(-D(f(x),\mu_i))}$;
CEC($x$) ← $\mathbb{1}_{[\hat{y}=\tilde{y}]} \cdot \hat{c} \cdot \tilde{c}$ ;
```

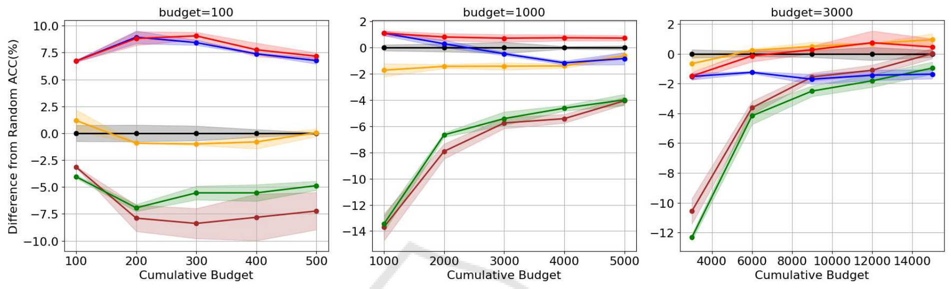
Algorithm 2: CEC.

4 EXPERIMENT AND DISCUSSION

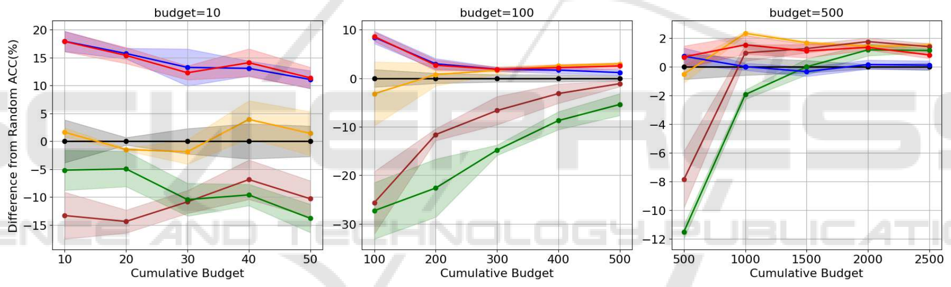
To verify whether CTypiClust performs superiorly regardless of the budget, we use multiple datasets and compare it with related methods under various budget settings. Additionally, we conduct an ablation study of CTypiClust and CEC using the CIFAR-10 dataset.



(a) CIFAR-10



(b) CIFAR-100



(c) STL-10

Figure 2: Comparison of the ACC difference between each method and Random for each dataset. Results are shown from left to right for low, medium, and high budgets. The shaded area reflects standard error.

4.1 Experimental Settings

In this experiment, we evaluate based on the AL program proposed by (Munjal et al., 2022). The datasets used for evaluation are CIFAR-10 (Krizhevsky and Hinton, 2009), CIFAR-100 (Krizhevsky and Hinton, 2009), and STL-10 (Coates et al., 2011). The comparison methods are TypiClust(Hacohen et al., 2022), Margin(Roth and Small, 2006), DBAL(Gal et al., 2017), BALD(Kirsch et al., 2019), and Random. We set three types of budgets (low, medium, and high) and configure them for each dataset as follows: low=10, medium=100, high=1000 for CIFAR-10; low=100, medium=1000, high=3000 for CIFAR-100; and low=10, medium=100, high=500 for STL-10. We use ResNet-18 (He et al., 2016) as the model. For TypiClust and CTypiClust, we use features extracted from models pre-trained with SimCLR (Chen et al., 2020). The models for learning each dataset are also pre-trained with SimCLR. The parameter T_c for CTypiClust is set to 0.8, and the distance function D is the Euclidean distance. The evaluation metrics are accuracy (ACC) and Area Under the Budget Curve (AUBC) (Zhan et al., 2021). AUBC is a metric that calculates the area under the ACC curve for each budget. Other detailed settings are described in the Appendix.

4.2 Performance Comparison of Different Methods

To evaluate whether CTypiClust performs highly regardless of the budget compared to other methods, we compare it with related methods under various budget settings for multiple datasets.

Table 1: Mean and standard deviation of AUBC for each method under low, medium, and high budgets. The numbers in parentheses indicate the budget size. The highest performance is shown in red, and the second highest in blue.

	CIFAR-10			CIFAR-100			STL-10		
Budget	Low(10)	Medium(100)	High(1000)	Low(100)	Medium(1000)	High(3000)	Low(10)	Medium(100)	High(500)
Random	49.77 (±1.67)	74.20 (±0.70)	82.56 (±0.12)	20.90 (±0.44)	44.83 (±0.13)	53.94 (±0.20)	41.88 (±1.98)	71.00 (±0.85)	83.67 (±0.42)
Margin	46.78 (±7.11)	74.89 (±0.05)	83.52 (±0.20)	20.37 (±0.16)	43.48 (±0.13)	54.33 (±0.18)	42.39 (±1.83)	72.21 (±1.66)	85.14 (±0.08)
DBAL	33.58 (±6.65)	68.75 (±1.97)	82.10 (±0.07)	13.58 (±1.38)	37.84 (±0.43)	51.05 (±0.26)	30.89 (±1.06)	62.30 (±1.82)	83.85 (±0.35)
BALD	29.32 (±2.86)	56.03 (±1.46)	80.34 (±0.27)	15.28 (±0.43)	38.48 (±0.16)	50.16 (±0.35)	33.24 (±2.45)	55.38 (±1.73)	82.17 (±0.14)
TypiClust	60.84 (±1.28)	75.66 (±0.26)	82.47 (±0.18)	28.77 (±0.15)	44.53 (±0.11)	52.48 (±0.19)	56.00 (±1.26)	73.83 (±0.49)	83.72 (±0.24)
CTypiClust	61.13 (±1.70)	76.29 (±0.22)	83.39 (±0.12)	29.04 (±0.24)	45.63 (±0.14)	54.03 (±0.42)	55.97 (±0.59)	74.07 (±0.28)	84.84 (±0.17)

Figure 2 compares each method with Random under different budgets for each dataset. TypiClust performs well when the budget is small, such as in low-budget settings, but it performs worse than Random as the budget increases. On the other hand, Margin, which samples data with high uncertainty, performs better than Random in high-budget settings but worse than Random in low-budget settings. The proposed method CTypiClust performs better than Random in most cases regardless of the budget.

Table 1 compares the results of each method in terms of AUBC. Similar to Figure 2, TypiClust performs well in low-budget settings but poorly in high-budget settings. Margin performs well in high-budget settings but relatively poorly in low-budget and medium-budget settings. CTypiClust ranks first or second in all budget settings, demonstrating high performance regardless of the budget.

4.3 Ablation Study

In this section, we conduct an ablation study on CTypiClust and CEC using the CIFAR-10 dataset. First, we compare CTypiClust using confidence $\hat{c}$ and CEC to evaluate the necessity of CEC in CTypiClust . Next, we assess whether CEC mitigates the issue of overconfidence. Finally, we examine the performance differences based on the parameter T_c in CTypiClust.

Comparison of CTypiClust Using Confidence $\hat{c}$ and CEC. To evaluate the necessity of CEC in CTypiClust, we compare the performance of CTypiClust using simple confidence $\hat{c}$ (w/o CEC) and CTypiClust using CEC (w/ CEC). Table 2 shows the AUBC of w/o CEC and w/ CEC. In low-budget settings, the AUBC of w/o CEC is 59.34%, that of w/ CEC is 61.13%, approximately 1.79% improvement by using CEC. In high-budget settings, the AUBC of w/o CEC is 83.43%, slightly better than the 83.39% of w/ CEC, but the difference of 0.04% is very small, with almost no difference between them. This is likely because in high-budget settings, the model's ACC improves, and overconfidence is mitigated, allowing w/o CEC to perform well. Thus, in scenarios with relatively low overconfidence like high-budget settings, the necessity of CEC is low, but in scenarios prone to overconfidence like low-budget settings, the necessity of CEC is high.

Table 2: Comparison of AUBC for CTypiClust without CEC (w/o CEC) and with CEC (w/ CEC). The numbers in parentheses indicate the budget size. The highest value for each budget is shown in bold.

Budget	Low(10)	Medium(100)	High(1000)
w/o CEC	59.34	75.81	**83.43**
w/ CEC	**61.13**	**76.29**	83.39

Verification of Overconfidence Mitigation by CEC. To verify the extent to which CEC mitigates overconfidence, we compare the overconfidence of simple confidence $\hat{c}$ and CEC. We use Expected Calibration Error (ECE) (Pakdaman Naeini et al., 2015) to quantitatively evaluate overconfidence. ECE ranges from 0 to 1, with lower ECE indicating less overconfidence. Figure 3 compares ECE for each budget between confidence $\hat{c}$ and CEC on the CIFAR-10 test data.

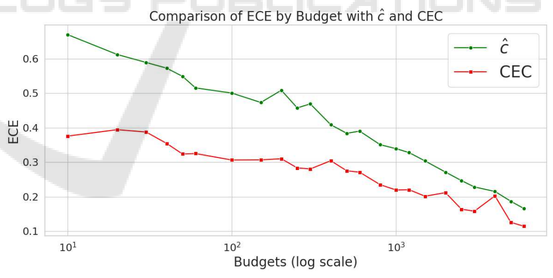


Figure 3: Comparison of ECE between confidence $\hat{c}$ and CEC.

Figure 3 shows that CEC has smaller ECE than $\hat{c}$ for all budgets. This indicates that CEC reduces overconfidence compared to $\hat{c}$. The difference in ECE between CEC and $\hat{c}$ is particularly large in settings with small budgets of 10 to 100.

The reason CEC mitigates overconfidence is likely due to the use of the agreement between the pseudo-label $\hat{y}$ obtained from the classifier g and the pseudo-label $\tilde{y}$ obtained by clustering the features. When the budget is low and learning is insufficient (low ACC), features are not well-separated by class, leading to many mismatched pseudo-labels and CEC values of 0 (from Equation 1). As the budget and ACC increase, features separate better, pseudo-labels match more, and CEC values rise.

To verify this, we visualized the relationship between ACC, test data feature space, and the two pseudo-labels. Figure 4 shows that with low ACC (as shown on the left side of Figure 4), features are poorly separated and pseudo-labels often mismatch, resulting in many CEC values of 0. With higher budgets and ACC (as shown on the right side of Figure 4), features separate better, pseudo-labels match more, and CEC values increase. Confidence $\tilde{c}$ also rises as clusters become more distinct.

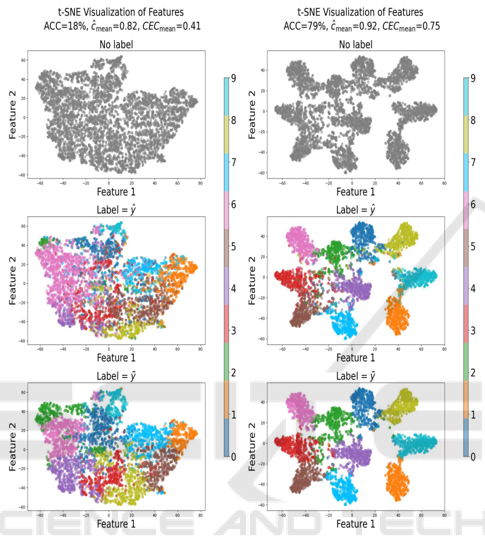


Figure 4: Visualization of CIFAR-10 test data features using t-SNE (Laurens and Hinton, 2008) and labeling the features with $\hat{y}$ and $\tilde{y}$. $\hat{c}_{\text{mean}}$ represents the mean of $\hat{c}$ for the test data, and CEC$_{\text{mean}}$ represents the mean of CEC for the test data. The left and right halves show the feature space when ACC is low and high, respectively. The top row represents the unlabeled feature space, the middle row represents the feature space labeled by $\hat{y}$, and the bottom row represents the feature space labeled by $\tilde{y}$.

Comparison of CTypiClust Performance for Different T_c. To investigate the impact of the parameter T_c on the performance of CTypiClust, we evaluate CTypiClust using various T_c values. In CTypiClust, the confidence CEC of typical data x_t determines whether x_t is used for training. The threshold for this decision is T_c, so we compare values from 0.5 to 0.9, excluding $T_c = 1$ as it corresponds to TypiClust. Figure 5 shows the performance differences of CTypiClust for each T_c on CIFAR-10. From Figure 5, CTypiClust performs better than Random for all budgets from low-budget to high-budget for any T_c. This is likely because, in CEC, $\mathbb{1}_{[\hat{y}=\tilde{y}]}$ in Equation 1 becomes 0 when the labels do not match, and CEC functions regardless of T_c when CEC is 0.

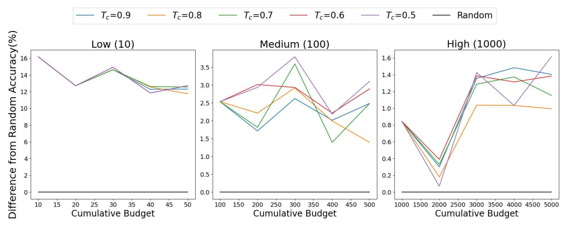


Figure 5: Graph showing the difference between CTypiClust and Random for each threshold T_c. Results are shown from left to right for low, medium, and high budgets.

Additionally, Table 3 shows the AUBC for each T_c. The difference between the maximum and minimum values for each budget is 0.12% (63.35-63.23) for low-budget, 0.71% (77.13-76.42) for medium-budget, and 0.27% (83.62-83.35) for high-budget, indicating that CTypiClust performs stably regardless of the parameter.

Table 3: AUBC for each T_c under different budgets. The numbers in parentheses indicate the budget size. The highest value for each budget is shown in bold.

T_c	0.5	0.6	0.7	0.8	0.9
Low(10)	63.25	63.25	**63.35**	63.23	63.23
Medium(100)	**77.13**	76.92	76.53	76.48	76.42
High(1000)	83.50	83.61	83.55	83.35	**83.62**

4.4 Limitations

Since CEC used in CTypiClust depends on the agreement between the model's classification results and the clustering results of the features, CTypiClust is specialized for classification problems and cannot be easily applied to regression tasks.

In the future, we aim to overcome these limitations and extend the method to make it applicable to various tasks, including regression tasks.

5 CONCLUSION

We proposed CTypiClust, which performs highly regardless of the budget. CTypiClust performs well in both low-budget and high-budget settings by considering confidence in TypiClust. Additionally, to address overconfidence in immature models like in low-budget settings, we proposed a confidence calibration method CEC and applied it to CTypiClust. We evaluated CTypiClust on CIFAR-10, CIFAR-100, and STL-10, and found that it performs well across various budgets. We also experimentally verified that CEC mitigates overconfidence. Since CTypiClust is specialized for classification problems, we plan to extend CTypiClust to make it applicable to various tasks, including regression tasks, in the future.

REFERENCES

Chen, L., Bai, Y., Huang, S., Lu, Y., Wen, B., Yuille, A., and Zhou, Z. (2023). Making your first choice: To address cold start problem in medical active learning. In *Medical Imaging with Deep Learning*.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*.

Coates, A., Ng, A., and Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*.

Gal, Y., Islam, R., and Ghahramani, Z. (2017). Deep Bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning*.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*.

Hacohen, G., Dekel, A., and Weinshall, D. (2022). Active learning on a budget: Opposite strategies suit high and low budgets. In *Proceedings of the 39th International Conference on Machine Learning*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jaiswal, A., Babu, A. R., Zadeh, M. Z., Banerjee, D., and Makedon, F. (2021). A survey on contrastive self-supervised learning. arXiv preprint arXiv:2011.00362.

Kirsch, A., van Amersfoort, J., and Gal, Y. (2019). Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. In *Advances in Neural Information Processing Systems*.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. Technical report, University of Toronto. Online.

Laurens, v. d. M. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9.

Mozafari, A. S., Gomes, H. S., Leão, W., Janny, S., and Gagné, C. (2019). Attended temperature scaling: A practical approach for calibrating deep neural networks. arXiv preprint arXiv:1810.11586.

Munjal, P., Hayat, N., Hayat, M., Sourati, J., and Khan, S. (2022). Towards robust and reproducible active learning using neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Pakdaman Naeini, M., Cooper, G., and Hauskrecht, M. (2015). Obtaining well calibrated probabilities using bayesian binning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 29.

Pereyra, G., Tucker, G., Chorowski, J., Łukasz Kaiser, and Hinton, G. (2017). Regularizing neural networks by penalizing confident output distributions. arXiv preprint arXiv:1701.06548.

Platt, J. (2000). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*.

Pop, R. and Fulop, P. (2018). Deep ensemble bayesian active learning : Addressing the mode collapse issue in monte carlo dropout via ensembles. arXiv preprint arXiv:1811.03897.

Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Gupta, B. B., Chen, X., and Wang, X. (2021). A survey of deep active learning. arXiv preprint arXiv:2009.00236.

Roth, D. and Small, K. (2006). Margin-based active learning for structured output spaces. In *Proceedings of the European Conference on Machine Learning*.

Sener, O. and Savarese, S. (2018). Active learning for convolutional neural networks: A core-set approach. In *6th International Conference on Learning Representations, ICLR 2018*.

Sinha, S., Ebrahimi, S., and Darrell, T. (2019). Variational adversarial active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Snell, J., Swersky, K., and Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*.

Wang, C. (2024). Calibration in deep learning: A survey of the state-of-the-art. arXiv preprint arXiv:2308.01222.

Yi, J. S. K., Seo, M., Park, J., and Choi, D.-G. (2022). Using self-supervised pretext tasks for active learning. In *Proceedings of the European Conference on Computer Vision(ECCV)*.

Zhan, X., Liu, H., Li, Q., and Chan, A. B. (2021). A comparative survey: Benchmarking for pool-based active learning. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*.

Zhdanov, F. (2019). Diverse mini-batch active learning. arXiv preprint arXiv:1901.05954.