Open World Classification

Lei Shu

What is Open World?

Assume that what is not known to be true is simply unknown.

Open World Assumption applies when a system has incomplete information

For example:

Virtual Assistant

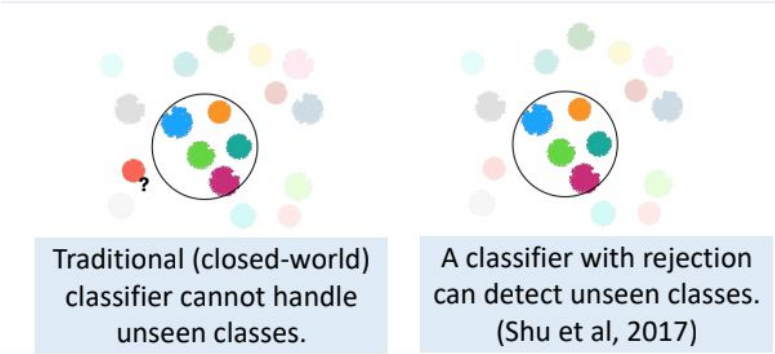
Close World Classification vs Open World Classification

Close World Classification

The classes appeared in the test data must have appeared in training.

Open World Classification

In an open environment, the ideal classifier should classify incoming data to the correct existing classes that appeared in training and detect those examples that do not belong to any existing classes.



Topics

Open World Classification Via Decision Boundaries [1]

Open Representation Learning [4]

Unseen Class Discovery [2]

Incremental Open World Classification [3]

Zero-shot Open World Classification [5]

Papers

[1] DOC: Deep Open Classification of Text Documents.

***Lei Shu**, Hu Xu, Bing Liu. EMNLP 2017*

[2] Unseen Class Discovery in Open-world Classification.

***Lei Shu**, Hu Xu, Bing Liu. arXiv:1801.05609*

[3] Open-world Learning and Application to Product Classification.

*Hu Xu, Bing Liu, **Lei Shu**, Philip S Yu. WWW 2019*

[4] ODIST: Open World Classification via Distributionally Shifted Instances.

***Lei Shu**, Yassine Benajiba, Saab Mansour and Yi Zhang. EMNLP-Findings 2021*

[5] Zero-Shot Open Set Detection by Extending CLIP.

*Sepideh Esmaeilpour, Bing Liu, Eric Robertson and **Lei Shu**. arXiv:2109.02748*

Relationships-Problem Definition

Open World Classification

m seen classes Y = {l_1, ...,l_m}, reject class is l_0

Training Corpus D_{train} = {(x_i,y_i)}, y_i must be in Y

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or

Unseen Class Discovery

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or C={c_1, ...}

Incremental Open World Classification

seen classes series **Y** = {Y_1, ...}, every Y has no overlapping.
Training Corpus Series $\mathbf{D_{train}}$ = {D_{train1}, ...} , D_{train1} = {(x_i,y_i)}, y_i must be in Y_1
Testing Corpus Series $\mathbf{D_{test}}$ = {D_{test1}, ...} , D_{test1} = {(x_j,y_j)}, y_j must be in Y_1 or l_0

Zero-shot Open World Classification

Training Corpus D_{train} = {(y_i)}, y_i must be in Y

Relationships-Technique

multiclass classifier

Open World Classification Via Decision Boundaries [1]

m seen class classifier >> m seen class decision boundaries

Open Representation Learning [4]

(m+1) (augmented out-out-distribution class) classifier >> (m+1) class decision boundaries

pairwise network to model the example similarities (next page)

Relationships-Technique

multiclass classifier (previous page)

pairwise network to model the example similarities

Unseen Class Discovery [2]

learn pairwise network from m seem class >> extend to testing examples for clustering

Incremental Open World Classification [3]

meta-learning pairwise network >> maintain each seen class's support examples for open world classification

Zero-shot Open World Classification [5]

pretrained vision-language similarity network >> compare testing image to seen labels and generated image description

Roadmap-Paper

[1] DOC: Deep Open Classification of Text Documents.

Lei Shu, *Hu Xu, Bing Liu. EMNLP 2017*

[2] Unseen Class Discovery in Open-world Classification.

Lei Shu, *Hu Xu, Bing Liu. arXiv:1801.05609*

[3] Open-world Learning and Application to Product Classification.

Hu Xu, Bing Liu, ***Lei Shu****, Philip S Yu. WWW 2019*

[4] ODIST: Open World Classification via Distributionally Shifted Instances.

Lei Shu*, Yassine Benajiba, Saab Mansour and Yi Zhang. EMNLP-Findings 2021*

[5] Zero-Shot Open Set Detection by Extending CLIP.

Sepideh Esmaeilpour, Bing Liu, Eric Robertson and ***Lei Shu****. arXiv:2109.02748*

Roadmap-Problem Definition

Open World Classification

m seen classes Y = {l_1, ...,l_m}, reject class is l_0

Training Corpus D_{train} = {(x_i,y_i)}, y_i must be in Y

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or

Unseen Class Discovery

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or C={c_1, ...}

Incremental Open World Classification

seen classes series **Y** = {Y_1, ...}, every Y has no overlapping.

Training Corpus Series $\mathbf{D_{train}}$ = {D_{train1}, ...} , D_{train1} = {(x_i,y_i)}, y_i must be in Y_1

Testing Corpus Series $\mathbf{D_{test}}$ = {D_{test1}, ...} , D_{test1} = {(x_j,y_j)}, y_j must be in Y_1 or l_0

Zero-shot Open World Classification

Training Corpus D_{train} = {(y_i)}, y_i must be in Y

Roadmap-Technique

multiclass classifier

Open World Classification Via Decision Boundaries [1]

m seen class classifier >> m seen class decision boundaries

Open Representation Learning [4]

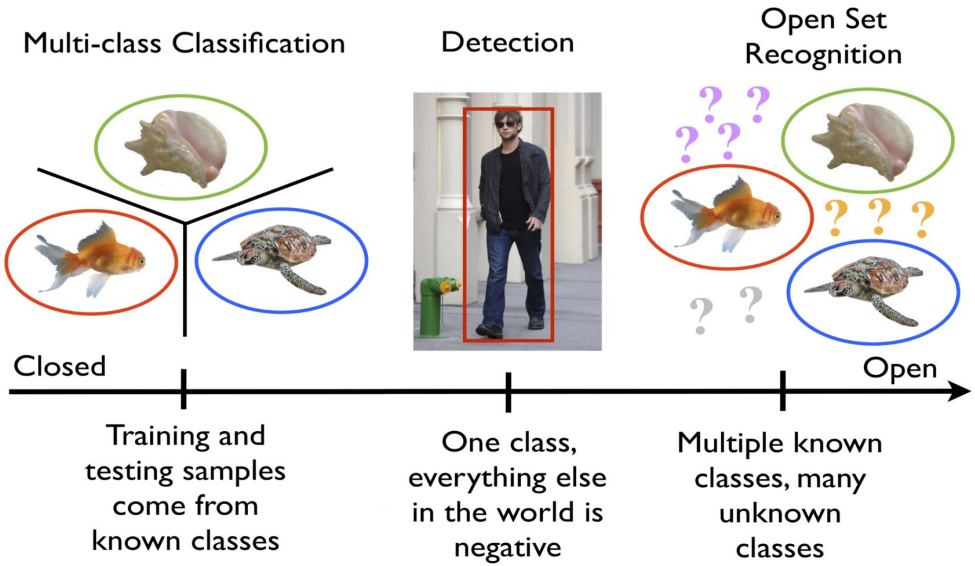
(m+1) (augmented out-out-distribution class) classifier >> (m+1) class decision boundaries

pairwise network to model the example similarities (next page)

Decision Boundary Finding

m seen classes $Y = \{l_1, ..., l_m\}$, reject class is l_0

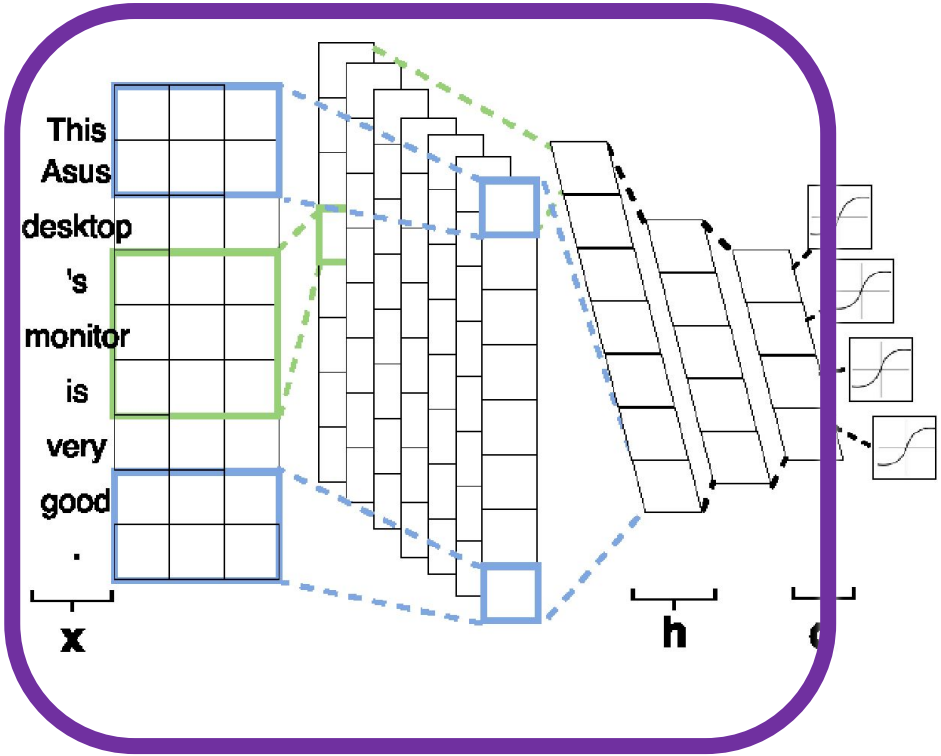
decision boundary finding $T = \{t_1, ..., t_m\}$ >> open space reduction



DOC: Deep Open Classification of Text Documents (Shu et. al, EMNLP 2017)

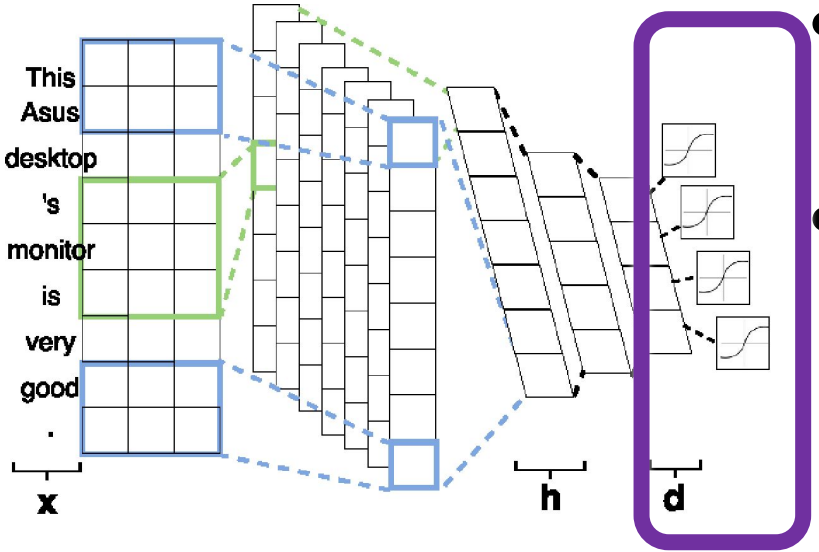
- Propose a novel method based on deep learning for open text classification (DOC).
- The proposed method markedly outperforms state-of-the-art existing approaches from both text classification and image classification fields.
- Idea:
  - Build a multi-class classifier with a 1-vs-rest final layer of Sigmoids
  - Reduce the open space risk further for rejection (l_0) by tightening the decision boundaries of Sigmoid functions with Gaussian fitting.
    - Open space risk: the classifier should not cover too much empty space.

CNN and Feed Forward Layers



- **Embedding layer** embeds words in document into dense vectors
- **Convolutional layer** performs convolution over dense vectors using different filters of varied sizes
- **Max-over-time pooling layer** selects the maximum values from the results of the convolution layer to form a feature vector
- **2 fully connected layers and one intermediate ReLU activation** reduce feature vector to m-dimensional vector

1-vs-Rest Layer



- m Sigmoid functions for m seen classes
  - i-th Sigmoid function corresponding to class l_i
- Model's loss function:
  - summation of all log loss of the m Sigmoid functions

$$\text{Loss} = \sum_{i=1}^{m}\sum_{j=1}^{n} -\mathbb{I}(y_j = l_i)\log p(y_j = l_i) - \mathbb{I}(y_j \neq l_i)\log(1 - p(y_j = l_i)),$$

Comparing 1-vs-Rest and Softmax

- Softmax as the final output layer
  - does not have the rejection capability
  - because the probability of prediction for each class is normalized across all training/seen classes

- 1-vs-Rest as the final output layer
  - allows rejection capability
  - for i-th class, it takes all training examples with y= l_i as positive examples and all the rest of the examples y != l_i as negative examples

Rejecting Instances during Testing

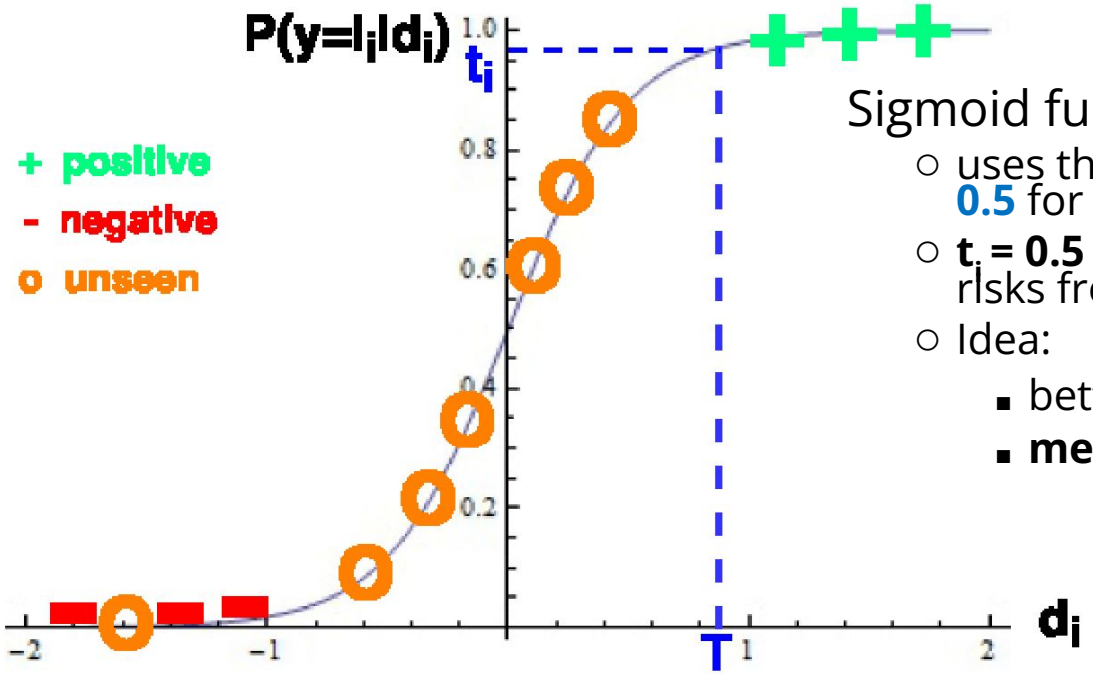
- Reinterpret the prediction of m Sigmoid functions to allow rejection

$$\hat{y} = \begin{cases} reject, \text{ if } \text{Sigmoid}(d_i) < t_i, \forall l_i \in \mathcal{Y}; \\ \arg\max_{l_i \in \mathcal{Y}} \text{Sigmoid}(d_i), \text{ otherwise.} \end{cases}$$

$\mathcal{Y}^m$

- For the i-th Sigmoid function
  - if the predicted probability Sigmoid(d_i) is less than a threshold t_i belonging to class l_i,
  - If all predicted probabilities are less than their corresponding thresholds for a test instance, the instance is rejected,
  - otherwise, its predicted class is the one with the highest probability.

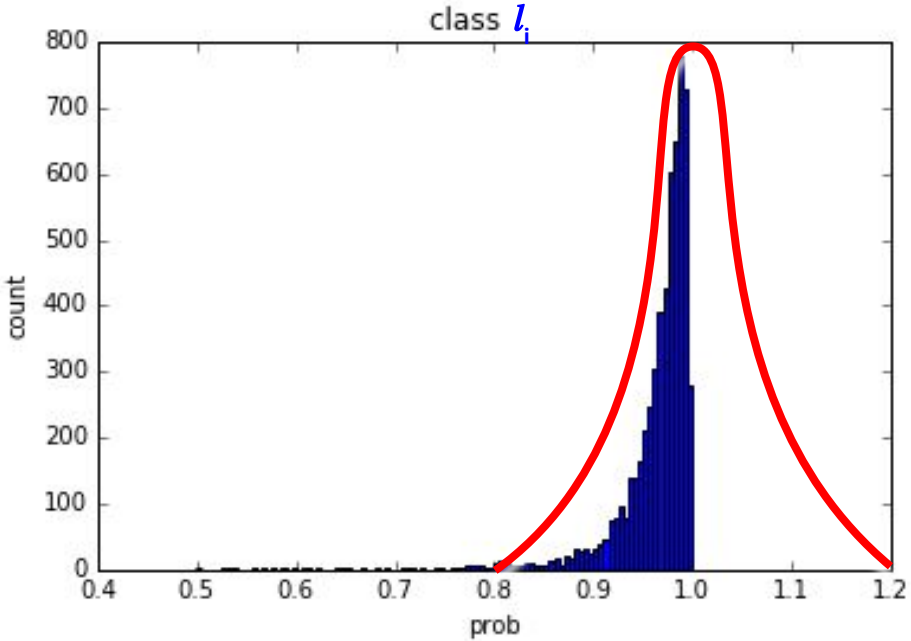
Reducing Open Space Risk Further



Sigmoid function

- uses the default probability threshold of t_i = **0.5** for classification of each class l_i
- **t_i = 0.5** does not consider potential open space risks from unseen (rejection l_0) class data
- Idea:
  - better probability threshold of t_i **>> 0.5**
  - **method**: **outlier detection**

Gaussian Fitting



- Dense positive examples with the probability threshold $t_i >> 0.5$
- Assume the predicted probabilities of all positive training data of each class follow half of the Gaussian distribution (with mean $\mu_i = 1$)
- Estimate the standard deviation σ_i
- Set the probability threshold $t_i = \max\{0.5, 1-\alpha\sigma_i\}$, usually $\alpha = 3$ as in outlier detection
- Different class l_i can have a different classification threshold t_i

Experiment Setting

- Datasets:
  - 20 Newsgroups: 20 non-overlapping classes. Each class has about 1000 documents
  - 50-class reviews: Amazon reviews of 50 type of products (classes). Each class has 1000 reviews.
- Training and test data: for each class, randomly sample 60% of documents for training, 10% for validation and 30% for testing.
- Vary the number of training/seen classes
  - use 25%, 50%, 75%, or 100% classes for training/seen and all classes for testing.
  - using 100% classes for training is the same as the traditional closed-world classification.
- Evaluation measure:
  - macro F1-score over m+1 classes (1 for rejection)

Experiment: Compared Methods

- **cbsSVM**: the latest method published in NLP
  - uses SVM to build 1-vs-rest CBS classifiers for multiclass text classification with rejection option.
  - all documents use TF-IDF term weighting scheme with no feature selection.

- **OpenMax**: the latest method published in computer vision
  - a CNN-based method for image classification
  - adapt it for text classification by using CNN with a softmax output layer, and adopt the OpenMax layer for open text classification
  - the result from softmax is reported when all classes are seen (100%), since OpenMax layer always performs rejection

- **DOC(t = 0.5):** Gaussian fitting isn't used to choose each t_i

- **DOC:** Gaussian fitting is used for furtherly reducing open space

Experiment: Hyperparameter Setting

- Use word vectors pre-trained from Google News (3 million words and 300 dimensions).
- For the CNN layers, 3 filter sizes are used [3, 4, 5]. For each filter size, 150 filters are applied.
- The dimension of the first fully connected layer is 250.

- Leverage Adam optimizer to optimize the loss function and empirically set the learning rate to 0.001, beta1 as 0.9, beta2 as 0.999 and epsilon as e^{-8}.

Results

Table 1: Macro-F_1 scores for 20 newsgroups

% of seen classes	25%	50%	75%	100%
cbsSVM	59.3	70.1	72.0	85.2
OpenMax	35.7	59.9	76.2	91.9
DOC ($t = 0.5$)	75.9	84.0	87.4	92.6
DOC	82.3	85.2	86.2	92.6

Table 2: Macro-F_1 scores for 50-class reviews

% of seen classes	25%	50%	75%	100%
cbsSVM	55.7	61.5	58.6	63.4
OpenMax	41.6	57.0	64.2	69.2
DOC ($t = 0.5$)	51.1	63.6	66.2	69.8
DOC	61.2	64.8	66.6	69.8

Conclusion

- Propose a novel deep learning based method, called DOC, for open text classification

- Using the same text datasets and experiment settings, we showed that DOC performs dramatically better than the state-of-the-art methods from both the text and image classification domains.

- We also believe that DOC is applicable to images.

Roadmap-Paper

[1] DOC: Deep Open Classification of Text Documents.

***Lei Shu**, Hu Xu, Bing Liu. EMNLP 2017*

[2] Unseen Class Discovery in Open-world Classification.

***Lei Shu**, Hu Xu, Bing Liu. arXiv:1801.05609*

[3] Open-world Learning and Application to Product Classification.

*Hu Xu, Bing Liu, **Lei Shu**, Philip S Yu. WWW 2019*

[4] ODIST: Open World Classification via Distributionally Shifted Instances.

***Lei Shu**, Yassine Benajiba, Saab Mansour and Yi Zhang. EMNLP-Findings 2021*

[5] Zero-Shot Open Set Detection by Extending CLIP.

*Sepideh Esmaeilpour, Bing Liu, Eric Robertson and **Lei Shu**. arXiv:2109.02748*

Roadmap-Problem Definition

Open World Classification

m seen classes Y = {l_1, ...,l_m}, reject class is l_0

Training Corpus D_{train} = {(x_i,y_i)}, y_i must be in Y

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or

Unseen Class Discovery

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or C={c_1, ...}

Incremental Open World Classification

seen classes series **Y** = {Y_1, ...}, every Y has no overlapping.

Training Corpus Series $\mathbf{D_{train}}$ = {D_{train1}, ...} , D_{train1} = {(x_i,y_i)}, y_i must be in Y_1

Testing Corpus Series $\mathbf{D_{test}}$ = {D_{test1}, ...} , D_{test1} = {(x_j,y_j)}, y_j must be in Y_1 or l_0

Zero-shot Open World Classification

Training Corpus D_{train} = {(y_i)}, y_i must be in Y

Roadmap-Technique

multiclass classifier

Open World Classification Via Decision Boundaries [1]

m seen class classifier >> m seen class decision boundaries

Open Representation Learning [4]

(m+1) (augmented out-out-distribution class) classifier >> (m+1) class decision boundaries

pairwise network to model the example similarities (next page)

Open Representation Learning

NLP example:

we have only learned features for ``it is red'' (for cherries) and ``it is yellow'' (for bananas) for a fruit classification task. The problem we are trying to overcome manifests when the model is exposed to a blueberry during testing.

during training, it does not possess a proper method to extract features for ``blue''. Ideally, we want a representation learning approach that can compute such a representation instead of using the representation of ``red'' or ``yellow''.

Distributional-shift Data Augmentation

Label: restaurant reservation

Text: can you make a reservation at the restaurant for tomorrow ?

Chunk

[can you] [make a reservation] [at the restaurant] [for tomorrow ?]

Mask

<mask> make a reservation … tomorrow ?	can you <mask> at the … for tomorrow ?	can you … reservation <mask> for tomorrow ?	can you … at the restaurant <mask>

Replace & Select the example which contradict the original text

Did you make … restaurant for tomorrow	can you tell us about your plans at the	can you make a reservation in advance for tomorrow	can … restaurant?
Can you make … restaurant for tomorrow	can you tell us what you are eating at	can you make a reservation at the hotel for tomorrow	can … restaurant that
Do you want to make … at the restaurant	can you please set up a table at the restaurant	can you make a reservation at the hotel for	can … restaurant?advertisement

Learning

Distributional-shift Instance Class: l_{m+1}

Supervised Classifier on Y and the augmented class Y' = {l_1, ...,l_m,l_{m+1}}

Decision Boundary Learning for Y', B' = {b_1, ...,b_m,b_{m+1}}, here we use adjustable decision boundary (ADB) method (Zhang et al 2021)

$$\hat{y} = \begin{cases} l_0 & \text{if } \tilde{y} = l_{m+1}, \\ l_0 & \text{elif } \forall j, 1 \leq j \leq m+1, \|\mathbf{r} - \mathbf{c_j}\| \geq b_j, \\ \tilde{y} & \text{otherwise .} \end{cases}$$

Experiment Setting

- Datasets:
  - Banking, OOS and Stack Overflow

	Banking	OOS	SO
Class	77	150	20
Train	9003	15000	12000
Valid	1000	3000	2000
Test	3080	5700	6000
Shift	127092	186219	143831

- Vary the number of training/seen classes
  - use 25%, 50%, 75%, classes for training/seen and all classes for testing.

- Evaluation measure:
  - precision, recall, and F1 score of unseen examples
  - macro-F1 of seen classes and overall accuracy

Result: Different Data Augmentation Methods

	P	R	F_1
ODIST-DB	**99.52**	**37.53**	**53.93**
ODIST-DB-Select	98.43	33.95	50.48
Word Delete 50%	98.51	7.26	13.52
Word Reorder 50%	96.61	5.0	9.5

precision, recall, and F1 score of unseen examples on OOS 25% setting

ODIST-DB: ODIST without decision boundary findings
ODIST-DB-Select: span replacement
Word Delete: randomly delete words in the text
Word Reorder: randomly reorder words in the text

Result: Close vs Open Representation

Dataset	Method	25%			50%			75%		
		Unseen	Seen	Acc	Unseen	Seen	Acc	Unseen	Seen	Acc
Banking	ADB	84.56	70.94	78.85	78.44	80.96	78.86	66.47	86.92	81.08
	ODIST	**87.11**±2.09	**72.72**±1.08	**81.69**±1.43	**81.32**±1.54	**81.79**±0.81	**80.90**±1.15	**71.95**±3.26	**87.20**±1.06	**82.79**±1.58
OOS	ADB	91.84	76.80	87.59	88.65	85.00	86.54	83.92	88.58	86.32
	ODIST	**93.42**±1.39	**79.69**±2.53	**89.79**±1.99	**90.62**±0.71	**86.52**±0.87	**88.61**±0.82	**85.86**±0.96	**89.33**±0.53	**87.70**±0.74
SO	ADB	90.88	78.82	86.72	87.34	85.68	86.4	73.86	86.80	82.78
	ODIST	**94.41**±1.36	**83.18**±2.54	**91.53**±1.96	**89.57**±1.04	**87.13**±1.41	**88.52**±1.26	**75.21**±1.23	**87.66**±0.87	**83.75**±0.94

F1 score of unseen examples, macro-F1 of seen classes and overall accuracy

ADB: m seen class classifier + m class decision boundary finding

ODIST: (m+1) class classifier + (m+1) class decision boundary finding, the extra 1 class is from distributionally-shifted instances

Roadmap-Paper

[1] DOC: Deep Open Classification of Text Documents.

***Lei Shu**, Hu Xu, Bing Liu. EMNLP 2017*

[2] Unseen Class Discovery in Open-world Classification.

***Lei Shu**, Hu Xu, Bing Liu. arXiv:1801.05609*

[3] Open-world Learning and Application to Product Classification.

*Hu Xu, Bing Liu, **Lei Shu**, Philip S Yu. WWW 2019*

[4] ODIST: Open World Classification via Distributionally Shifted Instances.

***Lei Shu**, Yassine Benajiba, Saab Mansour and Yi Zhang. EMNLP-Findings 2021*

[5] Zero-Shot Open Set Detection by Extending CLIP.

*Sepideh Esmaeilpour, Bing Liu, Eric Robertson and **Lei Shu**. arXiv:2109.02748*

Roadmap-Problem Definition

Open World Classification

m seen classes Y = {l_1, ...,l_m}, reject class is l_0

Training Corpus D_{train} = {(x_i,y_i)}, y_i must be in Y

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or

Unseen Class Discovery

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or C={c_1, ...}

Incremental Open World Classification

seen classes series **Y** = {Y_1, ...}, every Y has no overlapping.

Training Corpus Series $\mathbf{D_{train}}$ = {D_{train1}, ...} ,

D_{train1} = {(x_i,y_i)}, y_i must be in Y_1

Testing Corpus Series $\mathbf{D_{test}}$ = {D_{test1}, ...} ,

D_{test1} = {(x_j,y_j)}, y_j must be in Y_1 or l_0

Zero-shot Open World Classification

Training Corpus D_{train} = {(y_i)},

y_i must be in Y

Relationships-Technique

multiclass classifier (previous page)

pairwise network to model the example similarities

Unseen Class Discovery [2]

learn pairwise network from m seem class >> extend to testing examples for clustering

Incremental Open World Classification [3]

meta-learning pairwise network >> maintain each seen class's support examples for open world classification

Zero-shot Open World Classification [5]

pretrained vision-language similarity network >> compare testing image to seen labels and generated image description

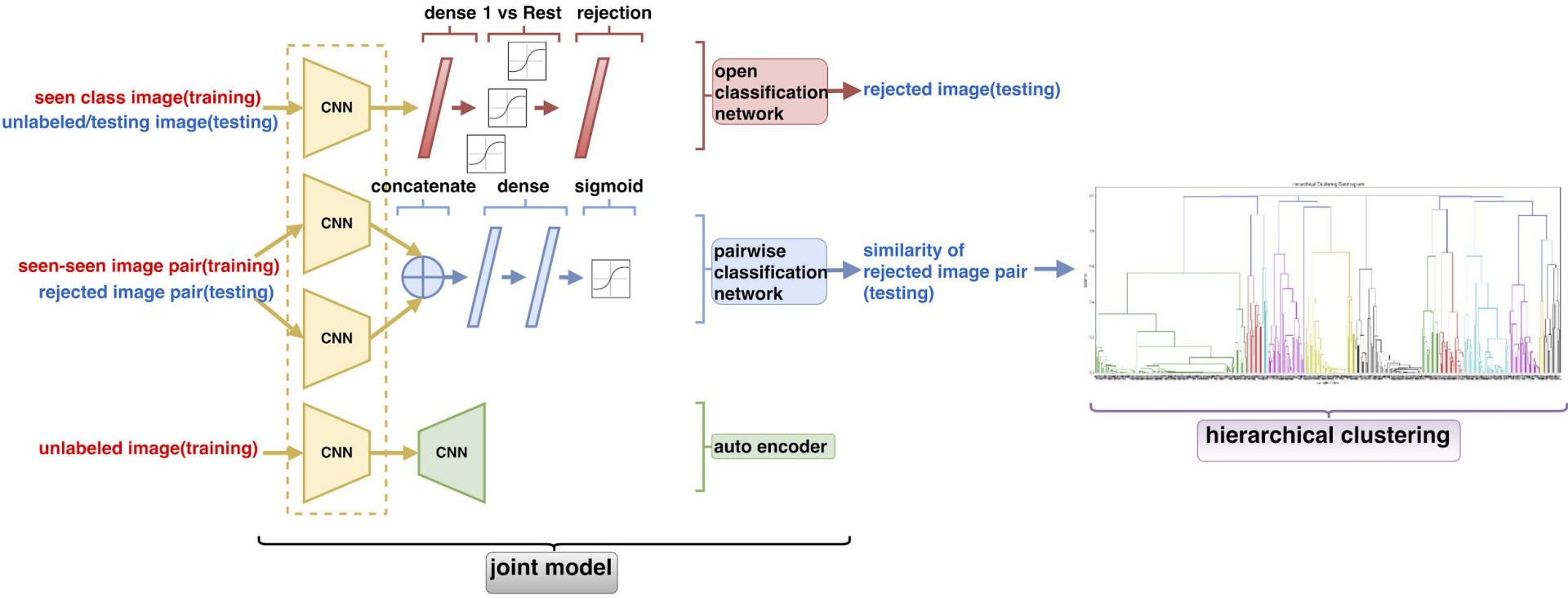
Unseen Class Discovery

Automatically discovering the hidden unseen classes of the rejected examples

Beyond reject examples to the class l_0, we aim to find the clusters $C=\{c_1, \ldots\}$ inside the rejected examples

Unseen Class Discovery in Open-world Classification (Shu et. al, arXiv 2018)

- the data from the seen training classes, which can tell us what kind of similarity/difference is expected for examples from the same class or from different classes.
- It is reasonable to assume that this knowledge can be transferred to the rejected examples and used to discover the hidden unseen classes in them.
- Idea:
  - joint open classification model
  - with a sub-model for classifying whether a pair of examples belongs to the same or different classes.
  - This sub-model can serve as a distance function for clustering to discover the hidden classes of the rejected examples.


dense 1 vs Rest
rejection
seen class image(training)
unlabeled/testing image(testing)
CNN
open classification network
rejected image(testing)
concatenate
dense
sigmoid
CNN
seen-seen image pair(training)
rejected image pair(testing)
CNN
pairwise classification network
similarity of rejected image pair (testing)
unlabeled image(training)
CNN
CNN
auto encoder
joint model
hierarchical clustering

Experiment Setting: Dataset

MNIST : handwritten digits (10 classes), which has a training set of 60,000 examples, and a test set of 10,000 examples. We use 6 classes as the set of seen classes and use the rest 4 classes as unseen classes (all randomly chosen).

EMNIST (Cohen et al., 2017): EMNIST is an extension of MNIST to commonly used characters such as English alphabet. It is derived from the NIST Special Database 19. We use EMNIST Balanced dataset with 47 balanced classes. It has a training set of 112,800 examples and a test set of 18,800 examples. We use 33 classes as the set of seen classes, 10 classes as the unseen classes and 4 classes as the validation seen classes (again, all randomly chosen).

We use the same validation classes from the following EMNIST dataset as the validation dataset for MNIST.

Experiment Setting: Evaluation Metrics

Number of clusters: We compare the number of discovered clusters and the true number of clusters in the unseen class test data.

Quality of clusters: Here we compare the cluster membership of the test images with these images' ground-truth labels using the popular evaluation metric (we re-defined some notations here): Normalized Mutual Information (NMI) (Pluim et al., 2000), which is a normalization of the Mutual Information (MI) to scale the results to between 0 (no mutual information) and 1 (perfect correlation).

Results

	MNIST				EMNIST			
algorithm	$(m+1)$ classes	rejection class			$(m+1)$ classes	rejection class		
	macro-$\mathcal{F}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	macro-$\mathcal{F}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$
OCN	0.914	0.920	0.824	0.869	0.832	0.664	0.47	0.554
OpenMax(weibull=20)	0.678	0.955	0.026	0.051	0.789	0.786	0.07	0.13
OpenMax(weibull=1000)	0.684	0.956	0.043	0.083	0.803	0.725	0.239	0.359

Table 1: Macro-$\mathcal{F}$ is average F-score on $m+1$ classes, where m is the number of seen classes and 1 is the rejection class. $\mathcal{P}$, $\mathcal{R}$ and $\mathcal{F}$ are precision, recall and F-score of the rejection class only

Type of Pair	MNIST	EMNIST
seen-seen	0.994	0.965
seen-unseen	0.752	0.874
unseen-unseen	0.700	0.810

Table 2: Accuracy of pairwise classification (whether two examples are from the same class or not)

Results

algorithm	GT	Encoder + HC		K-means NMI score		PCN + HC	
	# of C	# of C	NMI	K from GT	K from PCN+HC	# of C	NMI
MNIST	4	3	0.414	0.710	0.66	5	0.302
EMNIST	10	4	0.479	0.683	0.683	10	0.583

Table 3: Clustering results of unseen classes on MNIST and EMNIST. GT means the ground truth number of clusters for unseen classes, and # of C means the number of clusters.

algorithm	GT	OCN+Encoder+HC		K-means NMI score		OCN+PCN+HC	
	# of C	# of C	NMI	K from GT	K from OCN+PCN+HC	# of C	NMI
MNIST	4	4	0.478	0.563	0.591	6	0.320
EMNIST	10	4	0.312	0.586	0.543	14	0.500

Table 4: Clustering results of rejected examples on MNIST and EMNIST. GT means the ground truth number of clusters for unseen classes, and # of C means the number of clusters.

Roadmap-Paper

[1] DOC: Deep Open Classification of Text Documents.

***Lei Shu**, Hu Xu, Bing Liu. EMNLP 2017*

[2] Unseen Class Discovery in Open-world Classification.

***Lei Shu**, Hu Xu, Bing Liu. arXiv:1801.05609*

[3] Open-world Learning and Application to Product Classification.

*Hu Xu, Bing Liu, **Lei Shu**, Philip S Yu. WWW 2019*

[4] ODIST: Open World Classification via Distributionally Shifted Instances.

***Lei Shu**, Yassine Benajiba, Saab Mansour and Yi Zhang. EMNLP-Findings 2021*

[5] Zero-Shot Open Set Detection by Extending CLIP.

*Sepideh Esmaeilpour, Bing Liu, Eric Robertson and **Lei Shu**. arXiv:2109.02748*

Roadmap-Problem Definition

Open World Classification

m seen classes Y = {l_1, ...,l_m}, reject class is l_0

Training Corpus D_{train} = {(x_i,y_i)}, y_i must be in Y

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or l_0

Unseen Class Discovery

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or C={c_1, ...}

Incremental Open World Classification

seen classes series **Y** = {Y_1, ...}, every Y has no overlapping.

Training Corpus Series $\mathbf{D_{train}}$ = {D_{train1}, ...} , D_{train1} = {(x_i,y_i)}, y_i must be in Y_1

Testing Corpus Series $\mathbf{D_{test}}$ = {D_{test1}, ...} , D_{test1} = {(x_j,y_j)}, y_j must be in Y_1 or l_0

Zero-shot Open World Classification

Training Corpus D_{train} = {(y_i)}, y_i must be in Y

Relationships-Technique

multiclass classifier (previous page)

pairwise network to model the example similarities

Unseen Class Discovery [2]

learn pairwise network from m seem class >> extend to testing examples for clustering

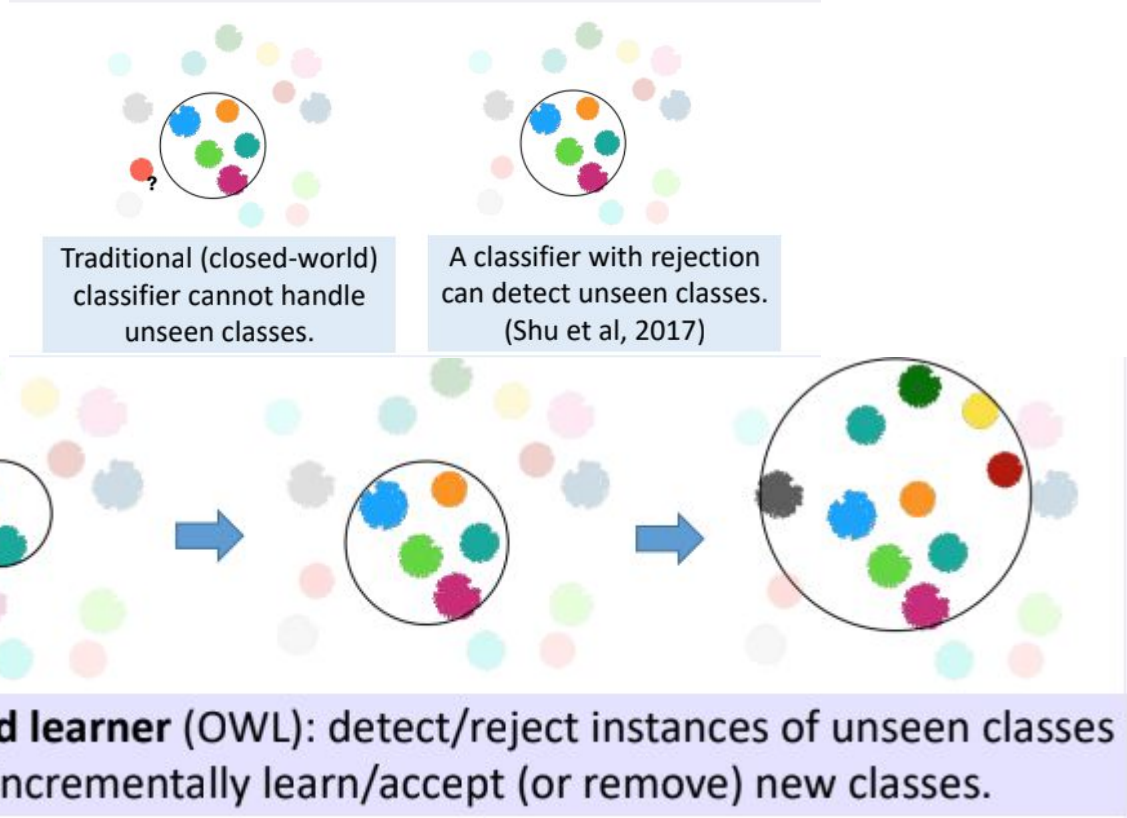
Incremental Open World Classification [3]

meta-learning pairwise network >> maintain each seen class's support examples for open world classification

Zero-shot Open World Classification [5]

pretrained vision-language similarity network >> compare testing image to seen labels and generated image description

Incremental Open World Classification

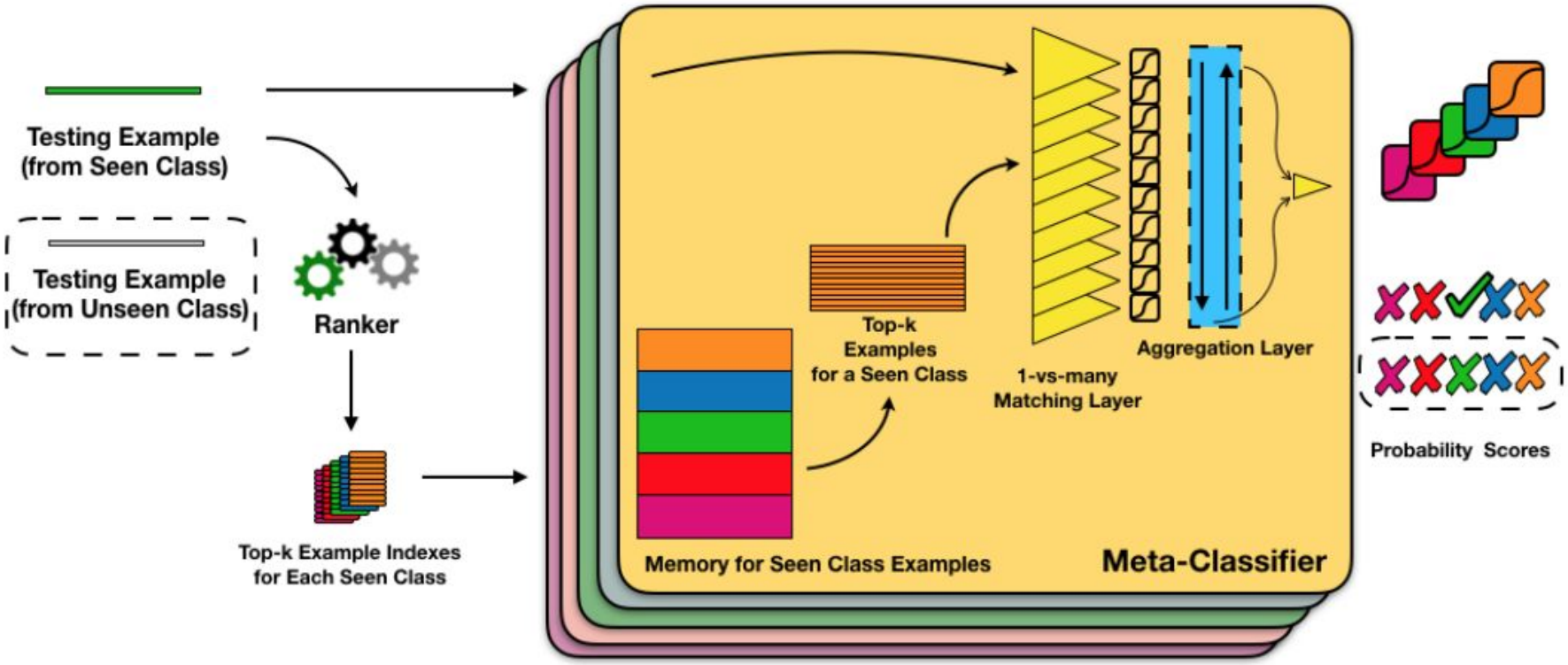


Problem Statement: At any point in time, the learning system is aware of a set of seen classes $S = \{c_1, \ldots, c_m\}$ and has an OWL model/classifier for S but is unaware of a set of unseen classes $U = \{c_{m+1}, \ldots\}$ (any class not in S can be in U) that the model may encounter. The goal of an OWL model is two-fold: (1) classifying examples from classes in S and reject examples from classes in U, and (2) when a new class c_{m+1} (without loss of generality) is removed from U (now $U = \{c_{m+2}, \ldots\}$) and added to S (now $S = \{c_1, \ldots, c_m, c_{m+1}\}$, still being able to perform (1) without re-training the model.

$$\hat{y} = \begin{cases} \textit{reject}, \text{ if } \max_{c \in S} p(c|x_t, x_{a_{1:k}}) \leq 0.5; \\ \\ \arg\max_{c \in S} p(c|x_t, x_{a_{1:k}}), \text{ otherwise.} \end{cases}$$

Proposed technique - L2AC – based on meta-learning

1. It maintains a dynamic set S of seen classes that allow new classes to be added or deleted with no model re-training.
2. Each class is represented by a small set of training examples.
3. In testing, the meta-classifier uses only the examples of the maintained seen classes so far on-the-fly for classification and rejection


Testing Example
(from Seen Class)
Testing Example
(from Unseen Class)
Ranker
Top-k Example Indexes
for Each Seen Class
Top-k
Examples
for a Seen Class
1-vs-many
Matching Layer
Aggregation Layer
Memory for Seen Class Examples
Meta-Classifier
Probability Scores

Results

Methods	$\|S\| = 25$ (WF1)	$\|S\| = 25$ (MF1)	$\|S\| = 50$ (WF1)	$\|S\| = 50$ (MF1)	$\|S\| = 75$ (WF1)	$\|S\| = 75$ (MF1)
DOC-CNN	53.25(1.0)	55.04(0.39)	70.57(0.46)	76.91(0.27)	81.16(0.47)	86.96(0.2)
DOC-LSTM	57.87(1.26)	57.6(1.18)	69.49(1.58)	75.68(0.78)	77.74(0.48)	84.48(0.33)
DOC-Enc	82.92(0.37)	75.09(0.33)	82.53(0.25)	84.34(0.23)	83.84(0.36)	88.33(0.19)
DOC-CNN-Gaus	85.72(0.43)	76.79(0.41)	83.33(0.31)	83.75(0.26)	84.21(0.12)	87.86(0.21)
DOC-LSTM-Gaus	80.31(1.73)	70.49(1.55)	77.49(0.74)	79.45(0.59)	80.65(0.51)	85.46(0.25)
DOC-Enc-Gaus	88.54(0.22)	80.77(0.22)	84.75(0.21)	85.26(0.2)	83.85(0.37)	87.92(0.22)
L2AC-*n*9-NoVote	91.1(0.17)	82.51(0.39)	84.91(0.16)	83.71(0.29)	81.41(0.54)	85.03(0.62)
L2AC-*n*9-Vote3	91.54(0.55)	82.42(1.29)	84.57(0.61)	82.7(0.95)	80.18(1.03)	83.52(1.14)
L2AC-*k*5-*n*9-AbsSub	92.37(0.28)	84.8(0.54)	85.61(0.36)	84.54(0.42)	83.18(0.38)	86.38(0.36)
L2AC-*k*5-*n*9-Sum	83.95(0.52)	70.85(0.91)	76.09(0.36)	75.25(0.42)	74.12(0.51)	78.75(0.57)
L2AC-***k*5-*n*9**	93.07(0.33)	86.48(0.54)	86.5(0.46)	85.99(0.33)	84.68(0.27)	88.05(0.18)
L2AC-*k*5-*n*14	**93.19**(0.19)	86.91(0.33)	**86.63**(0.28)	86.42(0.2)	85.32(0.35)	88.72(0.23)
L2AC-*k*5-*n*19	93.15(0.24)	86.9(0.45)	86.62(0.49)	86.48(0.43)	**85.36**(0.66)	88.79(0.52)

Table 1: Weighted F1 (WF1) and macro F1 (MF1) scores on a test set with 100 classes with 3 settings: 25, 50, and 75 seen classes. The set of seen classes are incrementally expanded from 25 to 75 classes (or gradually shrunk from 75 to 25 classes). The results are the averages over 10 runs with standard deviations in parenthesis.

Roadmap-Paper

[1] DOC: Deep Open Classification of Text Documents.

***Lei Shu**, Hu Xu, Bing Liu. EMNLP 2017*

[2] Unseen Class Discovery in Open-world Classification.

***Lei Shu**, Hu Xu, Bing Liu. arXiv:1801.05609*

[3] Open-world Learning and Application to Product Classification.

*Hu Xu, Bing Liu, **Lei Shu**, Philip S Yu. WWW 2019*

[4] ODIST: Open World Classification via Distributionally Shifted Instances.

***Lei Shu**, Yassine Benajiba, Saab Mansour and Yi Zhang. EMNLP-Findings 2021*

[5] Zero-Shot Open Set Detection by Extending CLIP.

*Sepideh Esmaeilpour, Bing Liu, Eric Robertson and **Lei Shu**. arXiv:2109.02748*

Roadmap-Problem Definition

Open World Classification

m seen classes Y = {l_1, ...,l_m}, reject class is l_0

Training Corpus D_{train} = {(x_i,y_i)}, y_i must be in Y

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or l_0

Unseen Class Discovery

Testing Corpus D_{test} = {(x_j, y_j)}, y_j can be in Y or C={c_1, ...}

Incremental Open World Classification

seen classes series **Y** = {Y_1, ...}, every Y has no overlapping.

Training Corpus Series $\mathbf{D_{train}}$ = {D_{train1}, ...} ,

D_{train1} = {(x_i,y_i)}, y_i must be in Y_1

Testing Corpus Series $\mathbf{D_{test}}$ = {D_{test1}, ...} ,

D_{test1} = {(x_j,y_j)}, y_j must be in Y_1 or l_0

Zero-shot Open World Classification

Training Corpus D_{train} = {(y_i)}, y_i must be in Y

Relationships-Technique

multiclass classifier (previous page)

pairwise network to model the example similarities

Unseen Class Discovery [2]

learn pairwise network from m seem class >> extend to testing examples for clustering

Incremental Open World Classification [3]

meta-learning pairwise network >> maintain each seen class's support examples for open world classification

Zero-shot Open World Classification [5]

pretrained vision-language similarity network >> compare testing image to seen labels and generated image description

Zero-Shot Open World Classification

Training Corpus $D_{train} = \{(y_i)\}$, y_i must be in Y

Testing Corpus $D_{test} = \{(x_j, y_j)\}$, y_j can be in Y or l_0

Advantage:

- No fine-tuneing
- Support incremental open world classification

Challenge:

- Bridge label(text) and image >> **CLIP (Contrastive Language–Image Pre-training)**
- How to decide out-of-distribution?

Solution

Generate image description

Use CLIP to compare image description and label descriptions and pick the highest probability

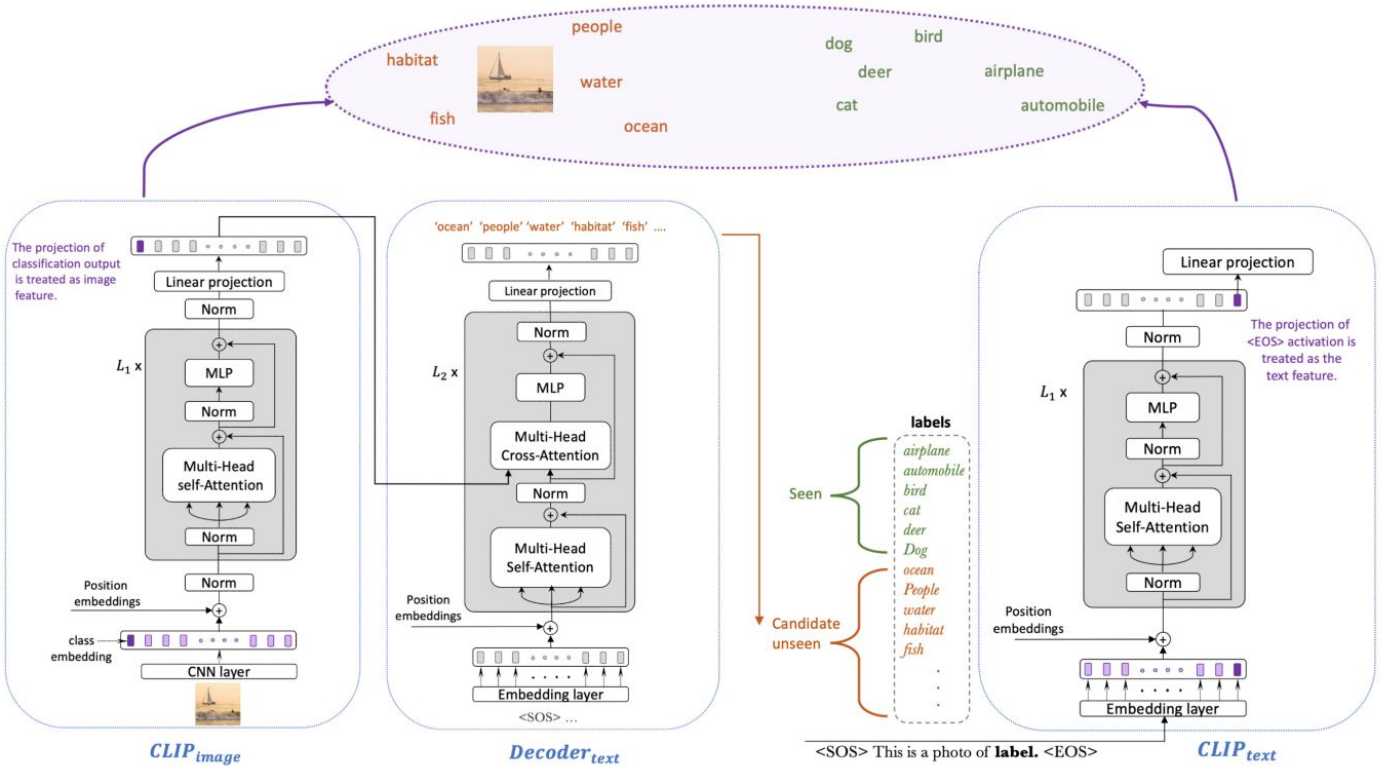


Figure 1: The diagram illustrates the inference steps of ZO-CLIP for a sample from an unseen class 'boat'. The available seen class labels (shown in green) are $\mathcal{Y}_s$={'airplane', 'automobile', 'bird', 'cat', 'deer', 'dog'}. In the first step, the image is encoded through $\text{CLIP}_{\text{image}}$ and then image description is generated in the output of $\text{Decoder}_{\text{text}}$. The description is in fact a set of candidate unseen labels $\mathcal{Y}_u$ (shown in orange). In the second step, $\mathcal{Y}_s \cup \mathcal{Y}_u$ are encoded through $\text{CLIP}_{\text{text}}$ on the right. The purple ellipsoid shows CLIP's feature space where the relevant labels are aligned with the image. CLIP quantifies the alignment by calculating the cosine similarity of each encoded label to the encoded image. Then $S(x)$ is obtained according to 2. The score is high for this image as it is more similar to the set of $\mathcal{Y}_u$ than $\mathcal{Y}_s$. The inference relies on CLIP pretrained encoders as well as $\mathcal{Y}_u$ generated by $\text{Decoder}_{\text{text}}$.

Results

	CIFAR10	CIFAR100	CIFAR+10	CIFAR+50	TinyImageNet
OpenMax (Bendale and Boult 2016)	69.5±4.4	NR	81.7±NR	79.6±NR	57.6±NR
DOC (Shu, Xu, and Liu 2017)	66.5±6.0	50.1±0.6	46.1±1.7	53.6±0.0	50.2±0.5
G-OpenMax (Ge et al. 2017)	67.5±4.4	NR	82.7±NR	81.9±NR	58.0±NR
OSRCI (Neal et al. 2018)	69.9±3.8	NR	83.8±NR	82.7±0.0	58.6±NR
C2AE (Oza and Patel 2019)	71.1±0.8	NR	81.0±0.5	80.3±0.0	58.1±1.9
GFROR (Perera et al. 2020)	80.7±3.0	NR	92.8±0.2	92.6±0.0	60.8±1.7
CAC (Miller et al. 2021)	80.1±3.0	76.1±0.7	87.7±1.2	87.0±0.0	76.0±1.5
CSI (Tack et al. 2020)	87.0±4.0	80.4±1.0	94.0±1.5	97.0±0.0	76.9±1.2
G-ODIN (Hsu et al. 2020)	63.4±3.5	79.9±2.3	45.8±1.9	92.4±0.0	67.0±7.1
MSP (Hendrycks and Gimpel 2016)	88.0±3.3	78.1±3.1	94.9±0.8	95.0±0.0	80.4±2.5
Zero-shot open set detection(ours)	**93.0**±1.7	**82.1**±2.1	**97.8**±0.6	**97.6**±0.0	**84.6**±1.0

Table 1: Open-set detection performance in terms of AUROC. The results are averaged over 5 splits of each dataset (± standard deviation). We generated the results for DOC, CSI, G-ODIN and MSP. The results for the rest of the baselines are taken from (Miller et al. 2021).

20 seen class labels from 'tinyimagenet' dataset:

Y_s = {'potpie', 'kimono', 'school bus', 'go-kart', 'cliff dwelling', 'ice lolly', 'sandal', **'espresso'**, 'centipede', 'oboe', 'orange', 'German shepherd', 'beaker', 'obelisk', 'orangutan', 'bowtie', 'suspension bridge', 'vestment', 'frying pan', 'trolleybus}

	'guacamole' (*unseen* class)			'espresso' (*seen* class)		
						
Seen labels	'espresso' 'potpie' 'frying pan' 'orange'	'espresso' 'potpie' 'frying pan' 'orange'	'espresso' 'potpie' **'frying pan'** 'orange'	**'espresso'** 'potpie' 'sandal' 'orange'	**'espresso'** 'potpie' 'sandal' 'orange'	'espresso' 'potpie' 'sandal' 'orange' . . . .
Generated labels	**'cooking'** 'pan' 'dish' 'salad' 'ingredients' 'soup' 'sauces' 'grilled' . . .	**'dish'** **'food'** 'salad' 'bowl' 'dinner' 'cooking' 'ware' 'container' . . .	'cooking' 'pan' 'dish' 'salad' 'ingredients' 'soup' 'sauces' 'grilled' . . .	'cocktail' 'beverage' 'drink' 'alcohol' 'tea' 'beer' 'glass' 'cup' . . .	'coffee' 'tea' 'cup' 'dish' 'plate' 'bowl' 'mug' 'pan' . . .	**'coffee'** 'mug' 'Starbucks' 'cup' 'cafe' 'beverage' 'tea' 'drink' . . .
;(x)	***0.92***	***0.78***	***0.62***	***0.28***	***0.50***	***0.94***

Future Works

zero-shot open text classification

zero-shot open world classification on image using multiple pretrained vision-language models for further performance improvement

few-shot without parameter updating/prefix tuning open text classification

extend to slot labeling/NER

Reference

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. arXiv preprint arXiv:2103.00020.

Hendrycks, D.; and Gimpel, K. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. arXiv preprint arXiv:1610.02136.

Hanlei Zhang, Hua Xu, and Ting-En Lin. 2021. Deep open intent classification with adaptive decision boundary. Proceedings of the AAAI Conference on Artificial Intelligence, 35(16):14374–14382.