Towards Open World Recognition: Supplemental Material

Abhijit Bendale, Terrance Boult
University of Colorado at Colorado Springs
{abendale,tboult}@vast.uccs.edu

In this supplemental section, we provide additional material to further the reader's understanding of the work on Open World Recognition, CAP models and the Nearest Non-Outlier algorithm that we presented in the main paper. We present additional experiments on ILSVRC 2010 dataset. We then present experiments on ILSVRC 2012 dataset to demonstrate the performance gain of NNO over NCM (see fig 3 in the main paper) are not feature/dataset specific. We then provide algorithmic pseudocode for implementing the NNO algorithm. Finally we discuss the 1vs-Set extension to liblinear, its parameter tuning and its computational savings.

1. Experiments on ILSVRC 2010

1.1. Thresholding NCM-Softmax for ILSVRC 2010

In section 4.1 of the main paper, we explain the process of rejecting samples from unseen categories to balance open space risk and defined in Eq. 8, a probability function which is thresholded at zero. At first it might seem like a viable idea to just threshold the original softmax probability used in NCM. As explained in the main paper this will fail for open set because the normalization is improper and hence the softmax probability calibration will bias results. To convince the skeptical reader, we add a small experiment, similar to fig 3 in the main paper, and show the performance of classifying samples as unknown by directly thresholding softmax probabilities. As this is a smaller experiment, we show 2D plots instead of a 3D surface as the system is tested in closed set settings (fig 1a) and open set settings with 100 unknowns (fig 1b). NNO algorithm performs comparable with NCM in closed set settings. The reader can observe the performance of NCM-STH is similar to NCM and significantly worse than NNO on open set testing with 100 unknowns. Just thresholding the softmax probability is not enough, because its normalization keeps it from decaying as one move away from known data. This result confirms the suitability of balancing open set risk with Eq 8,using transformed learned Mahalanobis distance to the NCM. The results from this experiment are shown in fig 1.

1.2. Performance of NNO for different values of τ

Section 4.1 and 5.1 in the main paper describes NNO algorithm in detail and steps involved in estimating optimal τ required to balance open space risk. It is natural to ask how sensitive are the results to the choice of τ. In this section, we show the effect of different values of τ on the performance of NNO tested in open set settings with 100 unknown categories, to provide reader the feeling for the sensitivity to the parameter.

In the experimental results shown in Fig 3 in the main paper, we used optimal τ for evaluation purpose, which was approximately 5000. The optimal value near the beginning of a broad peak and small changes in τ have minimal impact. Even increasing it by 20% has only a small impact on open set testing. Fig 2 shows performance for varying set of τ. τ_{opt} is the optimal threshold that was selected. We observe that performance of NNO continues to improve as we near the optimal threshold from above. For a threshold value lower than τ_{opt} (e.g. 4000), the number of false rejects raises significantly. Thus, a balance between correct predictions retained and unknown categories rejected has to be maintained by the selected τ_{opt}. The results are obtained on ILSVRC'10 dataset, similar to fig 3a in the main paper. In our experiments, we observed similar trends for all other experiments – poor performance below the optimal τ and insensitivity to modest changes above it.

2. Experiments on ILSVRC 2012 Dataset

As noted in section 5 (Experiments) in the main paper, we used ILSVRC 2010 dataset because we needed access to ground truth labels. Ground truth is necessary to perform the open world recognition test protocol, which includes selecting known and unknown set of categories. In this section, we perform additional experiments on the training subset of ILSVRC 2012 [8] [1] dataset across multiple features to show that the effectiveness of NNO algorithm for closed set and open set tasks does not significantly depend on feature type.

[1] ILSVRC dataset remained unchanged between 2012, 2013 and 2014. Ground truth labels are available for training data only.

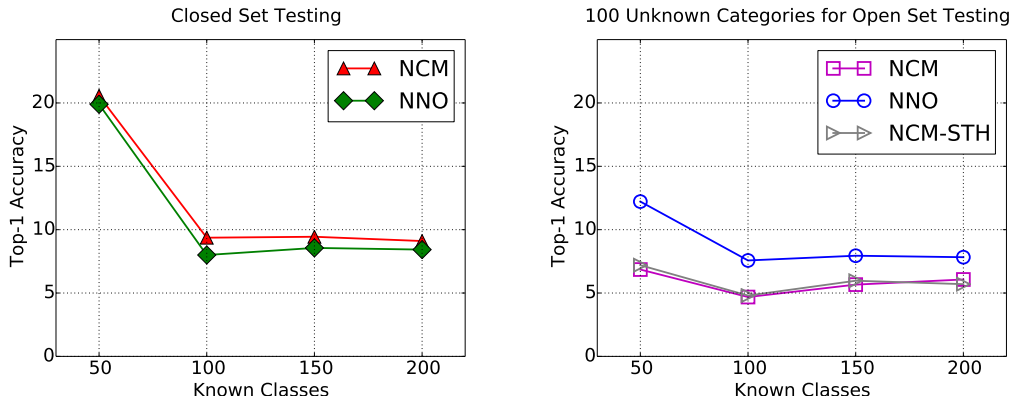

Figure 1: Effect of open set performance of thresholding softmax probabilities. Fig 1a shows performance with closed set testing and 1b shows performance on open set testing with 100 unknown categories. Metric learning was performed on 50 categories, followed by incremental learning phase with 50 categories in each phase. NCM-STH denotes NCM algorithm with open set testing with thresholded softmax probabilities. As can be seen clearly, just thresholding a probability estimates does not produce good open set performance

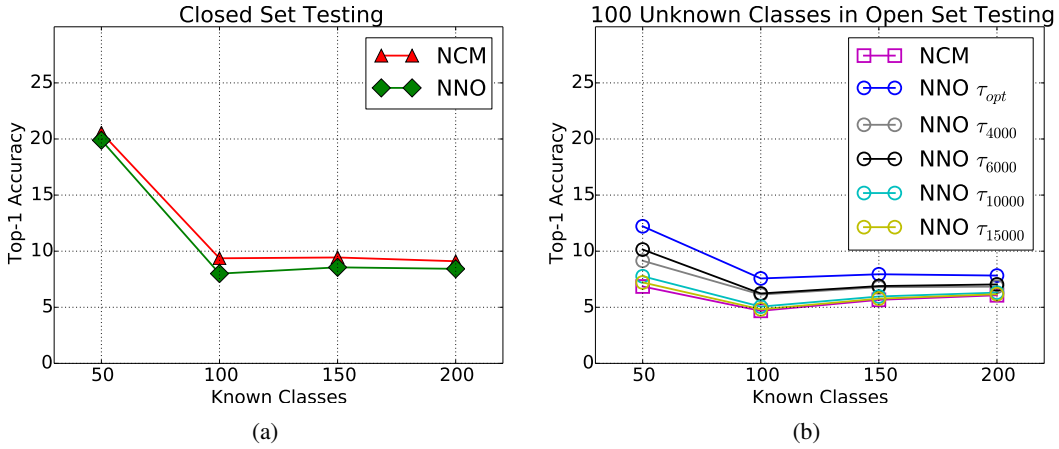

Figure 2: The above figure shows the effect of varying threshold τ on top-1 accuracy on ILSVRC'10 data. The results from closed set testing are shown in fig 2a and results from open set testing with 100 unknown categories are shown in fig 2b. Here $\tau_{opt} = 5000$, which was the selected threshold for experiments in fig 3a. For a threshold value lower than τ_{opt}, the number of correct predictions retained reduces significantly.

Since ground truth is not available for ILSVRC'12 dataset, we split the training data provided by the authors into training and test split. The number of categories is the same, this just limits the number of images per class used. We use 70% of training data to train models and 30% of the data for evaluation. This process is repeated over multiple folds. Once the data is split into training and test split the remaining procedure for metric learning and incremental learning is similar to that in section 5 (Experiments) in the main section. We conduct two sets of similar experiments on ILSVRC'12 data: metric learning with 50 and 200 initial categories as shown in Figs 3 and 4 in the main paper. The closed set and open set testing is conducted in similar manner as well. While the open world experimental setup for ILSVRC'12 is not ideal because of the smaller number of images per class, the goal of this experiment is to show that the advantages of NNO are not feature dependent.

We use pre-computed features as provided on cloudcv.org [1]. We consider three set of features as follows:

1. **DenseSIFT:** SIFT descriptors are densely extracted [6] using a flat window at two scales (4 and 8 pixel radii) on a regular grid at steps of 5 pixels. The three descriptors are stacked together for each HSV color channels,

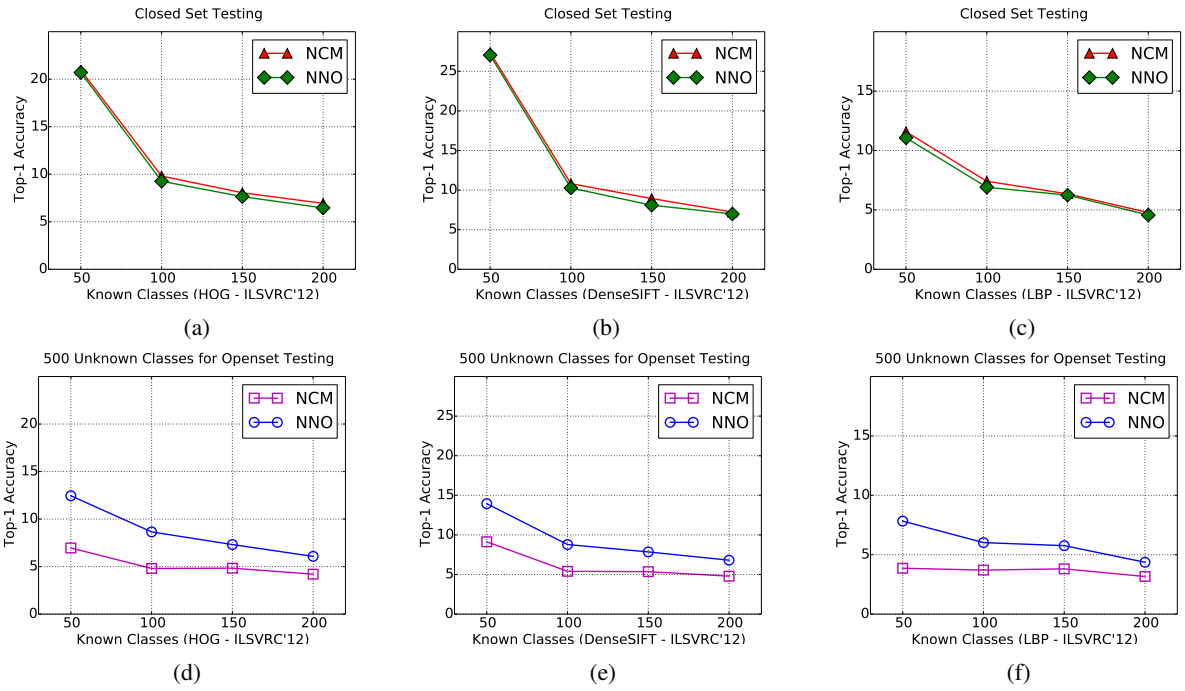

Figure 3: The above figure shows experiments on ILSVRC'12 data with 50 classes used for metric learning. The top row shows performance on closed set testing and bottom row shows performance on open set testing with 500 unknown categories. Figs 3a, 3d are for HOG features [2], figs 3b, 3e are for DenseSIFT features [6] and figs 3c, 3f are for LBP features [7]. The training data for ImageNet'12 was split into train (70%) and test split (30%). This is similar to experiment shown in fig 3a in the main paper. The absolute performance varies from feature to feature, however we see similar trends in performance as we saw on ILSVRC'10 data.

and quantized into 300 visual words by k-means. The features used in the main paper are similar to these features, except in the main paper, denseSIFT features were quantized into 1000 visual words by k-means.

2. **Histogram of Oriented Gradients (HOG):** HOG features are used in wide range of visual recognition tasks [2]. HOG features are densely extracted on a regular grid at steps of 8 pixels. HOG features are computed using code provided by [4]. This gives a 31-dimension descriptor for each node of the grid. Finally, the features are quantized into 300 visual words by k-means.

3. **Local Binary Patterns (LBP):** Local Binary Patterns (LBP) [7] is a texture feature based on occurrence histogram of local binary patterns. It has been widely used for face recognition and object recognition. The feature dimensionality used was 59.

Results on HOG [2] are shown in 3a, 3d, on DenseSIFT [6] are shown in 3b, 3e and on LBP features [7] are shown in 3c, 3f respectively. The absolute performance with DenseSIFT features is the best, followed by HOG and LBP. The DenseSIFT is very similarly to the results on ILSVRC 2010. Moreover, from these experiments we observe similar trends across all features to the trends seen in Fig 3 in the main paper. We see that as closed set performance of NCM and NNO is comparable while NCM with open set suffers significantly when tested with unknown set of categories. We continue to see significant gains of NNO over NCM with open set testing across HOG and denseSIFT features. We also observe the trend where as we add more categories in the system, the closed set and open set performance begin to converge. Thus, it is reasonable to conclude that the performance gain seen in terms of NNO on open set testing is not feature dependent. These observations are consistent with our observations from experiments on ILSVRC'10 data.

3. Algorithmic Pseudocode for Nearest Non-Outlier (NNO)

In this section, we provide pseudocode for Nearest Non-Outlier algorithm as described in section 4.1 in the main paper. The algorithm proceeds in multiple steps. In the first step, features are normalized by the mean and standard deviation over the starting subset. The initial set of

Algorithm 1 Nearest Non-Outlier Algorithm

Require: X_k, μ_k ▷ Initial Training Data X_k from k categories and their means μ_k
  function METRICLEARN(X_k, μ_k)
    W = NCMMetricLearn(X_k, μ_k) ▷ Train NCM Classifier
    for $i = 1 \rightarrow m$ **do** ▷ Over multiple folds
      X_{k_K}, X_{k_U} = SplitKnownUnknown(X_k) ▷ Split Training Data into known and unknown set
      τ_i = OpenSetThresh(X_{k_K}, X_{k_U}) ▷ Estimate optimal τ_i for each split
    end for
    $\tau = \frac{1}{m}\sum_{i=1}^{m} \tau_i$ ▷ Use average τ
    NNOModel$_k$ = $[W, \mu_k, \tau]$
  end function
Require: NNOModel$_k$, X_n, μ_n ▷ Add additional data X_n from n categories with means μ_n
  function INCREMENTALLEARN(NNOModel$_k$, X_n, μ_n)

    NNOModel$_{k+n}$ = $[W, [\mu_k, \mu_n], \tau]$ ▷ Update model with means μ_n
  end function

features is used to perform metric learning. Following this step, threshold τ for open set NNO is estimated using per class decisions using per Eq. 8 in the main paper and a cross class validation procedure of [5] training data splits. The complete pseudocode is given in Alg 1

4. Liblinear 1-vs-set extension Baseline

For this paper, we needed a baseline from an existing open set algorithm, so the performance of NNO was properly placed in context. Unfortunately, the 1-vs-all nature of the computations used in the various libsvm open set extensions developed to date ([10], [9], and [5]) make them very expensive for use in the scale of experiments in this paper. We tried the linear 1-vs-set machine of [10] and abandoned when it was not done with one fold of the smallest experiment after 3 weeks of computing. Recognizing that to scale we needed a more efficient implementation we adapted the concept/code from [10] into a liblinear ([3]) implementation. While still a 1-vs-all implementation, the liblinear library uses a much more efficient algorithm for estimation of the hyperplane than the libsvm and reduced the computation from more than 30,000 minutes (3weeks) to about 5 minutes for processing one fold of 50 classes. The default liblinear solver (L2-regularized logistic regression (primal)) did not converge so we switched to L2-regularized L2-loss support vector classification (primal).

A second issue we address in the extension is supporting searching for the "pressure" parameters discussed in the original 1-vs-set paper [10]. There is a parameter for the near plane, close to the negative data and a second parameter for the far plane. That paper provided no formal process for setting the pressures, saying it is problem dependent. Initially, we used strict 1-vs-set slabs with "-G 0 1" and the performance was quite weak with only about 5% top-1 accuracy on closed set testing. The out of the box liblinear was doing much better. Upon examination we found that liblinear was choosing the class with largest score, even if that score was negative. Thus, we needed to move beyond the default 1-vs-set machine parameters and add some near pressure to capture the negatives. The obvious process is a grid search of parameters using training data. For that process, we choose to mimic the process for selection of τ in the NNO algorithm. In particular we split the 50 classes into 3 folds of 32 classes each: first 32 classes, last 32 classes and first 16 + last 16). We trained a 1-vs-set machine with both near and far pressure and then used them to predict performance on the full 50 class training sample. Among the parameters that achieved at 90% or better recall on at least one fold, we choose the set of parameters that optimized F-measure. Our grid search included: near pressure = [0 .05 .1 .2 .3 .4 .5 .6. 7. 8. .9 1 2 3 4 5 6] and far pressure = [1 2 3 4 5 10 15 20 25 30 35 40 45 50 55 60 65]. We found the optimal parameters to be near pressure =.2, and far pressure = 55, i.e. a small expansion inside the margin space but a large expansion on the back side of the training data. We used the same parameters for 200 classes.

With that many parameters to grid search, we decided that retraining a new model with liblinear, even though it is efficient, was still computationally expensive at 56 hours for our grid search. Recognizing that other researchers may face a similar issue, we decided to develop a more efficient way to search. We adapted the liblinear 1-vs-set extension to support a mode that reads an existing liblinear model (raw or 1-vs-set), as well as the training data and 1-v-set parameters such as pressure, then computes the optimized 1-vs-set data from that starting model. The result is that for one fold it takes about 7 seconds to train model for a given set of pressures and 2 second to predict with that model on the full training data. Thus the full grid search of 225 values and 3 folds, was reduced to under 2 hours. This mode allows any-

one that has liblinear models to quickly adapt existing models for 1-vs-set testing without the costs of retraining the original model. The 1-vs-set liblinear extension discussed in this section is available as open-source from https://github.com/Vastlab/liblinear.git.

5. Numerical Values for Results

In this section, we provide numerical values for clarity. The following table shows numerical values for the surface plot presented in Fig 3a in the main paper for each algorithm. Metric learning/parameter estimation was performed on 50 categories followed by updating classifier with 50 additional categories. As 1vSet and SVM do not have incremental learning capabilities, results with only varying number of unknown categories in testing are shown.

NCM		# Known Categories in Training			
		50	100	150	200
# Unknown Categories in testing	0	20.5600	9.3667	9.4400	9.1033
	100	6.8533	4.6833	5.6640	6.0689
	200	4.1120	3.1222	4.0457	4.5517
	500	1.8691	1.5611	2.1785	2.6010

Table 1: NCM with metric learning performed on 50 categories

NNO		# Known Categories in Training			
		50	100	150	200
# Unknown Categories in testing	0	19.8933	8.0000	8.5600	8.4267
	100	12.2178	7.5700	7.9440	7.8267
	200	10.0267	7.2667	7.5752	7.5667
	500	9.0412	7.5656	7.7015	7.6867

Table 2: NNO with metric learning performed on 50 categories

1vsSet		# Known Categories in Training
		50
# Unknown Categories in testing	0	16.0267
	100	13.5689
	200	11.2827
	500	9.0412

Table 3: 1vSet algorithm tested with varying number of unknown categories. Model trained on 50 categories

The following tables show results shown in Fig 3b in the main paper. Metric learning/parameter estimation was performed with 200 initial categories. Similarly, we provide results for 1vSet and SVM algorithms with varying number of unknown categories. Incremental learning results for 1vSet and SVM are not provided as they do not possess those capabilities.

SVM		# Known Categories in Training
		50
# Unknown Categories in testing	0	21.12
	100	7.04
	200	4.224
	500	1.92

Table 4: SVM algorithm tested with varying number of unknown categories. Model trained on 50 categories

NCM		# Known Categories in Training			
		200	300	400	500
# Unknown Categories in testing	0	22.6133	10.1400	9.3300	7.2987
	100	12.4089	2.3550	2.6640	2.7489
	200	9.3067	1.8840	2.2200	2.3562
	500	5.3181	1.1775	1.4800	

Table 5: NCM with metric learning performed on 200 categories

NNO		# Known Categories in Training			
		200	300	400	500
# Unknown Categories in testing	0	22.4033	9.1000	9.2833	7.0307
	100	17.7378	6.3933	5.8520	5.3478
	200	16.2900	7.4627	6.8178	6.2276
	500	12.4343	7.3167	6.8926	1.6493

Table 6: NNO with metric learning performed on 200 categories

1vsSet		# Known Categories in Training
		200
# Unknown Categories in testing	0	14.0933
	100	12.4044
	200	11.6617
	500	10.8448

Table 7: 1vSet algorithm tested with varying number of unknown categories. Model trained on 200 categories

SVM		# Known Categories in Training
		200
# Unknown Categories in testing	0	19.25
	100	12.8333
	200	9.625
	500	5.5

Table 8: SVM algorithm tested with varying number of unknown categories. Model trained on 200 categories

References

[1] H. Agrawal, N. Chavali, M. C., Y. Goyal, A. Alfadda, P. Banik., and D. Batra. Cloudcv: Large-scale distributed computer vision as a cloud service, 2013. 2

[2] N. Dalal and B. Triggs. Histogram of oriented gradient for object detection. *CVPR*, 2005. 3

[3] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008. 4

[4] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE TPAMI*, 2010. 3

[5] L. P. Jain, W. J. Scheirer, and T. E. Boult. Multi-class open set recognition using probability of inclusion. In *Computer Vision–ECCV 2014*, pages 393–409. Springer, 2014. 4

[6] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *CVPR*, 2006. 2, 3

[7] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE TPAMI*, 2002. 3

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge, 2014. 1

[9] W. Scheirer, L. Jain, and T. Boult. Probability models for open set recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on (T-PAMI)*, 36(11):2317–2324, Nov 2014. 4

[10] W. J. Scheirer, A. Rocha, A. Sapkota, and T. E. Boult. Towards open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 36, July 2013. 4