GLOBECOM 2021 - 2021 IEEE Global Communications Conference | 978-1-7281-8104-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/GLOBECOM46510.2021.9685826

ATTL: An Automated Targeted Transfer Learning with Deep Neural Networks

Sayyed Farid Ahamed, Priyanka Aggarwal, Sachin Shetty
Virginia Modeling, Analysis and Simulation Center
Old Dominion University
Virginia, USA
{saham001, paggarwa, sshetty}@odu.edu

Erin Lanus
Hume Center
Virginia Tech
Virginia, USA
lanus@vt.edu

Laura J. Freeman
Dept. of Statistics
Virginia Tech
Virginia, USA
laura.freeman@vt.edu

***Abstract*—Success of machine learning algorithms hinges on access to labeled dataset. Obtaining a labeled dataset is an expensive, challenging and time-consuming process, leading to the development of transfer learning (TL) methodology. TL incorporates gained knowledge from a previously trained source model into specific yet similar task models with limited data domain coverage. In this paper, we propose an automated targeted transfer learning (ATTL) method to resolve the transferability between source and target with minimal data requirements. The ATTL method decides how much target data is essential for model training, along with selected source data, to obtain the skateholder's specified performance metrics. The ATTL framework optimizes the system to select minimal target data based on two approaches: combinatorial coverage and adaptive selection methodology, along with specific source data for fine-tuning given a pre-trained source model. We evaluated the ATTL method on the Kaggle's 'planes in satellite imagery' dataset and the results identified that acquiring a small number of intentionally well-chosen samples from the target environment can achieve model performance of 97% in comparison to the baseline transfer learning accuracy of 92%.**

***Index Terms*—Transfer learning, Deep neural networks, Automated source selection, Fine-tuning, Combinatorial coverage.**

I. Introduction

Machine learning (ML) systems have the ability to automatically gain knowledge and to improve from experience without being explicitly programmed. ML systems have impacted numerous fields such as computer vision, speech recognition, intrusion detection, text classification, distributed networks, sensor networks, semantic analysis and natural language processing [1], [2]. However, the effectiveness of ML systems depends on availability of large-scale, high-quality, well annotated and categorized data (e.g., in medical imaging field) which can be challenging and limits the applicability of deep convolutional neural networks (CNNs) to tasks with limited resources [3], [4].

Transfer learning solves these challenges by transferring knowledge from a related source task into a target task while compensating for the lack of sufficient task specific data [5]. Unlike traditional ML learning where model training is isolated and occurs purely based on a specific task, TL capitalizes on previously learned knowledge (in form of weights and features) by exploring its relatedness and gains additional insights into the target [6]. It has also been classified in terms of source and target structure and behavior, where structure concerns the sample spaces of the domain and task and behavior represents their probability distributions [7]. TL has been implemented using deep learning methodologies such as CNN, which has layered architecture to learn features (e.g., lines, shapes, corners) by applying features derived from the pre-trained source model [8]. It readily generates domain-invariant features for knowledge transfer between domains. Currently, very little fundamental research is being done to select a source for a given target despite the increasing availability of pre-trained networks. A deep learning framework has been proposed that will automatically perform transfer learning by emulating source network based on an estimate of dataset classification difficulty [9]. In another article, the author investigated a system to automatically rank source CNN models accordingly for a target task [10]. Fine-tuning is another popular way of utilizing knowledge in a pre-trained network for a new domain, but selecting fine-tuned layers automatically from a pre-trained network is a difficult task. Recently, some adaptive fine-tuning methodology such as SpotTune [11], Flex-tuning [12], AdaFilter [13] has been proposed where a lightweight policy network automatically finds the optimal fine-tuning decision for the target input. TL is also used to detect unknown cyber-attacks. In the article [14], the author proposed a transfer-learning based approach named HeTL that can automatically find the relation between the new attack and known attack in the cyber domain.

However, for meeting desired performance metrics, there is a need to further enhance the TL process with the ability to automatically determine additional target data needed from training and by fine tuning of the source model. In this paper, we propose ATTL approach based on combinatorial coverage and adaptive selection method, followed by fine tuning phase to address the above challenge of identifying the optimal set of features needed for transferability between source and target. The combinatorial coverage approach helps identify the unique features of target data not present in the source via the t-way value combination set differences between source and target metadata. The key idea behind the adaptation of combinatorial coverage to transfer learning is that metadata associated with the source and target domains is useful for systematically identifying the training domain of the model (source data) and the intended application domain (target data). Taking the set difference combinatorial coverage illuminates the differences between these domains. Adaptive selection framework is based on procuring low-level features from a deep model to measure of similarity between source and target based on distance metrics. We leverage pre-trained source CNN model's weighted layers to extract both source and task low-level features, compare their distributions to obtain the nearest neighbors for each target image, and re-train the model on selected source and target images. Low-level features are selected as they are general, encode very rich information which can completely reconstruct the original image and can outperform that using high-level semantic information as in [6].

II. Methodology

1) Satellite Images Dataset: To establish the transfer learning problem, we consider a situation where we train a model on the source satellite images and transfer it to the target satellite images with the task of determining whether a plane exists in the image or not. We use the "Planes in Satellite Imagery" dataset from Kaggle [15]. The images are separated by location (Northern California and Southern California). From these two sets of images, we identify the Southern California images (21,151) as the source dataset and Northern California images (10,849) as the target dataset due to the drop in performance in zero-shot transfer learning. The satellite imagery dataset has 32,000 colored images of size 20 by 20 pixels, out of which 8,000 images are "class=1" indicating that a plane occurs in the image, and the remaining 24,000 are "class=0" indicating no-plane occurs in the images. The combinatorial coverage approach is applicable to both data and metadata. Metadata is often cataloged during data collection. For images such as these, this may include categories like time of day, viewing angle, or geography. Due to the sparse metadata affiliated with this dataset – solely the general location of the images – twelve additional metadata are derived from the images to test the combinatorial coverage approach. Although the images are closely related, the combinatorial coverage over the derived metadata is different for the two locations and is discussed in § III. Fig. 1 illustrates the TL and fine tuning process on the satellite image dataset.

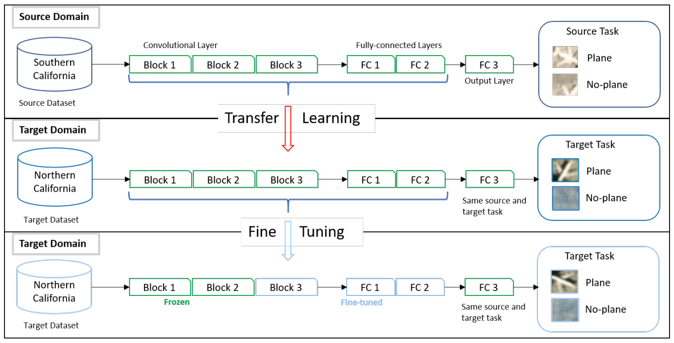

Fig. 1. Transfer learning and fine-tuning methodology.

2) Set Difference Combinatorial Coverage Method: Combinatorial coverage is a metric [16] for conventional software and hardware testing to describe the proportion of component combinations *covered* by a given test set. It is first adapted for use in testing ML systems in [17]. The authors also develop a related metric, set difference combinatorial coverage, and discuss applicability of these metrics to transfer learning. We provide a brief overview of the pertinent terms and set differencing metric in the context of describing the input space of metadata features. Let $\mathcal{S}$ and $\mathcal{T}$ be datasets with k metadata columns each. When the dataset includes labeled images, labels can be considered a special metadata column. A *t-way combination* is a selection of t of the metadata columns while a *t-way value combination* is an assignment of values to those t columns with strength t. To apply combinatorial coverage, continuous values must be discretized via a binning scheme applied to both sets. Let $\mathcal{T}_t$ and $\mathcal{S}_t$ be the set of t-way value combinations in $\mathcal{T}$ and $\mathcal{S}$, respectively. The set difference $\mathcal{T}_t \setminus \mathcal{S}_t$ is the set of t-way value combinations appearing in $\mathcal{T}$ that do not appear in $\mathcal{S}$. With $|\mathcal{T}_t|$ denoting set cardinality, the set difference combinatorial coverage of $\mathcal{T} \setminus \mathcal{S}$ is the proportion of t-way value combinations unique to $\mathcal{T}_t$, written $SDCC_t(\mathcal{T} \setminus \mathcal{S}) = \frac{|\mathcal{T}_t \setminus \mathcal{S}_t|}{|\mathcal{T}_t|}$. When labels are available, *label centrism* is a special way of counting combinations requiring that every t-way combination include the label column and $t-1$ other metadata columns; *non-label centrism* is when all $\binom{k}{t}$ combinations of columns are considered. *Label exclusion* excludes the label column entirely and considers the other $\binom{k-1}{t-1}$ combinations.

When metadata is collected alongside images, value combinations of metadata describe contexts of strength t in which images are collected. Some metadata features act as a surrogate for features in the images themselves; for example, "time of day" is a surrogate for lighting effects such as brightness or shadowing in outdoor images. Combinations of metadata describe contexts in which images are taken; for example, "location=Australia," "time of day=midnight," and "month of year=July" results in dramatically different images than "location=Greenland," "time of day=midnight," and "month of year=July" due to the combination of climate, hemisphere, and latitude differences. Label centrism, then, describes contexts in which a labeled object is seen by the algorithm. Higher strength contexts are more specific.

In addition to providing a quantitative measurement of how well the source set $\mathcal{S}$ covers the target set $\mathcal{T}$ in the input space of the metadata, the set difference provides an image selection method for targeted retraining. The set difference $\mathcal{T} \setminus \mathcal{S}$ is computed to identify the value combinations of a given strength t appearing in the target but not the source. These are pertinent as it represent contexts where the model may exhibit poor performance in transfer learning due to lack of training. The set of images from the target containing any value combinations in the set difference are identified.

3) Adaptive Selection Methodology: In this section, we discuss how to identify subset of images from source data whose low-level characteristics are similar to those of the target task.

Only these selected subset of source and target images will be made accessible for retraining the target specific CNN model. The rational for selecting low-level features is its ability to encode rich and varied information that can reconstruct the original image with the added benefit of preventing overfitting inherent in the fine tuning process [6]. The source domain $D_s = (x^s_j, y^s_j)^{n_s}_{j=1}$ constitutes of 21,151 images (as explained above) taken in Southern California, where x_s is set of images with labels y_s. The target domain $D_t = (x^t_l, y^t_l)^{n_t}_{l=1}$ encompass 10,849 images from Northern California leading to a different data distribution transfer learning problem. A CNN model is trained on the source dataset to form the pre-trained CNN model. Using this pre-trained CNN module, we extract the histogram of features obtained from the CNN low-level layers for each source and target image and evaluate their similarity measure. Let $\phi^{j,s}_i$,$\phi^{l,t}_i$ represent the histogram of the features extracted from the i_{th} convolutional layer for source image x^s_j and target image x^t_l respectively. The distance between the histograms of the kernel response between a source and target image is calculated by eq. (1).

$$H(x^{l,t}, x^{j,s}) = \sum_i w_i[K(\phi^{j,s}_i, \phi^{l,t}_i) + K(\phi^{l,t}_i, \phi^{j,s}_i)] \quad (1)$$

where $w_i = 1/N_i$, N_i is the number of convolutional kernels in the corresponding layer i and K is the KL divergence between distribution $\phi^{j,s}_i$ and $\phi^{l,t}_i$(eq. (2)).

$$K(\phi^{j,s}_i, \phi^{l,t}_i) = \sum_x \phi^{j,s}_i(x) log \frac{\phi^{j,s}_i(x)}{\phi^{l,t}_i(x)} \quad (2)$$

For a source image, if the calculated KL distance H meets the specified threshold, it is selected as one of the neighbors for the target image. Depending on the threshold, a target image can have none, one or more nearest neighbors. The target images, which have no match in the source domain, are called the no-matched or hard samples and are augmented to the training set. The selected source images and the hard samples form the new training set for the CNN model. The maximum selected source images for a target image are capped to prevent system bias. Efforts are made to incorporate the least target images as possible, with or without labels. In the case where target labels are incorporated in source selection process, its called supervised selection process and where labels are not used, unsupervised selection.

4) Automated Targeted Transfer Learning: We propose an automated targeted transfer learning process to quantify the performance of the model in a given environment. Fig. 2 shows the flowchart of this process. Here, in each case, the ATTL process checked whether the model met the desired metric requirement or not? If the result is yes, the process stops and the operator obtains the desired results; otherwise it will continue.

Let us consider that the stakeholder's/ operator's desired performance metrics in the target environment are precision, recall, f1-score and accuracy. In case one, a model is trained on the source set and tested on the target set, which is

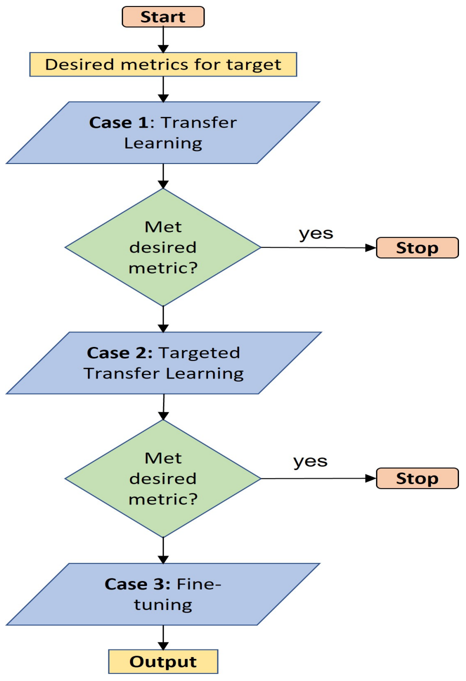
Start
Desired metrics for target
Case 1: Transfer Learning
Met desired metric?
yes
Stop
Case 2: Targeted Transfer Learning
Met desired metric?
yes
Stop
Case 3: Fine-tuning
Output

Fig. 2. Automated targeted transfer learning flowchart.

the conventional transfer learning method where the trained model never interacts with the target set. In this paper, the source and target datasets are derived from the same "satellite plane images" but located at two different locations. If the operator does not get the required metrics from case one, case two (targeted transfer learning) will be applied. In this case, we first select the subset of the images that have value combinations in the set difference and create new train and test sets. Here, we use the combinatorial coverage method to create the new sets. Another method of selecting images based on similarity measure (KL metrics) is also incorporated in the paper. After selection, the selected images will be added to the source set for training and use the remaining target set as a test set. Details of this case is described in the result and discussion section. If the previous two cases do not meet the desired operator metrics (transfer learning, targeted transfer learning), the last case – fine-tuning process – is initiated. After this process, the stakeholder will obtain the best accuracy in any given environment. Here, emphasis is placed to achieve the desired performance metrics with the minimal amount of target training images. During the system execution, the operator will deduce the minimal number of images that should be transferred from target to source domain to achieve the optimum level. Through our ATTL process, we are able to answer three key questions: Can a pre-trained model, trained on the source domain, provide desired metrics when applied on the target domain (case 1 – transfer learning)? If not, can the same model be trained on source and selected target images to meet the desired metrics (case 2 – targeted transfer learning)? If these two processes cannot provide the desired accuracy, can fine-tuning the pre-trained model on the minimal target images give the desired accuracy (case 3 – fine-tuning)?

III. Experimental Results

Only original metadata (i.e. geo location data) is used to partition the Planesnet [15] dataset into source and target sets. The derived metadata for the combinatorial coverage experiments are the mean and variance each for the red, green, blue, hue, saturation, and luminance values for an image. The metadata values are discretized by forming three bins encompassing equal-sized ranges. An example of a two-way label centric combination is the class and red mean. A two-way non-label centric combination could be one metadata and the class but could also include two metadata, such as red mean and saturation variance. An example of a two-way label centric value combination is "class=0" and "red mean=0" for an image with no plane and the mean red value for the image falling in the first bin. The Northern California dataset has higher label-centric combinatorial coverage than the Southern dataset as $CC_2(Northern) = \frac{67}{72} = 0.93$ while $CC_2(Southern) = \frac{60}{72} = 0.83$; of the 72 valid value combinations of metadata, the Northern dataset contains 67 while the Southern contains 60. So despite the Southern dataset having roughly double the number of images (21,151 versus 10,849), the Northern dataset has more complete coverage of the metadata input space and is considered as the target domain. When the Southern set is treated as the source and the Northern set as the target, a drop in performance is noted for zero-shot transfer. For this direction, the label centric set difference combinatorial coverage is $SDCC_2(Northern \setminus Southern) = \frac{8}{67} = 0.12$. That is, of the 67 two-way label centric value combinations of metadata present in the Northern dataset, eight of them do not appear in the Southern dataset. The drop in performance does not occur when trained on the Northern set and transferred to the Southern set and, notably, $SDCC_2(Southern \setminus Northern) = \frac{1}{60} = 0.02$. That is, of the 60 two-way label centric combinations present in the Southern dataset, only one does not appear in the Northern dataset. In summary, of the Southern contexts in which a plane or no-plane is seen by the algorithm, most are covered by the Northern set, but many Northern contexts are not covered by the Southern set. For this dataset, $SDCC_2$ is correlated with a drop in performance in zero-shot transfer learning. We can calculate these set differences at higher values of t to define differences between the sets at higher strengths of combinations of the metadata factors.

However, as mentioned earlier in the methodology section, our proposed automated targeted transfer learning (ATTL) process is divided into three cases and all cases are described briefly below. We present the result of each case individually with minimal data requirements to achieve the desired performance. In targeted transfer learning case, both combinatorial coverage and adaptive selection methods are used to identify the set difference images. To validate the performance, targeted transfer leaning case is repeated for random dataset from target. Lastly, a fine-tuning is applied on the remaining dataset.

A. Result analysis

1) Transfer learning process - Case 1: The described CNN model is trained on source (Southern California) set and tested on the target (Northern California) set, which is the conventional/baseline transfer learning process. The model has been trained only on the source dataset with 90% training and 10% validation images. All pre-trained model parameters are kept frozen while predicting the target label.

learning graph.JPG

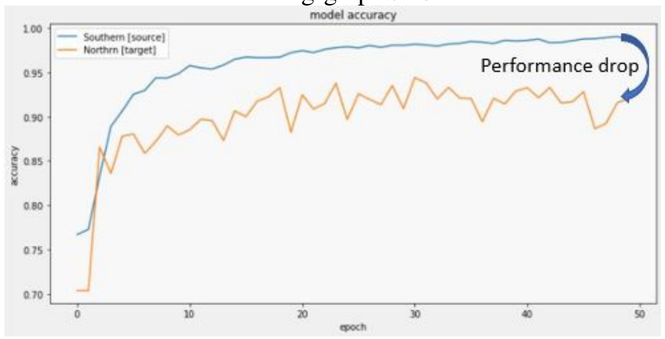

Fig. 3. Drop in performance (Transfer Learning).

The CNN model gave an accuracy of 98% when source data was used for both training and validation. However the accuracy significantly drops to 92.19% when the model is tested on target data under zero-shot transfer scenario as illustrated in 3. The zero-shot accuracy forms the baseline accuracy in this study. The obvious question arises: how can we improve the performance if we are limited in our ability to collect a large amount of data in target domain? Below are automated attempts to resolve this problem of a drop in metrics through targeted transfer learning training.

2) Targeted Transfer Learning - Case 2: Targeted transfer learning will automatically be applied when the operator does not get the required metrics from the basic zero-shot transfer learning implementation. In this case, set difference combinatorial coverage is used to identify images for inclusion in the augmented training set. For the adaptive selection case, only selected source images along with selected target images will form the training set while in combinatorial coverage all source data is used. The model's accuracy will then be investigated on the newly formed dataset by basic transfer learning process.

Combinatorial Coverage: As described in § II-2, we compute the set difference $Northern \setminus Southern$ for strength t. Every image in the Northern set containing a value combination from the set difference in its metadata is selected for inclusion in the targeted images and added to the augmented training set; call this *targeted selection*. A model is trained using the augmented set of images. The selected images are removed from the target set and the remaining images in the target set forms the test set. Our experiments vary two parameters: the strength $3 \leq t \leq 5$ and the counting method as LC (label centric) or NLC (non-label centric). We also consider the possibility that any improvement is simply due

TABLE I
TARGETED SELECTION VS. RANDOM SELECTION

		Value	#	Targeted		Random	
t	Count	Comb.	Images	AC	IF	AC	IF
3	LC	1188	150	94.22	0.014	94.12	0.013
3	NLC	7128	240	95.14	0.012	94.41	0.009
4	LC	11880	395	96.47	0.011	94.90	0.007
4	NLC	51975	758	96.49	0.006	95.36	0.004
5	LC	80190	879	97.08	0.006	96.15	0.005

to a larger training set and not due to the targeted selection of images via combinatorial coverage. For each setting of strength and counting method, we perform random sampling to select the same number of images and train a model on the augmented set. Again, the selected images are removed from the target set and the test set forms the remainder. Due to randomness in training a model that can result in variation in outcomes, we replicate the above process three times; for every set of images added to the augmented training set, we train three models and evaluate each on the test set. The appendix includes all metrics for each of the three runs for targeted selection (Table IV) and random selection (Table V). As precision, recall, and F1-score trend with accuracy, we focus our analysis on accuracy.

For t=2, the number of targeted images identified is less than 100 which is a negligible number of images to transfer. Therefore, we trained models for $3 \leq t \leq 5$. Table I gives the results of targeted transfer learning in terms of number of targeted images identified for each setting of strength and counting method, as well as the number of valid value combinations under consideration. It also gives the mean accuracy (AC) of the three models for each set of targeted images and randomly selected images (of the same size), as well as a measure of the improvement in accuracy, i.e. a ratio of number of images provided, or *information factor per image* (IF) computed by

$$\text{IF} = \frac{\text{accuracy} - \text{baseline accuracy}}{\text{number of images}}$$

We use the accuracy (92.19%) from zero-shot transfer as the baseline accuracy. Accuracy and IF are plotted in Fig. 4 and Fig. 5, respectively. Several general trends are observed:

1) as t increases, so does the number of value combinations and the number of images targeted,
2) as the number of images to include in training increases, so too does accuracy,
3) as the number of images increases, the IF decreases, and
4) targeted selection results in higher accuracy and higher IF for the same number of randomly selected images.

The trends are not surprising. As value combinations increase in strength, they become more specific and more numerous, thus it becomes less likely that an image will contain a given value combination. For set differencing, as the strength increases, the set difference tends to increase; it becomes less likely that the two sets will contain exactly the

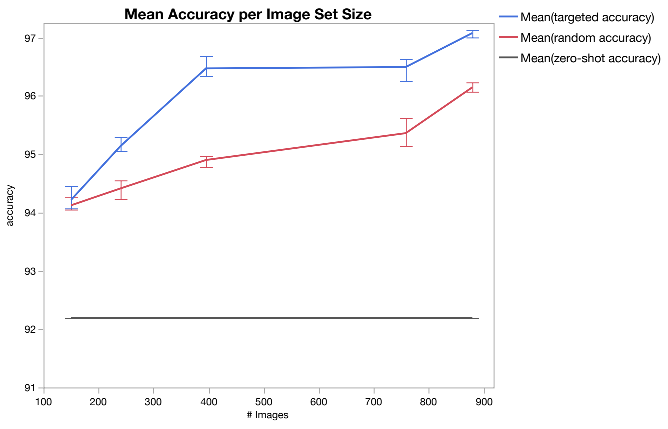

Fig. 4. Accuracy by image set size (error bars are min, max)

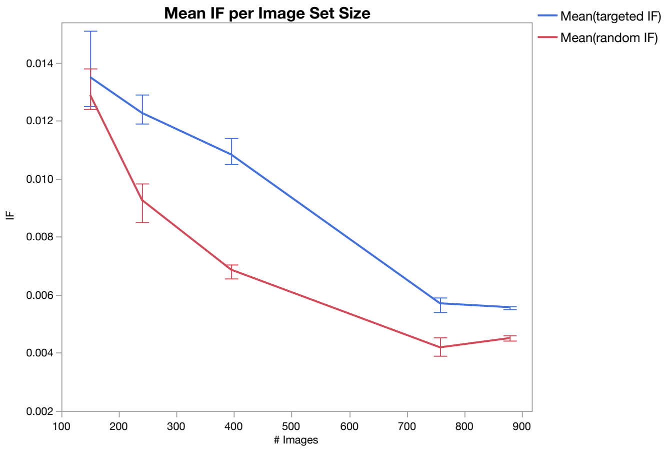

Fig. 5. IF by image set size (error bars are min, max)

same value combinations. The increased set difference size allows us to find more images containing value combinations in the set difference.

In our experiments, when $t = 4$, LC produces nearly the same mean accuracy as NLC despite identifying approximately half as many images for inclusion in training. In fact, LC produces higher minimum and maximum accuracy scores than NLC. The difference in IF between LC and NLC for $t = 4$ ($\Delta = 0.00146$) is also much smaller than that for random selection for the same number of images ($\Delta = 0.00240$). From a combinatorial perspective, for the same strength, LC considers fewer combinations of metadata columns and fewer value combinations than NLC which supports why it identifies fewer images. For example, in this dataset with 12 metadata columns each with three binned values and one label column with two binned values, $k = 13$. For $t = 4$, the number of combinations is $\binom{k-1}{t-1} = 220$ for LC and $\binom{k}{t} = 715$ for NLC, while the number of value combinations is $\binom{k-1}{t-1}(2)(3^{(t-1)}) = 11,880$ for LC and $\binom{k-1}{t-1}(2)(3^{(t-1)}) + \binom{k-1}{t}(3^t) = 51,975$ for NLC. As LC for t is the same as considering every $t-1$ label exclusive combination along with the label column, it resides between NLC strengths. From an ML perspective, we hypothesize that LC provides better targeted transfer learning by focusing on contexts surrounding labels which is central for classification.

The IF decreasing trend suggests diminishing returns with additional images. This may be due to achieving sufficient coverage over the input space. Additionally, the IF for LC at t is closer to that of $t-1$ NLC than t, again suggesting that LC may be a useful counting method for targeted image selection. The decreasing IF rate may provide feedback in the automated process to guide whether to continue with targeted transfer learning or switch to fine-tuning.

Last, the improvement in accuracy does not seem to be attributable to increased training set size alone. Targeted image selection using $t = 4$ and LC produces higher mean accuracy with 395 images than randomly selecting 879 images. The targeted selection achieves 96.47% mean accuracy while random selection achieves only 96.15% with over twice as many images. This suggests that targeted selection using set difference combinatorial coverage over the input space is a useful mechanism for producing sets for minimal retraining.

In the automated process, suppose the desired operator metric is 96% accuracy. From our results, $t = 4$ LC evaluation provides the desired accuracy with the fewest number of targeted images. As we get the desired performance metrics from the targeted transfer learning by coverage theory, our automated process will stop; otherwise it will continue to the fine-tuning process.

3) Targeted Fine Tuning - Case 3 : The automated system will check if the desired performance is met or not met. In case its not, it will proceed with the fine-tuning process. During fine-tuning, we keep the first two convolutional layers of our CNN model frozen and fine-tune the last few layers with the lowest labeled target images. For fair comparison, efforts are made to keep the selected target images fixed (total of targeted and fine tuning images) by varying the number of training and validation images in the model. In this paper, we fixed the number of total selected targeted images to be nearly 20% of the complete target dataset, consists of 10,849 images. For example, if the selected images for targeted case is 158 then the training images for fine-tuning portion would be 1,937, to make the total 2,095 target images be around 20% of the target set. We examine the results for adaptive selection framework process.

TABLE II
SUPERVISED FINE-TUNING RESULTS.

Dist.	Unmatch Img.	Fine-Tune Img.	Total Img.	Accuracy
3	460	1632	2092	96.156
4	243	1853	2096	95.968
5	158	1937	2095	95.988
6	112	1983	2095	95.819
7	82	2008	2090	95.445

Table II illustrates the results for the adaptive selection process for the supervised scenario. Distance column indicates the defined distance H calculated by eq. (1). Unmatched images are the target images that did not have any match or similar source images. These images are added to the targeted training set in case 2. The training images used for fine-tuning the model are given in column 3 while the total target images are in column 4 of the table.

TABLE III
UNSUPERVISED FINE-TUNING RESULTS.

Dist.	Unmatch Img.	Fine-Tune Img.	Total Img.	Accuracy
3	334	1755	2089	95.287
4	158	1937	2095	95.214
5	90	2006	2096	95.247
6	59	2035	2094	95.157
7	36	2055	2090	94.851

The unsupervised results where labels are not used in the source selection process are given in Table III. Results indicate that as the distance hyper-parameter is decreased, more and more target images are used but the overall performance matrix also increases. Further, there is a slight drop (less than 1%) in accuracy for the supervised vs. the unsupervised scenario. As the percentage of the training images increases, the performance of the system increases.

IV. Discussion and Conclusion

We have proposed an Automated Targeted Transfer Learning method for efficiently and automatically identifying samples to achieve desired results in a given environment. This paper provides the technical foundations for the targeted transfer learning process that provides the decision-makers assurance of obtaining the desired outcomes by utilizing minimal retraining and target data collection. In the future, we would like to leverage the proposed ATTL process into other machine learning platforms like cyber domain (malware dataset) and conduct a case study on how efficient the method is for minimal retraining.

The method of selecting targeted images for retraining may be further refined. For this preliminary work, all images with a value combination in the set difference were selected. The targeted images improved accuracy better than the same number of randomly selected images suggesting that covering the input space is useful for knowledge gained by the model. However, the targeted image set may include the same value combinations many times and thus include redundant coverage. In software testing where faults are deterministic, coverage of a value combination in at least one test is sufficient to detect the fault. Statistical learning rarely has the property that one observation is sufficient. It seems likely that as image set sizes increased, each additional coverage of a value combination yields diminishing returns. If this is true, a smaller subset of the target images that covers the set difference up to some threshold may increase performance equivalently with fewer images.

Lastly, in our experiments, we do not compare the models against the same test set. Instead, we model the scenario where transfer learning is conducted with retraining on a small sample of the target environment. We test on the entire remaining target minus the training augmentation to simulate deployment across all contexts in the target. If our hypothesis is correct and the set difference combinatorial coverage provides a method to identify contexts in the target not learned from the source dataset, these images that contain those contexts are where we expect performance to drop. By moving them from the target to the augmented training set and not allowing them in the test set, we are not able to measure performance on these contexts for the targeted selection, but we may test on them for the random selection. The drop in performance between targeted and random selection supports our hypothesis that these value combinations represent challenging, unlearned contexts. To test this hypothesis, we suggest that performance of the pretrained model should be measured on subsets of the target set partitioned by combinatorial coverage to determine if there is a drop in performance in subsets that contain contexts in the set difference versus those that do not. Further, a targeted set should be constructed that covers the set difference to a threshold λ without including all images in the set difference. The remaining images in the set difference should be left in the target set for inclusion in a test set.

Acknowledgment

The authors would like to thank Tyler Cody, Adam Edwards and Peter Beling for their contribution to the meta-data derivation. This work is supported by MITRE and the DoD Center of Excellence in AI and Machine Learning (CoE-AIML) under Contract Number W911NF-20-2-0277 with the U.S. Army Research Laboratory.

References

[1] G. Capuano and J. J. Rimoli, "Smart finite elements: A novel machine learning application," *Computer Methods in Applied Mechanics and Engineering*, vol. 345, pp. 363–381, 2019.

[2] R. Pandey, V. Gautam, R. Pal, H. Bandhey, L. S. Dhingra, H. Sharma, C. Jain, K. Bhagat, L. Patel, M. Agarwal *et al.*, "A machine learning application for raising wash awareness in the times of covid-19 pandemic," *arXiv preprint arXiv:2003.07074*, 2020.

[3] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into imaging*, vol. 9, no. 4, pp. 611–629, 2018.

[4] T. Kim, K. Lee, S. Ham, B. Park, S. Lee, D. Hong, G. B. Kim, Y. S. Kyung, C.-S. Kim, and N. Kim, "Active learning for accuracy enhancement of semantic segmentation with cnn-corrected label curations: Evaluation on kidney segmentation in abdominal ct," *Scientific reports*, vol. 10, no. 1, pp. 1–7, 2020.

[5] P. Marcelino, "Transfer learning from pre-trained models," *Towards Data Science*, 2018.

[6] W. Ge and Y. Yu, "Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1086–1095.

[7] T. Cody, S. Adams, and P. A. Beling, "A systems theoretic perspective on transfer learning," in *2019 IEEE International Systems Conference (SysCon)*. IEEE, 2019, pp. 1–7.

[8] O. Lucena, A. Junior, V. Moia, R. Souza, E. Valle, and R. Lotufo, "Transfer learning using convolutional neural networks for face anti-spoofing," in *International conference image analysis and recognition*. Springer, 2017, pp. 27–34.

[9] T. Balaiah, T. J. T. Jeyadoss, S. S. Thirumurugan, and R. C. Ravi, "A deep learning framework for automated transfer learning of neural networks," in *2019 11th International conference on advanced computing (ICoAC)*. IEEE, 2019, pp. 428–432.

[10] M. J. Afridi, A. Ross, and E. M. Shapiro, "On automated source selection for transfer learning in convolutional neural networks," *Pattern recognition*, vol. 73, pp. 65–75, 2018.

[11] Y. Guo, H. Shi, A. Kumar, K. Grauman, T. Rosing, and R. Feris, "Spottune: transfer learning through adaptive fine-tuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4805–4814.

[12] A. Royer and C. Lampert, "A flexible selection scheme for minimum-effort transfer learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 2191–2200.

[13] Y. Guo, Y. Li, L. Wang, and T. Rosing, "Adafilter: Adaptive filter fine-tuning for deep transfer learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 4060–4066.

[14] J. Zhao, S. Shetty, J. W. Pan, C. Kamhoua, and K. Kwiat, "Transfer learning for detecting unknown network attacks," *EURASIP Journal on Information Security*, vol. 2019, no. 1, pp. 1–13, 2019.

[15] "Kaggle Dataset," https://www.kaggle.com/datasets.

[16] D. R. Kuhn, I. D. Mendoza, R. N. Kacker, and Y. Lei, "Combinatorial coverage measurement concepts and applications," in *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation Workshops*, 2013, pp. 352–361.

[17] E. Lanus, L. J. Freeman, D. R. Kuhn, and R. N. Kacker, "Combinatorial testing metrics for machine learning," in *2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 2021, p. to appear.

Appendix

TABLE IV
Metrics Each Training Run – Targeted Selection

# Images	Precision	Recall	F1-score	Accuracy
150	94.37	94.13	93.96	94.13
150	94.57	94.45	94.32	94.45
150	94.16	91.07	93.94	94.07
240	95.10	95.05	94.97	95.05
240	95.29	95.29	95.23	95.29
240	95.16	95.08	94.99	95.08
395	96.38	96.39	96.37	96.39
395	96.36	96.34	96.94	96.34
395	96.67	96.68	96.67	96.68
758	96.24	96.25	96.23	96.25
758	96.65	96.60	96.62	96.60
758	96.64	96.63	96.63	96.63
879	97.09	97.10	97.09	97.10
879	97.12	97.13	97.13	97.13
879	96.99	97.00	97.00	97.00

TABLE V
Metrics Each Training Run – Random Selection

# Images	Precision	Recall	F1-score	Accuracy
150	94.27	94.26	94.27	94.26
150	94.08	94.05	94.00	94.06
150	94.12	94.05	93.93	94.05
240	94.52	94.46	94.37	94.46
240	94.45	94.23	94.03	94.23
240	94.55	94.55	94.46	94.55
395	94.93	94.45	94.91	94.95
395	94.80	94.78	94.71	94.78
395	95.00	94.97	94.89	94.97
758	95.38	95.32	95.25	95.32
758	95.19	95.14	95.07	95.14
758	95.65	95.62	65.56	95.62
879	96.12	96.14	96.12	96.14
879	96.08	96.07	96.03	96.07
879	96.22	96.23	96.20	96.23