Machine learning approaches for small data in sensor fusion applications

Dinesh Verma*[a], Graham Bent[b], Geeth de Mel[b], Chris Simpkin[c],
[a]IBM TJ Watson Research Center, 1110 Kitchawan Road, Yorktown Heights, NY, USA 10598,
[b]IBM Research, Hursley, SO212JN, UK
[c]Cardiff University, Cardiff, CF10 3AT, UK

ABSTRACT

Machine learning approaches like deep neural networks have proven to be very successful in many domains. However, they require training on a huge volumes of data. While these approaches work very well in a few selected domains where a large corpus of training data exists, they shift the bottleneck in development of machine learning applications to the data acquisition phase and are difficult to use in domains where training data is hard to acquire. For sensor fusion applications in coalition operations, access to good training data that will be suitable for real-life applications is hard to get. The training data sets available are limited in size. For these domains, we need to explore approaches for machine learning which can work with small amounts of data. In this paper, we will look at the current and emerging approaches which allow us to build machine learning models when access to the training data is limited. The approaches examined include statistical machine learning, transfer learning, synthetic data generation, semi-supervised learning and one-shot learning.

Keywords: machine learning, small data learning, one-shot learning, transfer learning

1. INTRODUCTION

Artificial Intelligence, and specifically machine learning holds significant promise for revolutionizing coalition operations. The major driver behind the re-emergence of machine learning has been the availability of a large amount of data for training the machine learning algorithm. The big data available in many domains enables training of deep neural networks and other models to capture patterns in the data. In domains where training data is plentiful and can be curated easily to remove imperfections for model training, the benefits of machine learning are starting to manifest themselves, e.g. in speech to text conversion.

Unfortunately, there are many domains where access to plentiful data for machine learning remains difficult to obtain. When we look at specialized domains, e.g. military ISR, coalition operations, industrial automation, etc., training data is not readily available and has to be painstakingly curated. The data available from Internet based sources is rarely representative of the workload in such domains, and it is unclear whether a machine learning system based on publicly available data sources will be applicable to these specialized domains.

Therefore, there is a need to explore machine learning algorithms that can perform well even when there is only limited amount of training data. In this paper, we present a survey of approaches that can be used for domains where the amount of training data available is limited. In order to keep the length of the paper reasonable, the survey is at a relatively high level. Nevertheless, this paper can serve as a good starting point for researchers faced with a limited amount of training data for their domain.

The approaches for small data learning can be broadly classified into the following categories:

Parameter Estimation Approaches: These approaches use domain knowledge to define a parametrized model, reducing the need for data to be the amount required to estimate the parameters adequately.

Transfer Learning: These approaches reuse models trained in one domain where sufficient training data is available and adapt them to domains where the amount of training data may be limited.

Feature Definition: These approaches rely on partial specification of the latent variables that are extracted from the input data as part of the model-building process. One shot learning proposed for image classification is an example of this approach.

Data Generators: These approaches try to generate additional training data based on the available training data to feed into the models. The field of semi-supervised learning falls within this general category.

The four approaches defined above are not mutually exclusive, and several papers in AI propose techniques which can be viewed as a combination of two or more among them. Nevertheless, the broad classification helps in understanding the techniques, and helps in the selection of an approach that can be applicable in a specific sensor fusion scenario. Before providing an overview of the techniques, it is useful to review the prevailing guidelines about the amount of training data required for machine learning.

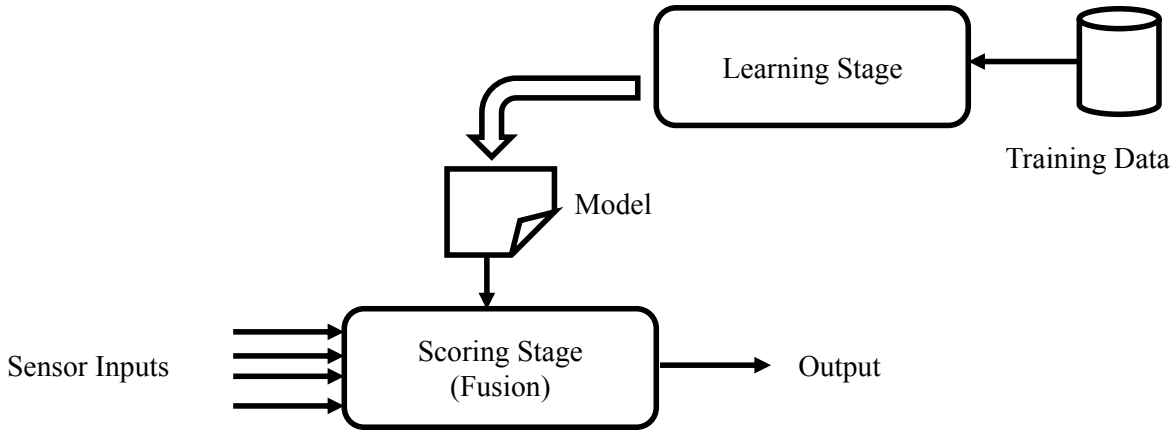

Figure 1. AI enabled approach for Sensor Fusion

2. ASSUMPTIONS AND BACKGROUND

In this section, we introduce a generic model for machine learning enabled sensor fusion applications, the process assumed for such applications and prevailing guidelines for the amount of training data that is needed.

2.1 Assumed approach for AI enabled Sensor Fusion

We assume that the AI-enabled sensor fusion algorithm works in an environment where the algorithm is receiving several sensor inputs. On receiving the different sensor inputs, the fusion algorithm produces a fused output. The fused output can be viewed as a function of the different inputs. Depending on the environment of the fusion algorithm, the functions and inputs could have a time-delay, could be continuous or could be discrete. In order to convert the inputs into an output, the fusion algorithm uses a model. The model is the data structure with enables the conversion of inputs into output. The model could be in a variety of formats, e.g. it can be a neural network, a decision tree, a set of rules, a set of Dempster–Shafer belief rules, a definition of clusters on a feature space, parameters for a Kalman-filter etc.

As shown in Figure 1, the algorithm is assumed to work in two distinct stages, a learning stage and a scoring stage. In the learning stage, the algorithm uses a set of training data to create the model, i.e. it determines the right weights of the neural network, the set of rules etc. In the scoring stage, it uses the trained model to combine different sensor inputs and to produce a fused output. The scoring stage is the actual fusion operation on different sensor inputs.

The learning stage can operate offline before any fusion or scoring happens, or it can happen concurrently online with the scoring stage. In the offline mode, approaches for supervised learning can be used, where the training data includes the expected output function along with the input functions that produces them. For simplicity in this paper, we assume that a supervised approach is used for training the model offline, and then the scoring happens in the online model. However, the discussions and approaches in this paper would remain valid even when considered in the context of an online learning process.

From a sensor fusion perspective, the easiest assumption to make is that the model is learning the function that expresses the output in terms of the input parameters. Formulating learning as a function estimation is general enough to cover a large variety of applications of sensor fusion. If the output function learnt takes only some discrete values, it can be used as a classifier. If the output function takes a binary value, it can be used to represent detection of anomalies. Thresholds applied to the output function can be used to produce tokens, which can represent applications such as speech to text, or mapping text documents into sentiment of a person. Similarly, the output function can represent a decision being taken.

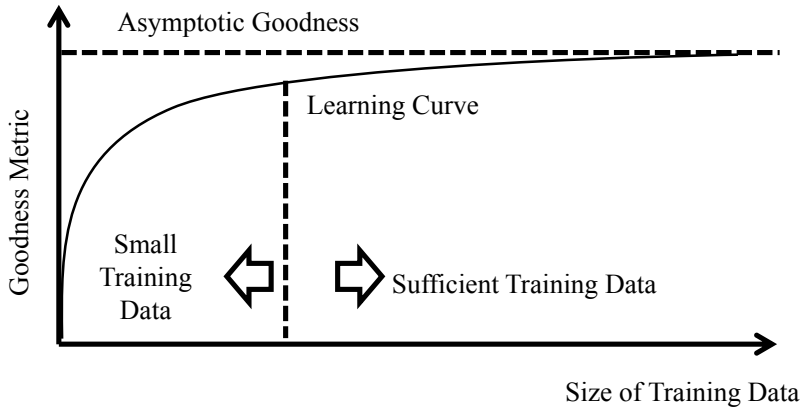

Figure 2. The Typical Learning Curve

We assume that there is a ground truth function which always produces the right output from the given inputs. The learning process provides the best approximation to that ground truth which can be determined from the training data. If the training data is collected in a manner that there is no noise in the data, i.e. the ground truth is accurately reflected, and the model building algorithms can be perfected, one can imagine estimating the ground truth without any errors. In practice, the training data often would have noise in its collection process, the model building algorithm would have its limitations, and the ground truth can only be approximated with some error. Assuming that the noise introduced has a mean error of zero, an increase in training data will help in reducing the error and improve the fidelity with which the ground truth is approximated. The fidelity would also be constrained by the limitations of the training algorithm.

2.2 Requisite amount of training data

The amount of training data that is needed for machine learning depends on many different factors, including the type of model being learnt, the characteristics of the domain, the assumptions that can be made about the domain and the model being learnt etc. There is a broad consensus that one needs sufficient amount of data, but the definition of sufficient amount of data is unclear. There are some rules of thumb available based on exploration of statistical patterns in data mining [1] which recommend that for very simple models and small number of features (less than 10), 50-100 samples per class for classification problems should be adequate. However, larger numbers of features or more complex training models, e.g. neural networks, would require significantly more number of training data samples.

Another rule of thumb recommends looking at the degrees of freedom that exist in the model being trained [2]. The degree of freedom is the number of effective independent parameters that a model needs to support, and it is recommended that the number of training data be more than 10 times the effective number of independent parameters.

The relationship between the performance metrics of a learning model (precision or recall) is usually captured by means of learning curves, which plot a metric of how good the training is from a learning algorithm as the size of the training data set increases. The metric being examined could be error rate, the Kolmogorov-Smirnov (KS) statistic, likelihood ratios, specificity and sensitivity or precision and recall etc., with different metrics being suitable for different types of learning algorithms and the specific use-case. Learning curves can be estimated using statistical models which assume some distribution among the types of features expected in the training space and the analysis of the different algorithmic properties [3]. An alternate way to estimate learning curves is by experimentation on real training data or synthetically generated data traces [4] [5].

Each of these approaches to estimate learning curves has its challenges. It is difficult to extend the results from real or synthetic data sets to learning algorithms on other data sets, and the statistical distribution assumptions made for statistical

analysis, which provide results that hold in general, are very rarely known to be satisfied in real data. While both of these approaches for estimating learning curve have limitations, they do provide useful insights into the amount of training data that is needed for a good model.

In general, a learning curve would have a shape like one shown in Figure 1 and if one has enough training data to be at a point where the learning curve is good enough, one can assume that the model that is trained on it will be a valid one. Despite the challenges of determining the right learning curve, the availability of big data in many domains has led to a situation where we can safely assume sufficient training data is available. Applications of machine learning to domains where large volumes of data are available, e.g. purchase history available to popular online merchants, search histories of queries made on popular online search engines, songs being played on popular media sites, news items being clicked on popular news sites, can safely assume that they are operating beyond the knee of the learning curve. However, for many other domains where training data may not be as plentiful, not curated well, or be difficult to collect, one has to assume one is operating in a region with small training data set and explore ways to learn when the data is not adequate. In many domains of machine learning that are of specialized industrial or military use-cases, the small training data case is more likely to be true.

The reason for the goodness metric to increase with the size of training data is the presence of noise or errors in the training data. In most real-world sensor fusion applications, there is a significant amount of noise in the process of collecting training data. If there was no noise, a small amount of training data would be able to give a fairly good estimate of the function. However, with the presence of noise, an increase in the training data helps in cancelling out the impact of noise and makes the function become closer to the ground truth. This factor explains the typical shape of the learning curve shown in Figure 2.

In this paper, we look at the different approaches for machine learning that can be used to build a good machine learning model when one is unsure if sufficient training data has been collected.

3. PARAMETER ESTIMATION APPROACHES

AI approaches like training a deep neural network require a significant amount of training data because they do not make any assumptions about the structure of the functions that they are trying to learn. This can be viewed as the black box approach where no information is assumed about the relationship between the sensor inputs and the output shown in Figure 1. An alternative approach is to assume a white-box approach, in which the relationship between the sensor inputs and the output is known. If an equation computing the output from sensor inputs is known, training is not necessary. The parametric approach is a hybrid between the black box approach and the white-box approach (which we can call a grey-box approach), where the relationship between the input and the outputs is partially known. The type of relationship is known, but some parameters in the relationship may not be known. The learning process estimates the parameters for the model from the training data that is provided.

Suppose we can assume that the sensor fusion process can be modeled as $o = f(x_1, x_2, \ldots x_N)$, where o is the output, and $x_1, x_2, \ldots x_N$ are the sensor inputs. If the function f is fully specified, we need no training data since the output can be computed readily from the inputs. However, in most cases, we may know the shape of the function, but not the complete specification. As an example, in a very simple case, we can assume that $f(x_1, x_2, \ldots x_N) = \Sigma a_i x_i$, where $a_1, a_2, \ldots a_N$ are the unknown parameters. This is a simple linear relationship, and assuming that each sample of the training data is independent and free of any noise, N samples should be sufficient to estimate the value of the N unknown parameters connected by a set of linear equations. In the real-world, there would be noise in the data, and a linear regression equation that minimized a metric for error such as least square distance from the data points, would need to be used. Such a linear regression equation can be learnt efficiently using much fewer number of points than the ones needed if no assumptions can be made. For other types of models, e.g. non linear relationships, more points are needed. However, the minimum number required for estimating the models can be determined based on the number of parameters and the shape that they have.

The approach can be illustrated using the case of one sensor input x and one output y in Figure 3. While the use of just two variables presents a rather trivial use-case, it provides a simple visualization of this approach. The solid line in figure 3(a) represents the ground truth which is the line $x^2+y^2 = 1$, and Figure 3(b) shows the dots that will be needed in the training data if we made no assumptions about the nature of the ground truth. The dots are scattered roughly across the ground truth, but we will need several of the dots to be present before we can estimate what the ground truth is. A learning method

will try to estimate the relationships which are implied in the solution, and in some sense try to guess every type of function that the points may represent. However, supposed we know that the ground truth is a relationship that satisfies the relationship that $x^n+y^n = c$ which uses two parameters n and c. Only a few points, e.g. the two or three dots shown in Figure

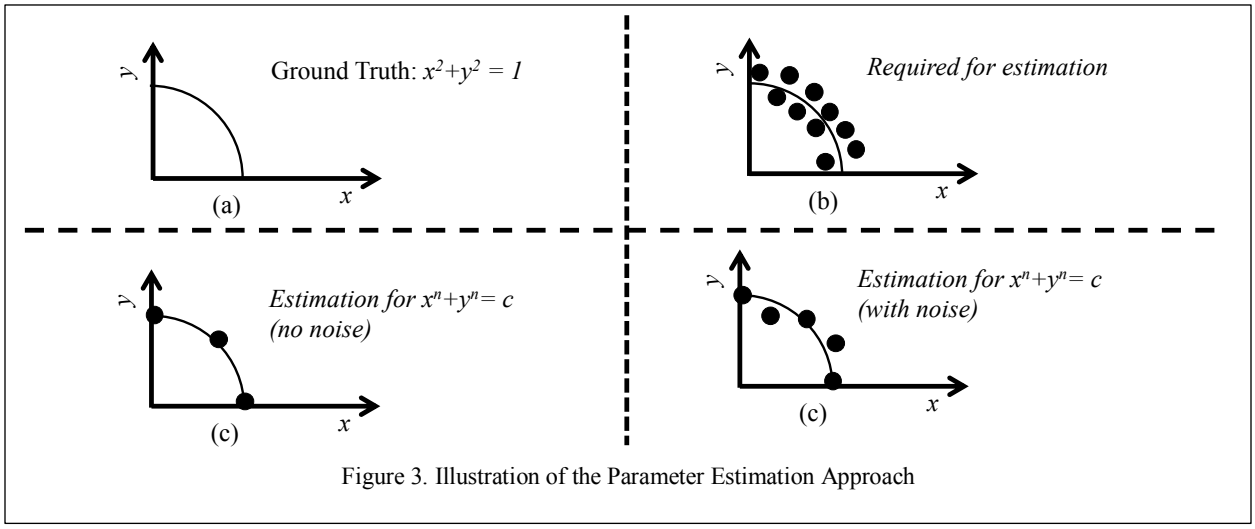

Figure 3. Illustration of the Parameter Estimation Approach

3(c) are enough to determine the values of *n* and *c* if the dots are always on the ground truth line. If the dots cannot always be assumed to be on the line, a few more dots as shown in Figure 3(d) can allow us to get the best estimate, but the required number would be much smaller than in Figure 1(a).

There are many examples where the grey-box learning approach has been applied. For an estimation of distributed systems performance, queueing models have been assumed to characterize their behavior and model parameters estimated using a learning approach [6] [7]. For a set of biomedical class of applications, it has been shown that an artificial neural network based method and an approach based on parameter estimation using an ARMA model for time-series are equivalent [8]. The performance of virtualized infrastructures, which are commonly used in modern cloud computing and data center managements, can also be modeled effectively using learning approaches with a small number of parameters [9]. In the domain of sensor fusion, algorithms based on Kalman Filter can be viewed as an example of the parametric approach for machine learning. In the field of control theory, a well-studied problem is that of system identification, where the challenge is to learn the impulse function of a system based on its input. Parametric inferences along with different variations of linear regression schemes to estimate the best function for system identification has been surveyed in [10].

Statistical models that assume some type of structure leading to a mapping from the input variables to the output function can be postulated based on the domain knowledge, with the Bayesian networks being a popular representation. Given such statistical representations, additional assumptions can be made about how different layers in a model interact. Assuming that nodes are connected so that the output of the next layer is determined by means of a probabilistic combination of previous layer's, a Noisy-OR gate can reduce the amount of data that is required to learn the parameters of a statistical model [11].

The one limitation associated with the parametric approach is that it requires a human to make a guess about the type of model that defines the system behavior, and that guess may be wrong. If the wrong model is used, the parameters may be too noisy, or the fit may be poor. The black box machine learning approaches, while requiring more training data, have the advantage that they can learn the right model based on the data available.

4. TRANSFER LEARNING

Transfer Learning provides a way to create new models by modifying existing models from other application areas. The basic idea in transfer learning is that the reuse of models allows us to use only a small amount of data to adapt and extend the models in new situations. The concept of transfer learning was introduced in [12], and it has been used most widely in

problems related to classification [13]. However, we can interpret it in other machine learning contexts as well, including that of function estimation.

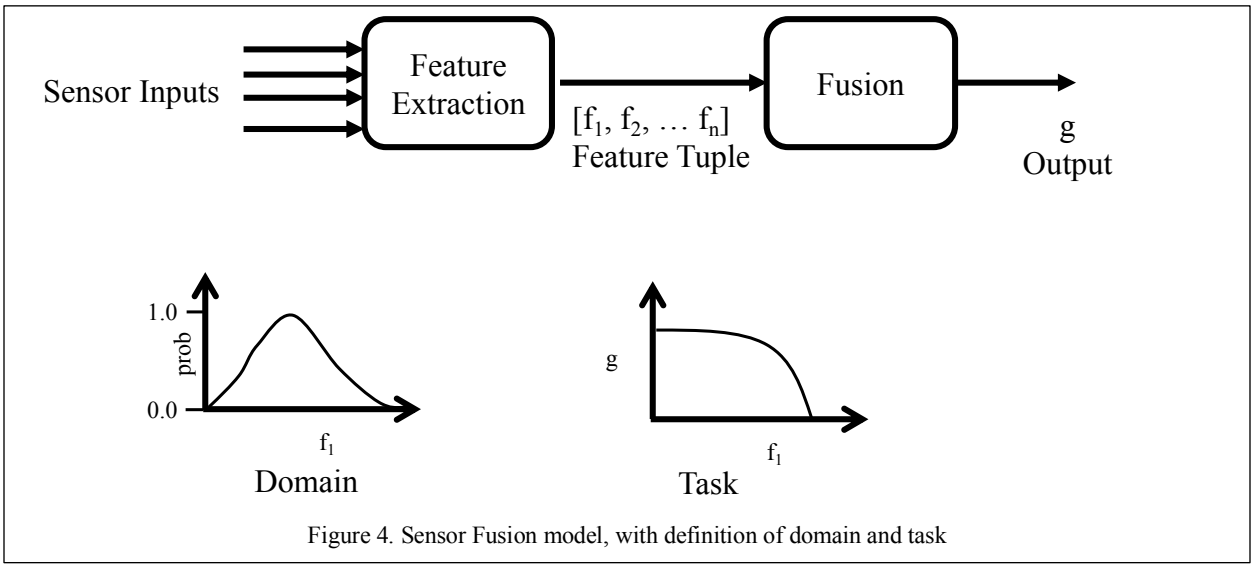

Figure 4. Sensor Fusion model, with definition of domain and task

In the context of sensor fusion model shown in Figure 1, the sensor fusion process can be viewed as consisting of taking some set of input signals, converting them to a set of feature vectors, and then trying to determine an output value based on the specific value of the features. The training data can be viewed as a set of features along with the right output for them. This view of sensor fusion is shown at the top of Figure 4. As an example, the sensor input may be time-varying values, e.g. acoustic signals, or vibration signals, and the features extracted may be the Fast Fourier Transform (FFT) coefficients of the signals, which can then be mapped to decide a binary output of normal state of operations, or abnormal state of operations.

Transfer learning views any AI model to be applicable to a "domain" and a "task". A domain consists of a set of features and a probability distribution among different combinations of features. A typical domain distribution with a single feature could be as shown in the bottom left side of Figure 4. A task is a relationship between the different features that defines the domain and the function being produced by the AI model. As an example, a domain could be acoustics for engine rooms, which has a single feature, the FFT coefficient of the sound produced by the engine. For acoustics in engine rooms, the different features have a specific joint distribution. A task may be to determine whether an engine is on or off by measuring the sound. In this case, *g* is a binary variable which is true if some of the FFT coefficients in the sounds is present. Another task may be to determine whether the engine is behaving normally or is malfunctioning. In this case, *g* is a different binary variable which is true or false depending on which different FFT coefficients are present. This task may result in a different curve indicating the relationship between *g* and the features (FFT coefficients). The probability of distribution of the FFT coefficients in automotive engine sounds will be different than that of engine room acoustics. Thus, the acoustics for automobile and acoustics for engine rooms will be two different domains.

The transfer learning situation is illustrated in Figure 5. We have a domain and a task for which sufficient training data is available. As a result, a good model can be trained. There is a new domain and a task for which there is limited amount of training data. The goal of transfer learning is to use the insights from the first domain/task/model to help in making a better model for the second domain/task/model. The reduction in training data requirements depends on knowledge which can be assumed to hold between the distributions in domains and tasks. There are many different ways in which the insights can be transferred. These include (a) use the training data available in the first domain/task to create new training data for the second domain/task and then learn a model based on this training data (b) use the insights about features from the first domain/task to improve the model for the second insight/task and (c) adapt the model learnt in the first domain/task as a base model which can be adapted/enhanced for the second domain/task.

Let us consider the task of expanding the available training data for the learning of the second model. If the two domains have the same set of features and differ only in how the probability distribution is different, then a few training data points in the new domain can be used to estimate the probability distribution of features in new domain, and the training data available in the first domain used to expand upon the training data available in the new domain. This approach has been

used in using different strategies for selecting the subset of training data from the first domain that can be transferred over. One approach [14] is to use the training data set in the new domain as testing data for the new AI model, and see whether

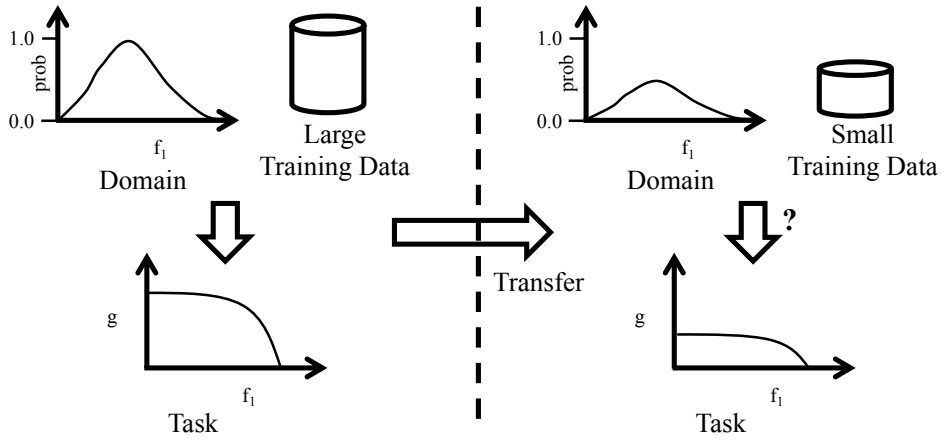

Figure 5. Transfer Learning Problem

correct or incorrect results are produced. Training data from the old domain that is close to the training data in the new domain and produces correct results is selected with a higher probability, while training data that is close to points that produce wrong results are selected with a lower probability. Another approach [15] is to estimate the distribution of features in the new domain with the smaller amount of training data, and to select training data from the old model according to the new distribution.

In many AI models, the mapping from sensor inputs to the set of features is also learnt as part of the model building process. The advantage of this approach is that it reduces the effort required to manually define features for a domain/task. However, this also requires the presence of large training data set so that the features could be extracted and learnt. If we can assume that the same features are used for another task, the definition of the features can be transferred to the new task. Thus, the smaller training data set can be transformed to a training data set which consists of the features for training. By such change in features, the smaller data set can be used to learn the remaining function. An optimization algorithm to determine which features correspond in different domains given the two sets of training data is given in [16]. Another approach to transfer feature information is to implement clustering algorithms on both training data sets, and to find correspondence between the clusters [17].

The model which is learnt from the first domain/task can also be used as the basic model on which the new model in the new domain/task can be based upon. One can assume that the basic nature of the relationship learnt between different features would remain the same, and just some parameters would be changed. These parameters can then be estimated using lesser amounts of data in the second domain/task. This can be viewed as using the learning in the first domain/task as a way to get the parametric representation discussed in Section 3. As an example, assuming that the relationship between different features is a Gaussian distribution in both domains, parameters can be estimated from one domain and some of them reused/mapped to a new domain [19][20].

Another part of transfer learning attempts to define relationships or mapping between the features in the first domain, and the features in the second domain. As an example, one domain may consist of images taken from one viewpoint, e,g. directly in front of an object, while the other domain takes images from a different viewpoint, e.g. from the side. A mapping between images that are taken from one camera viewpoint and those taken from another viewpoint can be defined. If a lot of images, taken from the first viewpoint, are available and only a few available from the second viewpoint, the mapping can be used to translate features from the first domain to the second. This allows for better image object recognition with fewer images in the second domain [18]. Another way to define these mappings is to identify a Markov model that characterizes the transition probability among different features, find the logical relationships that hold between these transitions, and assume that the logical relationships hold true in the second domain [21]. This allows for transfer of relationships across domains and has been shown to be useful in transferring knowledge from molecular biology to web page content curation [21].

Transfer learning can prove to be effective provided there is an underlying reason why the knowledge learnt in one domain can be transferred into another. In many cases, the determination of appropriateness, i.e. whether the transfer is proper or improper, remains a black art rather than a something that can be done in a rigorous manner. In some cases where transfer can be shown to be valid using statistical analysis, the assumptions made on the two domains and tasks may be hard to find in practice.

5. FEATURE SPECIFICATION

The learning of an output function from the sensor inputs can be modeled as a two-stage process, instead of being a single stage process. In the first stage of the process, the sensor inputs are transformed into a (usually) a smaller set of features, and in the second stage of the process the features are used to define the output function. If we view the learning process as training a neural network, one can view the features as a representation of the nodes that are in the hidden or intermediate layers of the neural network.

The estimation of functions is essentially trying to find the right function in the feature space. Suppose, we are able to use traditional training with some amount of training data to estimate the function for a given combination of features. The idea in the feature specification way is that if the features can be identified properly, and a function has been learnt using a few samples in some region of the feature space, it can be extended to other regions of the feature space with only a few samples. The properties that have been learnt about the output functions and the features are used to extrapolate and extend the region to the unexplored space. This requires fewer points for the extension than for the original space.

Feature specification can be done manually by means of feature engineering or be transferred across domains as part of the transfer learning process, as described in Section 5. When done for the specific tasks of classification, feature learning has been used to define the area called “one-shot learning”. The idea of one-shot learning, which is used for classification tasks, is to use features to characterize the different classes instead of the original input data. When a new data point, whose features are very different than the features seen till now, is encountered, it can define a new class. Thus, new classes can be introduced with a few sample data points characterizing their features, instead of using a large number of training data points.

An example of this approach is found in one-shot learning which has been used in applications such as visual object recognition [22] [23] and gesture recognition [24]. The basic idea behind this approach is to learn the features that are most important for identifying the previous categories/classes in the system, with the characterization usually modeled as the probability distributions of features in a category of objects. When a single (or a limited number) of training data points belonging to a new class is introduced, the system compares their features against existing classes to rapidly determine that it is not one of the previous ones, and then defines a new class based on the features that make up the new training data points.

Feature learning can be very useful in practice and has been shown to work effectively in some domains. However, the ability to find the right features, and to map them to new classes effectively relies on the ability of the training data to reflect the right features correctly, or the ability of a human to identify the right features. The extension of this approach to learn new functions has not been explored well in the current set of reviewed literature.

6. DATA GENERATION APPROACHES

When there is not sufficient training data available, one way to address the problem is to create more training data. For a training algorithm that requires significant amount of training data, but we have access to only a few samples, expanding the few samples into a larger set of training data would allow the use of the required training algorithms.

In many domains, the collection of unlabeled data is easy to do, but labeling the data is the difficult, often manual, part. As an example, it is easy to collect lengthy footage of video or several images by having a camera take several pictures, but the labeling of objects in those pictures by hand is a tedious process. A web-crawler can easily collect several web-pages, but the labeling of the type of each page requires manual intervention. If we have a few labeled data and access to a large set of unlabeled data, one can use the labeled data as a way to convert more of the unlabeled data into labeled ones. The use of labeled data and unlabeled data together is usually given the name of semi-supervised learning [25]. One way to achive semi-supervised learning is to run a clustering algorithm on the unlabeled dataset, identify the clusters, find the labeled points closest to the clusters, and then use that to assign a label to them [26][27]. While not all of the unlabeled

data can be mapped to a label, a significant fraction can be, leading to an increase in the amount of training data. Another way to do such conversion is called co-training. It tries to learn combinations of features from the labeled data [28], where the presence of any of the features leads to a good classification, then use these to assign labels for unlabeled data containing any of those features. Several other approaches for semi-supervised learning have also been proposed [25].

Another approach for expanding upon the set of available data is the use of diffusion techniques to generate new artificial sample points [29]. The approach relies on the existence of a distance metric between two available data points in the training set. New points are introduced between the existing entries in the training set with the label for the data point assigned based on the distance between the two points in a probabilistic manner. Diffusion neural networks [30] employ this approach. They are neural networks trained on a data set containing the actual data along with several artificial data samples introduced by the diffusion of points into the real data set.

Transfer Learning can also be used to expand upon the set of available training data [14] [15] as described in Section 5.

Data generation allows one to increase the set of available data for training but can be critiqued as just repeating the patterns present in available small training data set. Such expanded data may lead to models that have been overfitted to the available data and may not reflect the ground truth accurately.

7. CONCLUSIONS

In this paper we have surveyed several approaches for creating AI models when only a small amount of training data is available. We have divided those methods into four different categories, those that rely on specification of a model and estimating a set of parameters, those that rely on transferring knowledge from another domain, those that rely on the use of higher level features, and those that expand the amount of training data that is available. We have also discussed the strength and weaknesses of the different approaches.

Despite the existence of many algorithms for training on small data sets, each approach has a set of limitations that are hard to overcome. It is reasonable to assert that learning on small data sets remains an unsolved problem that requires further research and investigation.

8. ACKNOWLEDGEMENTS

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copy-right notation hereon.

REFERENCES

[1] Raudys, S., & Jain, A, "Small sample size effects in statistical pattern recognition: Recommendations for practitioners," IEEE Transactions on pattern analysis and machine intelligence, 13(3), 252-264 (1991).

[2] Mustafa A., "Interpreting the VC Dimension", Caltech Machine Learning Video Library. http://work.caltech.edu/library/073.pdf.

[3] Cortes C., Jackel, L., Solla, S., Vapnik V., and Denker, J., “Learning curves: asymptotic values and rate of convergence,” Advances in Neural Information Processing Systems, 6, 327–334 (1994).

[4] Perlich, C., Provost, F., & Simonoff, J., “Tree induction vs. logistic regression: A learning-curve analysis,” Journal of Machine Learning Research, 4(Jun), 211-255 (2003).

[5] Caruana, R., & Niculescu-Mizil, A., “An empirical comparison of supervised learning algorithms,” Proc. of the 23rd ACM International conference on Machine learning, 161-168 (2006).

[6] Kraft, S., Pacheco-Sanchez, S., Casale, G., & Dawson, “Estimating service resource consumption from response time measurements,” Proc. of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools, 48- (2009).

[7] Liu, Z., Wynter, L., Xia, C. H., & Zhang, F., "Parameter inference of queueing models for it systems using end-to-end measurements," Performance Evaluation, 63(1), 36-60 (2006).
[8] Chon, K. H., & Cohen, R. J., "Linear and nonlinear ARMA model parameter estimation using an artificial neural network," IEEE Transactions on Biomedical Engineering, 44(3), 168-174 (2007).
[9] Spinner, S., Kounev, S., Zhu, X., Lu, L., Uysal, M., Holler, A., & Griffith, R., "Runtime vertical scaling of virtualized applications via online model estimation," Proc. IEEE Eighth International Conference on Self-Adaptive and Self-Organizing Systems, 157-166 (2014).
[10] Pillonetto, G., Dinuzzo, F., Chen, T., De Nicolao, G., & Ljung, L., "Kernel methods in system identification, machine learning and function estimation: A survey," Automatica, 50(3), 657-682 (2014).
[11] Oniśko, A., Druzdzel, M. J., & Wasyluk, H., "Learning Bayesian network parameters from small data sets: Application of Noisy-OR gates", International Journal of Approximate Reasoning, 27(2), 165-182 (2001).
[12] Pratt, L., "Discriminability-based transfer between neural networks", Proc. of Advances in Neural Information Processing Systems, 204-211 (1993).
[13] Pan, S. J., & Yang, Q., "A survey on transfer learning," IEEE Transactions on Knowledge and Data Engineering, 22(10), 1345–1359 (2010).
[14] Dai W., Yang, Q., Xue, G., & Yu,Y. "Boosting for transfer learning," Proc. of the 24th International Conference on Machine Learning, Corvalis, Oregon, USA, 193–200 (2007).
[15] Jiang, J. & Zhai, C., "Instance weighting for domain adaptation in NLP," Proc. of the 45th Annual Meeting of the Association of Computational Linguistics. Prague, Czech Republic, 264–271 (2007).
[16] Blitzer, J., McDonald, R., & Pereira, F., "Domain adaptation with structural correspondence learning," Proc. Conference on Empirical Methods in Natural Language, Sydney, Australia, 120–128 (2006).
[17] Dai, W., Yang, Q., Xue, G., & & Yu, Y., "Self-taught clustering," Proc. 25th ACM International Conference of Machine Learning, 200–207 (2008).
[18] Shao, L., Zhu F., & Li, X., "Transfer Learning for Visual Categorization: A Survey," IEEE Transactions on Neural Networks and Learning Systems, 25(5), 1019-1034 (2015).
[19] Lawrence, N., & Platt, J., "Learning to learn with the informative vector machine," Proc. ACM International Conference on Machine Learning, Banff, Alberta, Canada, July (2004).
[20] Schwaighofer, A., Tresp, V. & Yu, K., "Learning Gaussian process kernels via Hierarchical Bayes," Proc. Annual Conference on Neural Information Processing Systems, Cambridge, 1209–1216 (2005).
[21] Davis, J. & Domingos, P., "Deep transfer via second-order Markov logic," Proc. AAAI Workshop on Transfer Learning for Complex Tasks, Chicago, Illinois, USA, (2008).
[22] Lake, B., Salakhutdinov, R., Gross, J., & Tenenbaum, J., "One shot learning of simple visual concepts," Proc. Annual Meeting of the Cognitive Science Society, (2011).
[23] Li, F., Fergus, R., & Perona, P., "One-shot learning of object categories," IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(4), 594-611 (2006).
[24] Wu, D., Zhu, F., & Shao, L., "One shot learning gesture recognition from RGBD images," IEEE Computer Society Computer Vision and Pattern Recognition Workshop, 7-12 (2012).
[25] Zhu, X., & Goldberg, A., "Introduction to semi-supervised learning," Synthesis lectures on artificial intelligence and machine learning, 3(1), 1-130 (2009)
[26] Dara, R., Kremer, S., & Stacey, D., "Clustering unlabeled data with SOMs improves classification of labeled real-world data," Proc. World Congress on Computational Intelligence, (2002)
[27] Demiriz, A., Bennett, K., & Embrechts, M., "Semi-supervised clustering using genetic algorithms," Proc. Artificial Neural Networks in Engineering, (1999)
[28] Blum, A., & Mitchell, T., "Combining labeled and unlabeled data with co-training," Proc. Annual Conference on Computational learning theory, 92-100 (1998).
[29] Li, D., Wu, C., Tsai, T., & Lina, Y., "Using mega-trend-diffusion and artificial samples in small data set learning for early flexible manufacturing system scheduling knowledge," Computers & Operations Research, *34*(4), 966-982 (2007).
[30] Huang C., & Moraga C., "Diffusion Neural Network for learning from small samples", International Journal of Approximate Reasoning, 35, 137–161 (2004).