University of Exeter
Department of Computer Science

Training-ValueNet: A new approach for label cleaning on weakly-supervised datasets

Submitted by Luka Smyth to the University of Exeter
as a thesis for the degree of
Masters by Research in Computer Science
In September 2019

I certify that all material in this thesis which is not my own work has been identified and that no material has previously been submitted and approved for the award of a degree by this or any other University.

Signature: .....

Abstract

Manually labelling training data for machine learning has always been incredibly time-consuming and expensive. For those who seek to apply modern deep learning algorithms however, the cost of acquiring enough accurately labelled data is quickly becoming *the* single greatest obstacle impeding progress. Weakly-supervised learning offers a promising alternative by enabling practitioners to rapidly apply weak sources of supervision to large amounts of data. Unfortunately, the presence of label noise in these datasets remains a critical issue as it can severely impair the performance of a machine learning model. In this thesis, we investigate a new approach for performing label cleaning on weakly-supervised data *without* human supervision. We propose that the boundary between correctly labelled and mislabelled examples might best be described in terms of the impact that an individual training example has on performance. Specifically, we hypothesise that mislabelled training examples will reliably detriment the generalization performance of a classifier and can be identified as such. To this end, we present the Training-Value approximation network (Training-ValueNet) which learns to estimate the *training-value* of each example - an objective measure of its impact on performance. In a series of three key experiments, we demonstrate that by simply discarding examples with a negative training-value, Training-ValueNet can significantly reduce the proportion of label noise in weakly-supervised datasets and improve the final performance of an image classification model as a result. In a label noise detection task, our method achieves a substantial *39% lower detection error* than the current state-of-the-art outlier detection method for label cleaning. Furthermore, we demonstrate that when our method is used for label cleaning, weakly-supervised learning can achieve comparable performance with the fully-supervised paradigm. This highlights the potential for data-driven approaches like ours to eradicate the need for manual label cleaning all-together. We have released an easy-to-use, open-sourced implementation[1] of Training-ValueNet in the hope that researchers and practitioners can benefit from it.

[1] `https://github.com/lukasmyth96/Training-ValueNet`

Acknowledgements

First and foremost I must express my sincere gratitude to my supervisor Dr Nicolas Pugeault for all his guidance over the past two years. I shall forever be grateful for the freedom he afforded me to pursue my ideas and for his invaluable knowledge.

I would also like to thank Dr Dmitry Kangin for always being so generous with his time and advice, and Dr David Walker for his part in inspiring me to pursue this degree in the first place. I am grateful for my colleagues at nate who inspire me each day to continue learning and improving myself.

Last, by not least, I would like to thank my parents for always supporting me in whatever I do, and Rebecca, without whom this may not have been finished any time soon.

Contents

List of Figures

List of Tables

Summary of Notation

x	Single data point (example) for a machine learning task
x^{meta}	Meta-data for example x for example an image caption
X	Set (dataset) of data points
y	True label for a single data point x
$\hat{y}$	Weak label for a single data point x
$l(x, x^{meta})$	Labelling function
Y	Set of labels for a corresponding dataset X
T	A machine learning classification task
K	Number of classes in a classification task
X^W	Set of weakly labelled examples
X^T	Set of training examples
X^V	Set of validation examples
$f(x)$	Set of features for example x extracted by f
$h(x;\theta)$	Machine learning model with parameters θ
$L(X)$	Loss (error) of a machine learning model over set X
$V(x)$	True training-value of example x
$\bar{V}(x)$	Monte-Carlo estimation of training-value for example x
$\hat{V}^C$	Training-ValueNet for a single class $C \in 1, .., K$
$\hat{V}(x)$	Predicted training-value for example x

List of Publications

A paper [2] [1] based on the work in this thesis was published at the 2019 Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics. This paper was co-authored with my supervisor Dr Nicolas Pugeault and Dr Dmitry Kangin of the Computer Science department at the University of Exeter.

[2] http://empslocal.ex.ac.uk/people/staff/np331/publications/SmythEtAl2019.pdf

1 Introduction

1.1 The new bottleneck in Deep Learning

Data has always played a crucial role in scientific discovery and progress. If it wasn't for the painstaking astronomical observations of Tycho Brahe for example, Johannes Kepler would never have discovered the laws of planetary motion and Newton would not have had the foundations for his law of universal gravitation. In recent decades, humanity has experienced an explosion in the amount of data that is available to us. The computer and later the internet have enabled us to generate, store, and share data in a way that was previously impossible. A natural consequence of this newfound abundance of data has been an increasing interest in techniques that allow us to make sense or use of it.

Machine learning is one such field that has advanced tremendously in the era of big data. An abundance of data provides not only the motivation for pursuing machine learning, however, it is a prerequisite for machine learning to work well in the first place. The power of machine learning algorithms lies primarily in the fact that, given enough labelled examples, they can learn incredibly complex functions between two domains that we humans could never write ourselves. Classifying the contents of an image, for example, is a task that young children accomplish with ease and yet we would have little hope of expressing this ability as a function of pixel values.

Prior to 2010, machine learning practitioners still had a significant role to play in extracting a useful set of features from their data which a machine learning algorithm could feasibly learn from. This all changed at the turn of the decade however when major improvements in parallel computing in conjunction with the introduction of huge labelled datasets such as ImageNet [2] sparked a re-birth of a sub-field of machine learning called deep learning. At their foundation, deep neural-networks learn a series of non-linear transformations that are applied to the input data via a series of hierarchical layers. In this way, deep neural networks are able to build up an increasingly abstract and composite representation of the input space *without* the need for manual feature

engineering.

In recent years, deep learning models have achieved astonishing performance in several domains and are set to have a significant impact on our daily lives. The power of deep learning comes at a cost however. The same characteristics which enable deep neural networks to learn such complex functions also makes them susceptible to *overfit* to spurious patterns in the data they are trained on, resulting in a model that generalises poorly to unseen examples. By *far* the most effective solution to the problem of overfitting is to simply expand the size of the training set. The more examples included in the training set, the better it will reflect the diversity and complexity of the target domain.

Unfortunately, however, additional training data is generally only beneficial provided we can obtain accurate labels for each new example - a task that is typically carried out by humans. And herein lies the problem that motivates this work. The manual labelling of large datasets is incredibly time-consuming and expensive. As a concrete example, the construction of the ImageNet dataset [2] took three years and required the services of 49,000 remote workers to label the 3.2 million images. Whilst the cost of the project was never revealed, it is clear that large scale data labelling is a significant financial investment.

Given this, it should be clear that the majority of small companies, charities, and research groups simply do not possess the time nor finances required to label data at this scale. What we risk as a society is a situation in which the power to leverage this technology will be concentrated in the hands of a small number of large, private organizations that have the means to acquire and label huge amounts of data. It is therefore imperative that we, as a research community, should seek alternative means of acquiring accurate labels for training data *without* the need for extensive human supervision.

1.2 Weak-supervision and the need for Label Cleaning

Given all the talk in recent years of artificially intelligent systems replacing skilled workers, it is certainly ironic that the majority of machine learning algorithms in use today rely so heavily on tiresome human labour. When one considers the task of automating the process of data labelling however, it soon becomes apparent that this reliance on human supervision is not without good reason. After all, any system that is capable of accurately labelling our data is already able to performing the desired task. This chicken and egg dilemma can only be resolved if we temporarily lower our expectations on the accuracy of labels. If we afford ourselves this concession, then we come to realise that it is often feasible to assign reasonably accurate, *weak* labels using some simple heuristic. This process is commonly referred to as weakly-supervised learning. In this case, rather than a human labelling each example individually, we use our prior knowledge to define a suitable labelling function that assigns a weak label $\hat{y}_i$ to each example x_i in our dataset.

$$\hat{y}_i = l(x_i, meta(x_i)) \tag{1}$$

Consider the labelling function illustrated in Fig. 1.1 for example. Here we have images of animals and their captions that have been scraped from news articles about pets. We can define a labelling function that simply returns 'dog' if the caption contains the word dog and likewise for cats. Whilst this type of heuristic labelling function will not be suitable in all situations, there are many problems for which they do offer a very quick and intuitive way to label large amounts of data.

In certain circumstances, we may observe that the dramatic increase in the size of dataset that can be obtained using weak supervision outweighs the detrimental impact of unreliable labels and leads to an acceptable level of performance. Unfortunately, however, this is not true in the majority of cases. Instead, the presence of *label noise* typically detriments the final performance of a model to such an extent that people either feel compelled to clean their data manually or are dissuaded from pursuing weakly-supervised learning altogether.

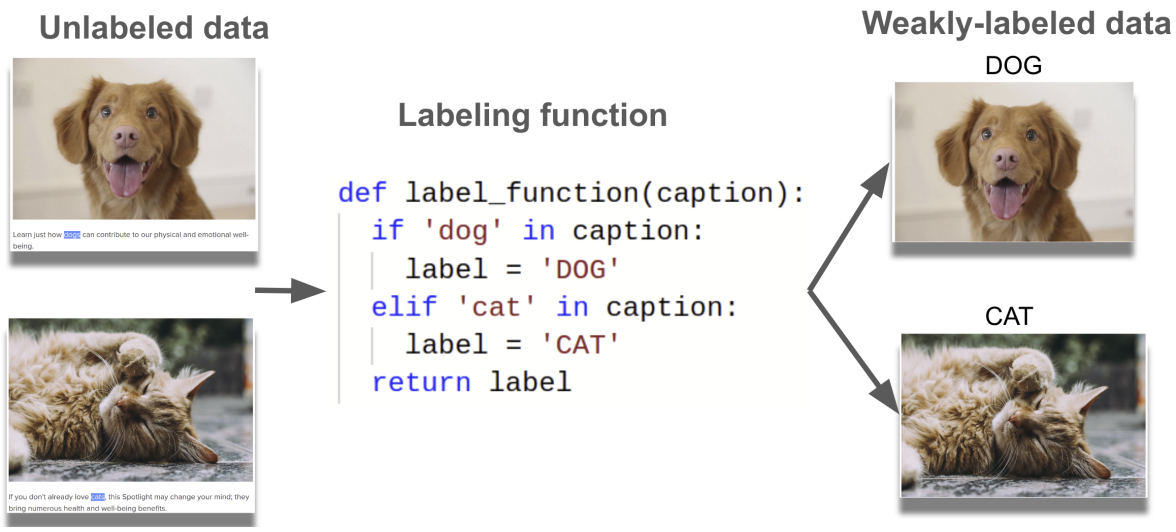

Figure 1.1: A simple example demonstrating how heuristic labelling functions can act as a useful source of weak-supervision.

This could be a very different story, however, if we could develop a method capable of reliably cleaning these weak labels *without* human supervision. Whilst it might be near impossible to automatically label examples with high accuracy the first time round, the task of automatically identifying mislabelled examples seems a more manageable task. And so this is to be the goal which we set out with in this project. Specifically, we seek to develop a general purpose method that, given some weakly-labelled training set X^W can identify and remove mislabelled examples without need for human supervision. We hold the following three criteria for such a method:

1. The method should be effective in the sense that it is capable of distinguishing mislabelled examples from correctly labelled ones with high precision and recall.

2. The method should require no manual labelling or verification of the weakly-labelled training set X^W.

3. The method should be general purpose in the sense that that it can be applied to any classification problem.

1.3 Outline of Existing Approaches to Label Cleaning

Now that we have established a clear motivation and objective for this work, the only question that remains is how to develop such a system for automated label cleaning that meets our criteria. Naturally, we are not the first to set out with such a goal and so before we present our proposal, it would seem prudent to provide the reader with a brief overview of the main approaches which others have taken. We reserve a thorough and detailed discussion of the literate for section 2 and instead focus here on the general characteristics of existing approaches, why we feel they are unsatisfactory and what we might learn from their limitations. Our job is made easier here by the fact that the overwhelming majority of existing work on this subject can be placed into one of two general categories of approach. These are semi-supervised label cleaning and outlier detection. Let us now briefly examine each approach in turn.

Semi-supervised label cleaning is the term we use to describe any method that, as the name suggests, allows for partial automation of the label cleaning process. The way such methods typically work is that a human will be required to verify the weak labels for some subset of the training examples. This can either happen all at once or periodically during the cleaning process. The verification labels supplied by the human are then used to train some secondary machine learning model that learns to identify and remove further mislabelled examples in the dataset. The utility of such methods lies in their ability to amplify the effect of any human effort made. They allow a user to trade off some (hopefully small) amount of label accuracy in exchange for no longer having to manually clean all training examples.

In general, semi-supervised approaches have proven to be rather effective, with popular large scale datasets such as LSUN [3] being created in this way. The downside of such approaches, however, is that they fall short of a fully-automated solution. If we needed to create a very large scale dataset, labelling even a small proportion of training examples would remain a formidable task. Furthermore, the effectiveness of semi-supervised methods is inevitably tied to the number of examples that the stakeholder is willing and able to label. This creates a clear to continue investing time and energy into man-

ual label cleaning. This is the antithesis of what we have set out to achieve in this work.

The second type of approach that is commonly proposed for label cleaning is outlier detection. In contrast to the semi-supervised approach, outlier detection offers the clear advantage of requiring no human supervision. Whilst individual methods differ substantially in their definition of what constitutes an outlier, they all work on the same fundamental assumption that mislabelled examples and outliers are equivalent. Under this assumption, any number of outlier detection methods have been applied to identify and remove outliers in the hope of reducing label noise.

With its rich history and seemingly rational premise, it is understandable why many have pursued outlier detection as a potential solution for label noise. Unfortunately, however, recent studies [4] [5] have shown that even state-of-the-art outlier detection methods typically fail to achieve good performance on 'real-world' [3] label noise. In this work, we would like to offer a possible explanation as to why this poor performance has been observed. Whilst we defer a complete analysis to section 2, the essence of our argument is that outlier detection will fail because the fundamental assumption that mislabelled examples and outliers are equivalent is inherently flawed.

The reason for this is that the presence of a mislabelled example in our weakly labelled dataset suggests that an error has been made by our labelling function. Such an error is deterministic not random and so, assuming there exists similar examples in the dataset, it is highly likely that they too will be mislabelled. As a result of this, we find that the majority of mislabelled examples in real weakly-supervised datasets tend to exist in dense clusters. As a result, outlier detection methods will fail to detect a significant proportion of mislabelled examples for the simple reason that they are not outliers.

Furthermore, we also contest the converse assumption that genuine outliers are generally mislabelled. In reality we have found that many outliers tend to be correctly

[3]By 'real world' label noise we mean mislabeled examples that are naturally occurring in a dataset due to errors made by one or more weak labelling functions. This is in contrast to the artificial, randomly generated label noise that many researchers have used as a proxy in their experiments.

labelled corner cases that are in some way atypical but no less valid. By mistakenly removing these examples, outlier detection methods will reduce the diversity in the examples that remain which can easily lead to a model that generalises poorly.

There exists a clear tradeoff between the effectiveness and scalability offered by existing approaches for label cleaning as we have illustrate in Fig. 1.2. On one end of the spectrum, there are semi-supervised methods which offer reasonable effectiveness but are limited in terms of scalability due to their continued reliance on human supervision. Alternatively, outlier detection methods are highly scalable but mostly ineffective. It is clear to us that no method which follows one of these two paths will simultaneously achieve the level of scalability and effectiveness that we desire. The limitations of each approach are simply too ingrained in the way they operate. If we are to achieve what we have set out to achieve, we must develop an entirely new approach to label cleaning, one that is free of the limitations of these two existing approaches.

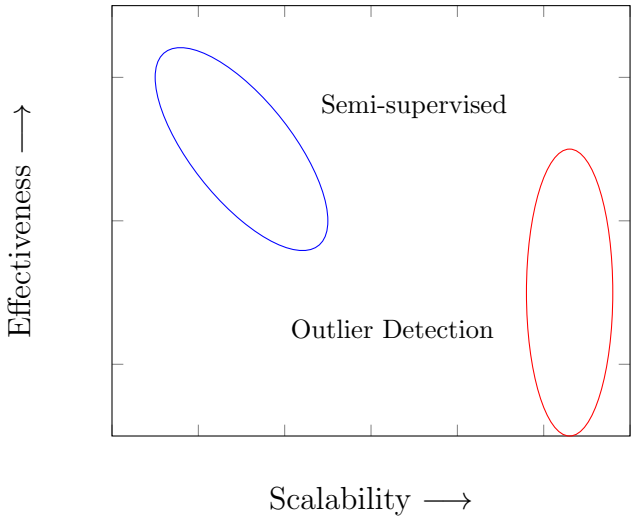

Figure 1.2: Illustration of the trade-off between scalability and effectiveness among existing approaches to label cleaning.

1.4 A new approach to label cleaning

If we are to successfully reconcile the existing trade-off between effectiveness and scalability, it is helpful to first observe that outlier detection and semi-supervised methods both fail for the same fundamental reason. They fail because of their continued over-reliance on human knowledge. It is clear how this is the case for semi-supervised label cleaning which leverages human knowledge explicitly in the form of verification labels. Whilst less obvious, the statement is equally true for outlier detection. As we have established, outlier detection performs poorly due to the misguided assumptions it makes about the distribution of label noise. Such assumptions, however, are of course merely a distillation of current human understanding. A useful starting point in developing a new approach to label cleaning, therefore, is to acknowledge the danger of relying on human knowledge which can be so costly and misguided.

It is clear that any label cleaning system which does not leverage human knowledge must instead *learn for itself* which examples are mislabelled. Furthermore, if a system is to learn in this way, it must surely do so by learning to approximate some intermediary measure which is so highly correlated with the probability that an example is mislabelled that it can be used to reliably detect label noise. Finally, in the absence of human intervention, there must be some principled manner in which we can determine a threshold on this measure which defines the boundary point between correctly labelled and mislabelled examples.

What we have just described is, of course, how all existing methods for label cleaning work internally. Outlier detection, for example, assumes that some measure of how outlying a particular example is will be sufficiently correlated with the probability of it being mislabelled that if they can only determine a suitable threshold on this measure, they can identify and remove mislabelled examples. An automated label cleaning method will therefore only succeed if the measure it uses *is* sufficiently well correlated with the probability of an example being mislabelled *and* if there is a principled way to determine a suitable threshold on that measure.

With this understanding in place, we are now ready to describe the theoretical framework upon which our new approach is built. Once again, we will defer a thorough description of the finer details of our method for section 3 and instead focus here on the essence of what we are proposing. Our theory comprises of one observation and one assumption.

Our primary *observation* is that for any classification task T for which we have a training set X^T and an evaluation set X^V, every training example $x_i \in X^T$ with label y_i possess some unique *training-value* which we denote $V(x_i)$. Whilst we leave a formal definition of training-value for section 3.3, it will suffice for now to say that the training-value of an example is a measure of the expected overall impact that training on that example will have on the generalization performance of the classifier. In relation to what we have discussed in the preceding paragraphs, it is the training-value of an example that we propose is sufficiently well correlated with the probability of it being mislabelled that it can be used as a basis for label cleaning.

The primary *assumption* of our theory is that whilst there are several factors that determine the magnitude of an example's training-value, the correctness of its class label is what primarily determines the *sign* of that value. Specifically, we assume that correctly labelled examples always offer at least some positive value to the training process whereas mislabelled examples will reliably detriment performance and therefore possess a negative training-value. In relation to what we discussed in the preceding paragraphs, this is equivalent to saying that the principled threshold on the measure of training-value is simply zero.

We have now established that training-value will be the intermediary measure we use to identify label noise and that we will use a threshold of zero to perform label noise detection. If we can learn to accurately estimate the value of each example in our training set then we will be able to effectively reduce label noise by simply discarding any examples that we have a negative value. Fig. 1.2 illustrates the proposed approach.

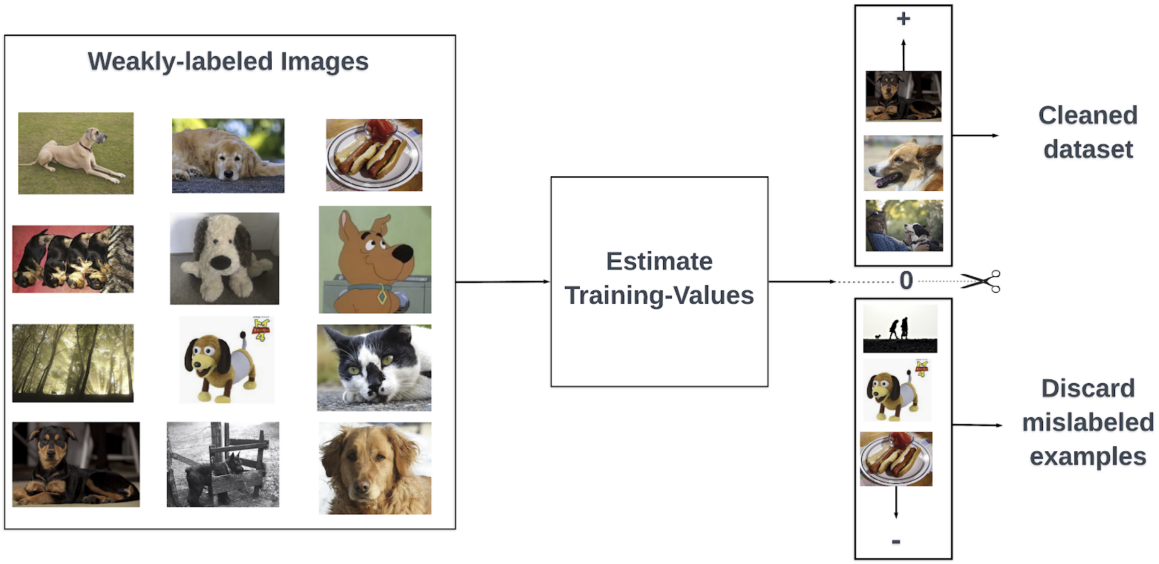

Figure 1.3: High level illustration of our approach to label cleaning.

1.5 Outline of Subsequent Chapters

The remainder of this thesis will be organised as follows. In section 2, we will conduct a thorough analysis of the existing literature surrounding weakly-supervised learning (section 2.1), the impact of label noise (section 2.2) and a number of different approaches for dealing with label noise (sections 2.3-2.4) which includes but is not limited to the two label cleaning approaches we have discussed so far. We conclude section 2 by clearly establishing which existing methods comprise the state-of-the-art in each category of approach. These methods will form a key point of comparison four our proposed method.

In section 3 we proceed to provide a detailed description of our proposed method. We provide a formal definition of Training-Value in section 3.3 and describe how we obtain an unbiased Monte-Carlo estimation of an example's training-value directly from episodes of experience. In section 3.5 we present the Training-Value approximation network (Training-ValueNet), a regression network that learns to predict the training-value of an example directly from its feature vector. The benefit of using the Training-ValueNet is that it allows us to efficiently estimate the training-values of very large

numbers of training examples.

We conduct a thorough evaluation of our proposed method in section 4 through a series of three key experiments:

1. Experiment 1 is a label noise detection task in which assess how accurate our method is in detecting which examples in a weakly-supervised dataset are mislabelled. We compare our results in this experiment with the state-of-the-art methods from semi-supervised label cleaning and outlier detection.

2. In experiment 2, we go on to evaluate how successful our label cleaning has been in terms of the resulting improvement in classification performance. Again we compare our results with the state-of-the-art semi-supervised method as well as weakly-supervised and fully-supervised baselines.

3. Finally, in experiment 3 we conduct a full comparison of weakly-supervised learning using our method with the fully-supervised paradigm. We are interested here not only in the final performance that each approach yields but also in comparing the entire pipeline of collecting, labelling, cleaning, and training.

We present our conclusions on this work in section 5, along with a summary of the contributions it has made to the label cleaning literature. We conclude the thesis in section 6 where we present several ideas for future work. In some cases, we have conducted some preliminary investigations to demonstrate why we think the direction is interesting.

2 Background and Related Work

2.1 Weakly-Supervised Learning

Weakly-supervised learning is a machine learning problem that belongs to the broader supervised learning paradigm. The goal of supervised learning is to learn a mapping between input and output domains by training a machine learning model on a set of labelled examples. Weakly-supervised learning differs from the traditional fully-supervised paradigm in that rather than training examples being accurately labelled by human subject matter experts (SME), we have:

- A dataset $X = x_1, ..., x_N$ with unknown true labels $y_1, ..., y_N$.
- We may have some meta-data [4] x^{meta} for each data point.
- One or more sources of weak supervision (labelling functions) $l_i(x, x^{meta}), i = 1{:}M$ such that each one has:
  - A *coverage set* $C_i \subset X$, which is the subset of the data points for which it is defined.
  - Some unknown accuracy over its coverage set which we assume to be < 1.

We refer to each individual source of weak supervision as a *labelling function*. There are several kinds of labelling function we might want to use, including but not limited to:

- **Heuristic rules** defined by a subject matter expert (SME). These allow an expert to distil their knowledge into a set of rules that mirror the intuition they would use when labelling the data themselves (see Fig. 1.1 for an example).

[4] We use the term meta-data here to refer to any additional information about a data-point which we may wish to utilise when assigning weak labels but will not be available when the model is used for inference. For example, the caption of an image offers information about its contents but is not something our final image classifier will be able to use.

- **Inaccurate labels from non-experts** may also be considered as a form of weak-supervision provided there is no reason to expect that the people doing the labelling will be generally correct.
- **Existing resources** such as knowledge-bases or predictive models can generate labels that are helpful but not perfectly accurate for this given task.

The power of these labelling functions is that they allow us to avoid requiring SMEs to label each individual example, something which is expensive and time-consuming. Instead, we can distil their knowledge and other resources we may have at our disposal into a small set of one or more labelling functions which provide a higher-level source of supervision. These functions can then be used to quickly obtain weak labels for huge amounts of data.

One of the major challenges facing weakly-supervised learning, however, is that as soon as we have more than one labelling function at our disposal it can become difficult to know how to weight the predictions of each when determining the final label for an example. One approach would be to simply average the predictions of all of our labelling functions. Unfortunately this is not ideal because the accuracy of labelling functions may vary significantly. Furthermore, it is quite possible that two or more of our labelling functions will be highly correlated in their predictions. In this case, giving equal weighting to each labelling function would lead to a double-counting problem.

In 2017, researchers at Stanford offered a solution to this problem. They proposed a new paradigm for the programmatic creation of training data which they termed *data programming* [6]. Data programming allows users to express several weak-supervision strategies simply in the form of a set of labelling functions. The primary contribution of this work, however, was that by modelling the labelling of training examples as a generative model, the authors were able to automatically infer the accuracy of each labelling function and analyse the correlations between them *without* access to any ground truth labels. This model is then able to learn a weighted combination of the labelling functions that leads to more accurate final labels.

More recently, the same research group released Snorkel [7], an open-source and easy-to-use implementation of their data programming paradigm. For the first time, Snorkel has made it quick and easy to apply multiple sources of weak-supervision to unlabelled data without having to worry about the varying reliability of, and interactions between labelling functions. Snorkel has been very well received by practitioners. Several experimental results have demonstrated that, provided a sufficiently large pool of unlabelled data is available, Snorkel is able to generate a weakly-labelled dataset in a matter of minutes that yields performance comparable with fully-supervised learning.

A follow-up study [8] conducted by researchers at Stanford and Google analysed this trade-off between weak supervision and hand-labelled data in more detail. In two text classification tasks, they trained a model on increasingly large, hand-labelled training sets until they reached the same level of performance that was achieved using weak supervision and Snorkel. On the first task, they found that they required approximately 12K hand-labelled examples to match the accuracy of their weakly-supervised classifier that was trained on 6.5M examples labelled using Snorkel. In the second task, they required 80K hand-labelled examples to match the performance achieved using 684K weakly-supervised examples.

The results of this study can be interpreted in two ways. On the one hand, hand-labelling even 12K examples is no small feat and it is therefore clear that Snorkel holds real potential to save its users a significant amount of time and money. On the other hand, to match the performance of fully-supervised learning Snorkel requires a significantly larger number of examples. In the first of these two tasks it took 540 times more data to match the performance obtained on a fully-supervised dataset. To borrow a term from reinforcement learning, we can say that the *sample efficiency* of weakly-supervised learning remains incredibly poo. Even when we start with a suitable set of labelling functions and use Snorkel to combine them, the presence of label noise in the resulting weakly-supervised dataset remains a significant issue. As such, despite the benefits that Snorkel and the data programming paradigm in general offer, we remain motivated to pursue our goal of automated label cleaning. Indeed, a general-purpose method for label cleaning might be an ideal additional to a tool such as Snorkel.

2.2 The impact of Label Noise

In the context of a classification task, label noise describes the phenomena whereby some proportion of the available dataset is mislabelled. In this instance, we might also refer to the labels themselves as being *noisy*. Whilst a small amount of label noise is likely to be present in even the most carefully labelled datasets, it is widely understood that this is largely unavoidable and generally unproblematic. Issues inevitably begin to arise, however, when the proportion of label noise in a dataset becomes non-negligible.

The effects of label noise on traditional machine learning algorithms have been well studied with the clear consensus being that classification performance will be negatively affected by the presence of label noise (see [9] for a comprehensive survey). Furthermore, it has been observed that label noise can also lead to an unwanted increase in model complexity making linear models more difficult to interpret. For example, Quinlan [10] warns that the size of decision trees may increase when there is label noise in the dataset, making them unnecessarily complicated. A similar phenomena was also shown for SVMs [11, 12].

Whilst there may be a clear consensus on the impacts of label noise on *traditional* machine learning models, the same can hardly be said of modern *deep* learning models. In a recent study, Rolnick et. al. [13] make the bold claim that deep convolutional neural networks (CNN) for image classification are robust to an arbitrary proportion of label noise provided that a minimum number of correctly labelled examples is maintained. They demonstrated this behaviour to differing extents on Cifar-10 [14], ImageNet [2] and finally Mnist [15] where remarkably, the CNN managed to maintain 90% performance when just one in every one-hundred examples were correctly labelled. Further support for the surprising robustness of CNNs to label noise comes from Flatow and Penner [16]. They showed that randomly permuting up to 70% of image labels on Cifar-10 caused less than a 50% decrease in performance.

Curiously, however, the remarkable robustness observed in these studies comes in stark contrast to what we see when deep learning models are trained on *real world* label noise

that is present naturally in a weakly-supervised dataset. The WebVision dataset [17], for example, was introduced as a noisy equivalent of ImageNet, sharing the same 1K classes that were used for the ILSVRC 2012 challenge. It was reported however that an image classifier trained on the WebVision training set achieved a substantial 9.4% lower top-1 accuracy on the ILSVRC 2012 test set than the same model trained on the fully-supervised ImageNet training set [17]. This is despite WebVision containing roughly twice as many training examples. Likewise, on the clothing 1M dataset [18], Patrini et. al.observed a 6.2% lower accuracy when they trained a classifier on 1M noisy images than with just 50K cleanly labelled examples [19].

The blatant disconnect between these two sets of results can be understood, however, if we consider the recent findings of Drory et. al. [5]. They demonstrated that robustness to label noise is heavily dependent upon the distribution of mislabelled examples in a dataset. When mislabelled examples were distributed uniformly among the correctly labelled examples for a particular class they observed almost no detriment to model performance. If, however, the same number of mislabelled examples were clustered into dense regions then the model's performance became substantially worse. The reason this result can help us make sense of the conflicting sets of findings is that it demonstrates that the process by which label noise is *generated* (and therefore distributed) will have a significant impact on how detrimental that label noise is likely to be. In the first two studies we discussed by Rolnick et. al. [13], and Flatow and Penner [16], label noise was artificially generated by randomly reassigning or permuting labels for some proportion of a cleanly labelled training set. Label noise that is generated in this fashion will follow a uniform distribution and therefore, as Drory et. al. [5] demonstrated, is unlikely to detriment performance.

This distinction between randomly-generated and real-world label noise is a crucial one that seems to have been largely overlooked by many researchers. Whilst there is nothing inherently wrong with studying the impact of stochastic label noise, it *is* very wrong to assume that these findings will generalise to the case of real-world label noise. The reason for this (as we discussed previously in section 1.3) is that label noise in a real weakly-supervised dataset exists as a result of errors made by the label functions

that were used to assign weak labels. These labelling functions are deterministic and therefore the errors they make are systematic, not stochastic. For each training example that gets labelled incorrectly by these functions, it is likely that there will be similar examples that are also mislabelled for the same reason. It is therefore the rule, not the exception, for label noise in weakly-supervised datasets to exist within dense clusters. As Drory et. al. [5] demonstrated, dense clusters of mislabelled examples can significantly detriment a model's performance which helps to explain the poor performance that has typically been observed when training on real weakly-supervised datasets.

We have made full and careful consideration of this distinction when designing the experiments to evaluate our proposed method. Whilst artificially generated label noise can be convenient to work with, we purposely restrict ourselves in this work to using *real* weakly-supervised datasets that contain naturally occurring label noise.

2.3 Approaches to Label Cleaning

Several approaches have been proposed for mitigating the negative impact of label noise. These approaches can first be divided into two categories based on their objective. The first type is label cleaning which we shall discuss in detail in this chapter. The objective of label cleaning methods is to identify and then remove or relabel any mislabelled examples in a dataset. Label cleaning can be partially or fully automated (as is the focus of our work) but can also be carried out manually by humans. Much of what we refer to as manual data 'labelling' is in actual fact label cleaning where a human is asked to confirm or correct a pre-existing weak label. In this work we are interested in methods that allow for partial or complete automation of the label cleaning process. As we discussed in the introduction, existing label cleaning methods can generally be placed into two categories. These are; semi-supervised label cleaning and outlier detection. Let us now consider each of them in detail.

2.3.1 Semi-supervised Label Cleaning

Semi-supervised label cleaning is the term we shall use to describe any method that allows for partial automation of the label cleaning process. The way such methods typically work is that a human will be required to either verify *or* correct the weak labels for some subset of the training examples. (The distinction here is that methods requiring *verification* of weak labels want to know whether each example is correctly labelled or mislabelled whereas other methods simply require some correctly labelled subset.) This manual labelling can either happen all at once or periodically throughout the label cleaning process. These labels are then used to train some secondary model that learns to identify and remove further mislabelled examples from the dataset. In doing so, the hope is that we can obtain an accurately labelled dataset whilst only having to label a small number of examples.

Semi-supervised approaches have proven their utility with several large scale datasets having being created using them. One example of this is the method proposed by Yu et. al. [3]. In this method, a small randomly selected subset of the training data is manually cleaned and then used to train an image classifier. The trained classifier is then used to predict the probability that each of the remaining images in the training set truly belongs to its recorded class. Images are then divided into three distinct groups based upon these predictions. Images with a high probability are immediately added to the final training set whilst those with a low score are assumed mislabelled and removed. Images that receive a predicted probability in-between the upper and lower thresholds are treated as an unknown and form a new subset for human labelling. This process is repeated until all images have been categorised as correctly or incorrectly labelled. This method was used to create the large scale LSUN dataset with 10M and 59M labelled images belonging to 10 scene and 20 object categories respectively [3]. The authors were able to achieve an estimated label accuracy of 90% whilst requiring only 2.5% images to be manually labelled. Despite such a major improvement in labelling efficiency, it is important to note that this still equates to 1.7M manually labelled images.

A major limitation of the majority of semi-supervised methods is that they require

images to be labelled in every class. As we look to scale up the number of objects we can classify, labelling even a small proportion of images from each class can quickly become unfeasible. A recently proposed method that addresses this issue is CleanNet [4]. Here an autoencoder is first trained on a representative sample of images from each class to create a class *prototype embedding*. A second encoder is then used to embed each weakly-labelled candidate image. This encoder is trained on a small seed of manually provided verification labels using a matching constraint such that an images embedding will be similar to its class prototype embedding only if it is correctly labelled.

The cosine similarity between the image and its class level embedding provides a relevance score for each image in the training set. Because the same prototype and query encoder is learnt for all images, information is able to be transferred to classes that contain no verification labels. CleanNet is demonstrated to achieve excellent performance on several challenging benchmarks including Food 101N [4], Clothing 1M [18] and Webvision [17]. Whilst the baseline version of CleanNet still requires 250 manually verified images per class, the authors were able to maintain reasonable label cleaning performance with up to 50% of classes receiving no manual labels. CleanNet represents a clear state of the art among the semi-supervised approaches to label cleaning and as such will act as our point of comparison for this group of methods.

2.3.2 Outlier Detection

Outlier detection methods take a very different approach to label cleaning. As we discussed in the introduction, outlier detection methods all share the same fundamental assumption which is that mislabelled examples are equivalent to outliers and can be identified as such. The main difference between the various outlier detection methods is precisely how they define an outlier. This will have a substantial impact on how effective the method is for performing label cleaning.

One of the simplest and most widely studied forms of outlier detection is *nearest-neighbour* methods which measure the relative proximity of each example to its neigh-

bours in feature space. Examples that are found to be further away on average from their neighbours are identified as outliers. Commonly used measures include isolation forest [20] and the local outlier factor [21], the latter of which was recently implemented in the context label cleaning for image classification by Xia et. al. [22].

An alternative is so-called *probability density* methods that treat outlier detection as a probabilistic modelling process. They attempt to estimate the probability density function of the training data to identify outliers as those examples which exist in regions of low probability. Non-parametric estimators are likely more suitable for this task as the generating distribution of weakly-supervised label noise is unknown and potentially very complex. Among this class of method, Robust-KDE [23] has been proposed as a variant of the classical Parzen-Rosenblatt kernel density estimator (KDE) [24, 25] that is more robust to label contamination. Specifically, RKDE learns separate density estimations for the nominal (inlying) and contaminating (outlying) examples.

The most promising class of outlier detection methods for label cleaning, however, are those which utilise the reconstruction error of an autoencoder to separate inliers from outliers. DRAE [22] is one such method that uses an autoencoder to compress an image into a lower-dimensional representation before mapping this back to a reconstructed copy of itself. The low dimensional intermediate representation acts as an informational bottleneck which effectively limits the autoencoder from accurately reconstructing mislabelled images that do not reflect the statistical regularities of the dataset. As a result of this, a large reconstruction error acts a strong indication that an image is outlying and thus mislabelled. Empirically, DRAE achieves consistently better performance than the prior state-of-the-art UOCL [26] and other outlier removal methods for label cleaning on weakly-supervised image data and therefore we consider it to be the state-of-the-art in outlier detection.

Whilst DRAE and other outlier detection tasks have demonstrated promising performance in a controlled setting, recent studies have shown just how poorly they can perform when tested on real-world label noise. On the weakly-supervised clothing 1M dataset, for example, it has been shown that DRAE performs no better than a naive baseline that simply classifies all examples as correctly labelled [4]. We would suggest that this disconnect between the results obtained on randomly generated vs real-world label noise can be explained by the different distributions of mislabelled examples in each case. In the case of randomly generated label noise, it is a perfectly reasonable assumption that mislabelled examples will be outliers. This is because each example is mislabelled with some probability that is independent of every other example. As a result, there is no reason to suspect that a mislabelled example will be particularly similar either to the correctly labelled examples in its class nor the other mislabelled examples.

As we discussed in the introduction, however, this is typically not the case for real-world label noise. The reason for this is that mislabelled examples are present in weakly-supervised datasets due to errors made by the labelling functions used. These labelling functions are deterministic and therefore the errors they make are systematic, not stochastic. For each example that ends up being mislabelled by these labelling functions, there will likely be several similar examples that are also mislabelled for the same reasons. As a result, many mislabelled examples in weakly-supervised are simply not outliers which can explain why outlier detection methods seem to perform so poorly on these datasets.

To illustrate this phenomenon to the reader we offer a concrete example. Fig. 2.1 depicts a T-SNE [27] visualization of the final layer convolutional features [5] of one-thousand weakly-labelled images of propeller planes. These images were taken from the Aircraft-7 dataset which we construct ourselves by searching Flickr for the names of several different classes of aircraft and scraping the most relevant results. Each of these images has been manually categorised as being correctly labelled (shown in blue) or mislabelled (show in red). We can clearly see from this visualization that the vast majority of mislabelled examples are not outlying but densely clustered with among other mislabelled examples.

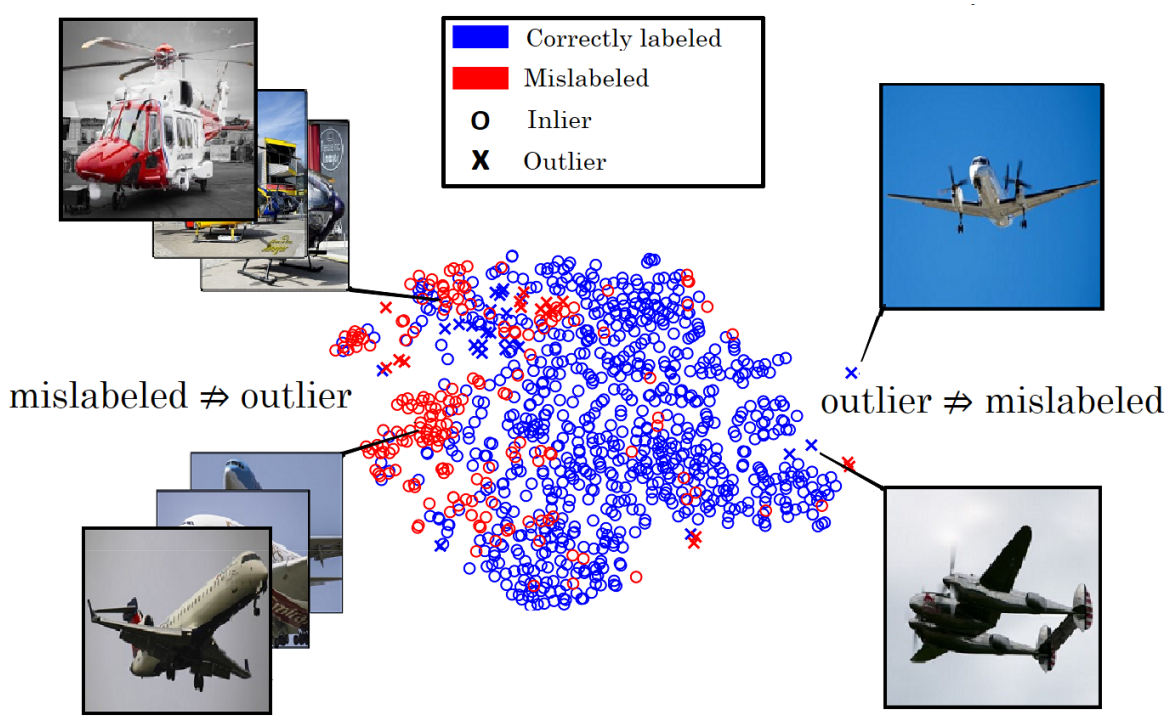

Figure 2.1: T-sne visualization of images from the Aircraft-7 dataset with weak label 'propeller plane'. Mislabelled images (shown in red) are mostly clustered in dense regions which is contrary to the assumptions of outlier detection.

[5]These features were extracted from a ResNet-50 image classifier that was pre-trained on ImageNet.

When we look more closely at the images which make up these clusters, we find that they typically contain images of other types of aircraft or scenes that are related to flying such as an airport or cockpit. This is quite understandable considering that these images were returned to us by Flickr based on the tags that people had assigned to their images. It is not difficult to imagine how the same tags that are assigned to images of propeller planes might also be included for a variety of related images. In this example, as well as several others that we have explored, outlier detection would fail to detect large numbers of mislabelled examples for the simple reason that they are not outliers.

In addition, we have found that the converse assumption that true outliers will be mislabelled is also questionable. To demonstrate this, we calculate the mean proximity of each image to its ten nearest neighbours in our 2D visualisation and use this distance as a rudimentary measure for outlier detection. What we find in this case is that of the 50 most outlying examples (shown in Fig. 2.2), only 16 were truly mislabelled. The remaining 34 examples are what we might consider *corner cases* - correctly labelled but visually atypical in some way. By mistakenly removing these corner cases, outlier detection actually risks damaging the generalisation performance of a classifier by reducing the diversity in the training set.

Figure 2.2: The 50 most outlying 'propeller plane' examples. Contrary to the assumptions of outlier detection, 34 of these outliers are in fact correctly labelled (green border) examples.

2.4 Robust Learning in the Presence of Label Noise

In this chapter, we will discuss methods that attempt to mitigate the negative impact of label noise by altering the learning process in a way that makes it more robust to its presence. As these methods do not attempt to remove or relabel mislabelled examples, they would not be considered to be performing label cleaning and will, therefore, form less of a point of comparison to our work. Nevertheless, several effective methods have been proposed.

2.4.1 Curriculum Learning

The first and most notable of these approaches is curriculum learning (CL). First introduced by Bengio et. al. [28], curriculum learning was initially conceived as a way of reducing the convergence time and improving the final performance of machine learning models. The idea behind CL is that machines, like humans, might learn more efficiently if we introduce training examples in some meaningful order. The natural way to do this would be to begin training on simple examples and gradually introduce more complex cases throughout training. This is, of course, how we design our educational curriculum for children.

Whilst CL methods achieved some promising results with simple machine learning models, they have so far failed to demonstrate significant benefit when used in conjunction with more complex neural networks [**?**]. It seems that heuristic measures of sample difficulty such as training loss are overly simplistic in this case. Recently, however, CL inspired approaches have seen a resurgence for dealing with label noise in weakly-supervised datasets. Rather than removing mislabelled images, these approaches aim to order the training examples in a way that mitigates their negative impact. Two methods in particular have been published in the past year which both achieved state-of-the-art performances on the Webvision dataset [17].

MentorNet [29] uses a secondary *mentor* network which provides a curriculum for its *student* image classifier in the form of a sample weighting scheme. The curriculum is

learnt dynamically using feedback from the StudentNet during training. At the time the paper was published, MentorNet achieved state-of-the-art performance on Webvision, narrowly beating CleanNet [4] which is the state-of-the-art in semi-supervised label cleaning.

Just a few months later, however, CurriculumNet [30] was published which furthered the state-of-the-art on Webvision by 1.3% - a fair margin for such a challenging and widely used dataset. CurriculumNet uses a simple yet surprisingly effective strategy. A measure of the relative density of each image in feature space is used to separate examples into three disjoint subsets of progressively noisier examples. The image classifier is initially trained only on the cleanest subset with images from the remaining groups being added gradually during the later stages of training.

Whilst we cannot deny the apparent effectiveness of these curriculum learning methods for label cleaning, we would like to express some level of concern about the idea of deploying machine learning models that were willingly trained on mislabelled data. The issue as we find it is that whilst curriculum learning may be able to preserve the *accuracy* of a classifier, training on mislabelled examples can lead to many other unexpected and undesirable results. A good example of this comes from one of the earliest experiments we carried out for this work. In this case, we were training a classifier to perform gender classification on a dataset of weakly labelled images we scraped from Flickr. We discovered that whilst the accuracy of the model was good, it would consistently classify images of lorries as being men with high confidence. The reason for this, as it turned out, was that our training set contained several images of 'MAN' branded lorries.

2.5 Summary of the state-of-the-art

To conclude this section on existing work we provide a summary of the state-of-the-art methods form each of the three main categories of approach we have discussed. We have designed our experiments to provide a clear comparison with these three methods.

- **Semi-supervised label cleaning.** CleanNet [4] is the clear state-of-the-art in the category of semi-supervised label cleaning. CleanNet has been shown to outperform other semi-supervised approaches on several benchmark weakly-supervised datasets and achieved state-of-the-art performance on Webvision at the time. Additionally, CleanNet has been shown to perform well even when manual supervision is unavailable for a subset of classes making it the most scalable semi-supervised approach we are aware of.

- **Outlier Detection.** DRAE [22] represents the state-of-the-art in outlier detection for the task of label cleaning. Empirically DRAE was shown to outperform alternative outlier detection methods including UOCL [26] which was the prior state-of-the-art. Crucially, DRAE was also shown to be robust to high proportions of label noise which is commonplace in many weakly-supervised datasets.

- **Curriculum Learning for Label Noise.** CurriculumNet [30] is considered to be the state-of-the-art curriculum learning method for dealing with label noise as it convincingly outperformed MentorNet [29] and achieved state-of-the-art performance on the Webvision dataset [17].

3 Training-ValueNet

In this section of the thesis, we introduce *Training-ValueNet*, our proposed method for automated label cleaning and the primary contribution of this work. Whilst we already explained the underlying rationale of our approach in the introduction (section 1.4), we begin here in section 3.1 by providing an overview of the proposed method. In addition to this, we make a number of important clarifications regarding our approach in section 3.2.

We then proceed in sections 3.3 - 3.5 to describe the full details of our proposed method. Specifically, we begin in section 3.3 by providing a formal definition of *training-value*. In section 3.4 we go on to describe how it is possible to obtain an unbiased, Monte-Carlo estimation for the training-value of each training example directly from episodes of training. Finally, in section 3.5 we introduce the Training-Value approximation network (Training-ValueNet) which learns a mapping between an individual training example (feature vector) and its training-value. The utility of the Training-ValueNet is that once it has been trained, it allows us to estimate the training-values of huge numbers of examples very quickly.

3.1 Overview of Proposed Approach

As we described in section 1.4, our proposed approach for automated label cleaning can be summarised at a high level as follows:

- For any classification task T for which we have a training set X^T and an evaluation set X^V, each training example $x_i \in X^T$ with label y_i possesses some unique *training-value* for the learning process which we denote $V(x_i)$. The training-value is an objective measure of the expected impact that an individual example $x_i \in X^T$ will have on the final performance of our classification model on the evaluation set X^V.

- The fundamental assumption of our approach is that whilst there will naturally be many factors that determine the *magnitude* of an examples training-value, it is the correctness of its class label that primarily determines the *sign* the training-value. Specifically, we assume that correctly labelled examples always offer at least *some* positive value to the learning process whereas mislabelled examples will reliably detriment performance and will, therefore, possess a negative training-value.

- Provided the above assumption proves reasonable, it follows that we can effectively reduce the proportion of label noise in a weakly-supervised dataset by:

    1. Estimating the training-value for all training examples $x_i \in X^T$.
    2. Discarding examples which have a negative estimated training-value from our training set.

3.2 Clarifications

Before we describe the finer details of our method, we feel it is important to briefly clarify some important points regarding what we have proposed so far.

First, please note that the assumption which we made regarding the relationship between training-value and label noise is subject to certain common sense qualifications. The most important of these is that the evaluation set must be accurately labelled for our assumption to be reasonable. If there is instead some non-negligible amount of label noise present in the evaluation set then there is no longer any reason to believe that mislabelled images will necessarily detriment performance on that set. A mislabelled training example that is sufficiently similar to mislabelled examples in the evaluation set could quite easily have a positive impact on performance. The practical implication here is that for our approach to work, we require access to at least a small, cleanly labelled evaluation set. For this reason, we cannot refer to our method as being entirely unsupervised - rather it is unsupervised on the *training set*.

We wish to stress however that there is no need for this validation set to be particularly large for our method to work well. In fact, we have found that one-hundred labelled examples per class is typically more than sufficient. Labelling such a small number of examples is unlikely to present a practical issue and therefore we do not feel this requirement detracts from our goal of automated label cleaning. Additionally, we would point out that no method for automated label cleaning will ever be perfectly reliable and as such we should always be prepared to provide a cleanly labelled evaluation set upon which to assess the performance of our models. For this reason, we would consider our approach to be *minimally supervised*.

Another reasonable requirement is that when estimating training-value, we must not continue to train the classifier once it begins to overfit on the training examples. This is because once overfitting begins, correctly labelled examples may start to detriment performance for reasons unrelated to the correctness of its class label. Once again, however, this does not present a real issue for our approach as we can simply set a

conservative early stopping criteria when estimating training-value.

Finally, we would like to acknowledge our apparent hypocrisy in simultaneously criticizing outlier detection for its reliance on unsubstantiated assumptions whilst we now propose an approach that makes an equally uncertain assumption. We would argue however that the assumption we have made here, at least that mislabelled examples reliably detriment performance is in some way the most minimal assumption one can make when devising a method for automated label cleaning. After all, if one does not assume that mislabelled examples are detrimental to performance then it is unclear what the motivation would be for removing these examples in the first place.

We concede that the converse assumption, which is that correctly labelled examples will always possess a positive training-value *is* more uncertain. It is not difficult to imagine an image for example, that despite being correctly labelled is so distorted or unusual that it would be detrimental to classification performance. We suppose the necessary distinction here is that a correctly labelled example will benefit performance *only* if it is sufficiently representative of the target domain (which is itself represented by the evaluation set). As we shall see in section 6.4, there are situations in which this is not the case for the majority of examples in the training set. At this point, we could easily fall into the rabbit hole of debating what exactly constitutes a correctly labelled example, as this itself is rarely well defined. What we can say about our approach, however, is that provided we have made a reasonably accurate estimation of the training-value, then any 'correctly labelled' examples that we remove in error should at least tend to be those that offer little to no benefit to performance anyway.

This last point can be developed further. Whilst the error in our estimation of training-value will inevitably lead us to make some false-positive and false-negative classifications, we can at least say that the likelihood of this happening for a particular example is negatively correlated with the magnitude of its true training-value. What we mean by this is that the more beneficial or detrimental an example is to performance, the *less likely* we will be to remove or retain it erroneously.

3.3 Definition of Training-Value

Training-value is a function that is well defined for any classification task T with K classes for which we have some set of training examples $X^T = \{x_i, .., x_n\}$ with (potentially weak) labels $Y^T = \{y_i, .., y_n : y \in [1 : K]\}$ and an accurately labelled validation set X^V which represents our target domain.

The training-value of each training example $x_i \in X^T$ is a function of x_i itself, its label y_i and the validation set X^V. It is crucial to note that a training example in isolation does not possess a training-value. It is only with respect to an assigned class label and the validation set that the training-value is defined. Moreover, if either of these changes then so too will the training-value of that example. For brevity however, we will henceforth denote the training-value of example x_i as $V(x_i)$ instead of $V(x_i, y_i, X^V)$.

Let $h(x; \theta)$ be our classification model with weights θ and let L be the loss function which we seek to minimise on X^T using some iterative gradient descent algorithm. We define the training-value $V(x_i)$ of each example $x_i \in X^T$ as the *expected immediate improvement in validation loss* that is obtained as a *direct result* of training our classifier on example x_i at a single time-step t. Whilst this is a valid definition, in theory, in practice it would incredibly difficult to assign the appropriate credit to an individual example if the updates to the weights of the classifier at each time-step depend on more than one example. For this reason, we append to our definition the constraint that at each time-step, updates to the classifier must *only* depend on a single training example.

What this constraint means in practice is that when we estimate the training-value of examples in our dataset, training must be carried out using 'vanilla' stochastic gradient descent [6] i.e. with a batch size of one, no momentum and a fixed learning rate. Under these constraints, the immediate change in validation loss at every time-step t is determined solely by a combination of the following three factors:

[6] We stress that these restrictions on the learning algorithm only apply when we estimate training-values. There are no such restrictions when we conduct final training on the cleaned dataset.

- The training example $x^{(t)}$ which we train on at time t.
- The current model weights $\theta^{(t)}$.
- Stochastic factors such as dropout regularization [31].

Given this, we can express the immediate improvement in validation loss $\Delta L(X^V)$ obtained by training on example $x^{(t)}{=}x_i$ at time-step t, as an expectation over the updated network weights $\theta^{(t+1)}$ denoted by the random variable $\theta^{'}$,

$$\Delta L(X^V \,|\, x^{(t)},\, \theta^{(t)}) = E_{\theta'}\Big[\, L(X^V \,|\, \theta^{(t)}) - L(X^V \,|\, \theta^{'}) \,\Big] \tag{2}$$

Finally, we arrive at our definition of the training-value $V(x_i)$ of example x_i by taking the expectation of this immediate change in loss over *all possible values* for the current weights $\theta^{(t)}$ which we will denote by the random variable θ,

$$V(x_i) = E_{\theta}\Big[\, \Delta L(X^V \,|\, x^{(t)}{=}x_i, \theta^{(t)}{=}\theta) \,\Big] \tag{3}$$

3.4 Monte-Carlo Estimation of Training-Value

Whilst explicit evaluation of equation 3 (i.e. of the training-value) would be intractable for any non-trivial problem, we can obtain an unbiased estimate of the training-value for each example using a straightforward process which we will describe here. This process is what we shall refer to moving forward simply as the Monte-Carlo estimation phase. It is perhaps most easily described by detailing the individual steps it requires:

1. Extract a suitable set of features $f(x)$ for all examples $x \in X^T \cup X^V$ in the dataset. For an image classification task, we might pass each image x through a CNN and extract the final layer features $f(x)$ as a representation of the image.

2. Initialise a simple feed-forward neural network $h(f(x))$ to act as our classifier.

3. Train the classifier on the training set X^T using the vanilla stochastic gradient descent algorithm as per the constraints laid out in the previous section. Train for M repeated episodes of e epochs each, randomly shuffling the training examples at the start of each epoch.

4. At each iteration n, compute the immediate improvement in validation loss [7] $\Delta L(X^V)^{(n)}$ and store this alongside the example $x^{(n)}$ that was trained on at that iteration in a cache $\Big\langle (x^{(1)}, \Delta L(X^V)^{(1)}), ..., (x^{(M \cdot e)}, \Delta L(X^V)^{(M \cdot e)}) \Big\rangle$. Note here that the *iteration* n is not reset to zero at the start of each new episode which is why it ranges from 1 to $M \cdot e$.

5. Once all episodes of training are complete, the estimated training-value $\bar{V}(x)$ for each training example $x \in X^T$ is simply the *mean immediate improvement* in validation loss that was observed when example x was trained on. That is,

$$\bar{V}(x_i) = \frac{1}{M \cdot e} \sum_{n=1}^{M \cdot e} \big\{ \Delta L^{(n)}(X^V) \mid x^{(n)} = x_i \big\}. \tag{4}$$

[7] Note that in practice we typically compute changes in loss on a *subset* of n_v examples per class from the validation set. This is simply to reduce the computational overhead of this step.

3.5 Training-Value Approximation Network

In the previous section we described a simple process which can be used to obtain unbiased, Monte-Carlo estimations of training-value directly from episodes of training. The issue with this process, however, is that for very large datasets it can become prohibitively expensive to obtain reliable estimates in this manner for all training examples. It is for precisely this reason that we introduce the Training-Value approximation network (Training-ValueNet) which we shall describe in this section.

The Training-ValueNet is simply a multilayer perceptron (MLP) regression network that will be trained to predict the training-value of an example directly from its feature vector. We must bear in mind however, that the training-value of a particular example is also conditional on its assigned class label. This means that we cannot accurately predict the training-value of an example without knowledge of its class label. Perhaps the most natural way of introducing this dependency into the Training-ValueNet would include an additional input which conveys the class label of the example. In this work, however, we have opted to use a different approach which is to simply learn a separate Training-ValueNet for each class. Whilst this may seem wasteful, each Training-ValueNet is such a small network that it is incredibly fast to train and it requires little memory to store their weights after training. In this case, we will, in fact, be training a separate Training-Value-Net $\hat{V}^C$ for each class $C \in 1{:}K$,

$$\hat{V}^C : \{f(x) : x \in X^T, y = C\} \mapsto \mathbb{R}$$

Training each Training-ValueNet is a simple process. We first select a random subset of n_t training examples per class. We then obtain Monte-Carlo estimations of training-value $\bar{V}(x)$ for all examples in this subset. Finally, we train the Training-ValueNet for each class using the estimates of training-value for examples from that class as the training targets. Once we have trained the Training-ValueNet for each class, we use it to predict the training-value for *all* training examples which have that corresponding class label.

The Training-ValueNet allows us to learn a *general* mapping between the feature space of our training data and the training-value. As such, it enables us to estimate the training-values of huge numbers of examples within a reasonable amount of time [8].

[8] We provide a detailed account of the computational complexity of the algorithm in section 6.2.

4 Experiments

4.1 Datasets

In order to empirically evaluate the effectiveness of our proposed approach we will, of course, require some datasets containing label noise. As previously stated in section 2.2, we have chosen *not* to conduct any of our experiments using label noise that is artificially generated. Despite this being a convenient and frequently used approach, we strongly feel that this type of noise poorly reflects the patterns of label noise we see in *real* weakly-supervised data and as such, can lead to invalid conclusions. Instead, we have conducted our experiments on the following two weakly-supervised datasets in which any label noise is naturally occurring as the result of errors made by the sources of weak supervision.

4.1.1 Clothing 1M

The clothing 1M dataset [18] contains precisely 1M weakly labelled training examples belonging to 14 categories of clothing. These images were scraped from several shopping websites including eBay and Amazon and their labels were assigned according to the presence of key-words in the surrounding text. A further 50K/14K/10K manually labelled images are included for train/validation/test purposes. Of these images, 25K/7K/5K have *both* their weak and ground-truth labels provided. By comparing these two sets of labels, we obtain the ground truth *correctness* of each of the weak labels. These 'verification' labels allow us explicitly evaluate the ability of Training-ValueNet to identify mislabelled examples.

Experimental results on Clothing 1M Clothing 1M [18] have been reported for each of the three state-of-the-art methods we wished to compare our method with; CleanNet [4], DRAE [22] and CurriculumNet [30]. This made it a natural choice of dataset for us.

4.1.2 Aircraft-7

In order to perform a comparison between weakly-supervised learning using our method with the fully-supervised paradigm, we decided to create a new large scale, weakly-supervised dataset from scratch. The resulting dataset which we call Aircraft-7 contains 75K images of 7 types of aircraft. These classes were picked because they were also available synsets in ImageNet. This ensured that we had a plentiful supply of cleanly labelled data with which we used to form validation and test sets as well a fully-supervised training set for the purposes of our comparison.

To obtain the training examples for the Aircraft-7 dataset we searched Flickr for the name of each class in turn and ordered the search results by relevance. We used publicly available bulk image downloader software [32] to quickly scrape as many images as possible for each class. In order to overcome a restriction on the number of images that can be downloaded at one time, we searched for images uploaded in each year for the past 20 years. In the end, we were able to collect a total of 75K images for this dataset. A breakdown of the number of examples obtained per class dataset is shown in Table 4.1. We also list an estimate for the mean label accuracy in each class which was calculated by manually verifying 250 images from each. The mean estimated label accuracy across all images in the dataset is 70%.

To obtain accurately labelled evaluation sets we set aside 100 images per class from the corresponding ImageNet synset to form a validation set and a further 100 per class for a held-out test set. All remaining ImageNet images for these classes were then combined to form our fully-supervised training set which we used for the purposes of comparison only.

Aircraft Dataset Statistics		
Class	num images	est. label accuracy (%)
Airliner	10,968	87
Fighter Jet	8,882	88
Glider	13,175	54
Helicopter	10,795	75
Prop Plane	11,309	85
Sea Plane	14,891	55
Stealth Bomber	4,597	46
Overall	**74,617**	**70**

Table 4.1: Statistics for the Aircraft-7 dataset.

4.2 Experimental Setup

We evaluate our proposed method via a series of three experiments, each of which assesses a slightly different aspect of the approach. In this section, we will explain the motivation behind each experiment and describe the experimental setup we use. To avoid repeating ourselves here, we shall defer a thorough description of the finer training details and hyperparameters we used for the following section (4.3) as the same settings were used for each experiment. Results for each experiment are then provided in section 4.4.

4.2.1 Experiment 1 - Label Noise Detection

In our first experiment, we evaluate our method on a label noise detection task. We wish to know simply how effective our method is at identifying mislabelled examples and how this performance compares with existing label cleaning methods. Accurate label noise detection is perhaps *the* single most important criteria for our method as this is the means by which we propose to reduce label noise and subsequently improve final classification performance.

We use the Clothing 1M dataset for this experiment. Specifically, we use the 25K images

we described in section 4.1.1 for which we have obtained the ground-truth *correctness* of their weak labels. We assign each of these 25K examples their weak label and use this as the training set. For the validation set, we use the designated manually labelled set of 14K examples.

We first train our baseline CNN image classifier on the entire weakly-labelled dataset from which we extract the final convolutional layer features for each image. We then apply our method, learning a Training-ValueNet for each class which we then use to predict the training-values of all 25K examples. We classify any example that has a predicted value of *less* that zero as mislabelled and the rest as correctly labelled. We compare our predictions for the correctness of each label with its ground-truth correctness and compute our mean detection error across the 25K examples. We compare our method with a number of existing label cleaning methods including CleanNet [4] and DRAE [22] which you will recall are the state-of-the-art methods for the semi-supervised and outlier detection approaches respectively. Results for all methods we compare with were reported in [4].

4.2.2 Experiment 2 - Image Classification

For our second experiment, we assess the extent to which our method improves the performance of an image classification model trained on weakly-supervised data. To do this, we take the already trained Training-ValueNets from experiment 1 and use them to predict the training-value of all 1M weakly-labelled training examples in the Clothing 1M dataset. We discard any examples with a negative predicted training-value and proceed to fine-tune our baseline image classifier on this cleaned training set.

We compute the accuracy of this final model on the test set of 10K examples. We compare this with the baseline model (trained on all 1M weakly-labelled images), a fully-supervised baseline (trained on the 50K manually labelled training images) as well as that achieved using several other methods for label noise.

4.2.3 Experiment 3 - Comparison with Fully-Supervised Learning

In this final experiment, we compare weakly-supervised learning using our method for label cleaning with the fully-supervised learning paradigm. We are interested not only in the level of performance that each achieves but also what the entire pipeline looks like for each. It is for this reason that we opted to create our own weakly-supervised dataset, Aircraft-7, from scratch (see section 4.1.2 for details).

As per our method, we begin by training a baseline image classifier on the entire weakly-labelled training set and extract the final convolutional layer features for each image. We also record the performance of this baseline on our held-out test set. We then apply our method and obtain Training-ValueNets for each of the 7 classes. These are then used to predict the training-value of all 75K training examples in our weakly-supervised training set. After discarding any examples with a negative predicted training-value, we proceed to fine-tune our baseline model on the cleaned training set. Again we record the accuracy this model achieves on the held-out test set.

For the fully-supervised comparison, we train the same image classification model on the fully-supervised training set that we took from ImageNet. We record the final accuracy of this classifier on the held-out test set. We then compare the results of each condition; weakly-supervised learning, weakly-supervised learning with our label cleaning and fully-supervised learning.

4.3 Training Details and Hyperparameters

In this section, we describe in detail the specific training details and hyperparameters settings we used for our experiments. The setup described here was used for all three of the experiments we conducted. All models were implemented in Python using Keras [33] with a TensorFlow backend [34]. Training was carried out using NVIDIA Tesla K80 GPUs on Amazon web-services. It is worth mentioning here that the Monte-Carlo estimation phase of our algorithm can be easily distributed across multiple machines as each episode of training is independent of every other episode. We took advantage of this and distributed the computation during this phase across eight GPUs. This substantially reduced the absolute run time for our experiments.

4.3.1 Baseline Image Classifier

Our experiments require that we choose an image classification model. To provide the closest comparison with recent works [4], we follow in their steps by using ResNet-50 [35] as our baseline model. Rather than training from a random initialization, we follow the common practice of fine-tuning a model that has already been pre-trained on ImageNet. Specifically, we use the pre-trained model provided within Keras [33]. In each experiment, we simply replaced the final dense layer with one that has the same number of units as there were classes in the dataset.

Each time we trained the ResNet-50 classifier on a *cleaned* training set, we opted to fine-tune the baseline model that had already been trained on the entire weakly-supervised training set rather than starting from the ImageNet pre-trained weights. Empirically both approaches achieved very similar performance however fine-tuning was far quicker to carry out. We follow the fine-tuning procedure described in [19], changing only the batch size from 32 to 64, again for more efficient GPU usage.

4.3.2 Monte-Carlo Estimation of Training-Value

The next stage in our method is the Monte-Carlo estimation phase in which we obtain direct estimates for the training-values of a random subset of examples. You will recall that the purpose of this stage is to obtain estimates that can be used as regression targets to train a Training-ValueNet for each class.

We select a random sample of n_T training examples from each of the K classes in the dataset. For each image, we use the final convolutional layer features that we extracted from our baseline model as the input to a small, fully-connected neural network classifier with no hidden layers. We train this classifier on the subset for M episodes of e epochs. As per the constraints described in section 3.3, we train this classifier using the 'vanilla' stochastic gradient descent (SGD) algorithm i.e. with a batch size of one, no momentum and a constant learning rate.

At each time-step during the training process, we compute the immediate *improvement* in the loss on a subset of the validation set. Using only a subset of validation examples simply reduces computational overhead. This subset contains a random sample of n_v examples per class. As with the training subset, the same subset of the validation data is used throughout all M episodes.

In all three of our experiments we use the following setting for these hyperparameters:

- M = 100 episodes of training.
- e = 1 epochs per episode [9].
- n_T = 1000 training examples per class.
- n_V = 100 validation examples per class.

An exhaustive list of other miscellaneous training details is provided in Table 4.2.

[9]The reason for using just one epoch of training per episode is that the small classifier converges very quickly when trained on pre-extracted convolutional features and we wanted to avoid any overfitting which could lead to invalid point estimates of training-value.

4.3.3 Training-ValueNet

Once we have obtained Monte-Carlo estimates for the training value of these n_T images in each class we use them to train the TrainingValue-Network for that class. The Training-ValueNet is simply a multilayer perceptron (MLP) regression network. It takes as input, the pre-extracted final convolutional layer features for a single image in a particular class and returns a real-valued prediction of the training-value for that image.

In all three experiments, we use a single hidden layer of 1024 neurons. Empirically, we find that a single hidden layer improves performance but that adding further layers provides no additional benefit. We add dropout regularization [31] at a rate of d=0.7 immediately after this hidden layer which helps prevent the network overfitting on what is a relatively small training set of n_T examples. All other relevant hyper-parameters are listed in Table 4.2.

Parameter Name	**Monte-Carlo Estimation Model**	**Training-Value Network**
Network Parameters		
Input Dimension	2048	2048
Hidden Layers	None	Single layer (1024 units)
Dropout [31]	None	After hidden layer (d=0.7)
Output Activation	Softmax	Linear
Output Dimension	K (# classes)	1
Training Parameters		
# Training Examples	1000 per class	800
# Validation Examples	100 per class	200
Episodes of Training	M = 100	1
Epochs per Episode	1	100
Optimization Parameters		
Optimization algorithm	SGD	SGD
Loss Function	Categorical-Crossentropy	Mean Absolute Error
Batch Size	1	32
Momentum	None	0.9 (Nesterov [36])
Learning Rate	0.01	0.01
Decay Rate	0	1e-3
Early Stopping	N/A	After 10 epochs of no improvement in val. loss

Table 4.2: List of hyperparameters used for our experiments.

4.4 Experimental Results

4.4.1 Experiment 1 Results - Label Noise Detection

The full set of results for this label noise detection task are shown in Table 4.3. Training-ValueNet achieved an average error rate of 23.66% across the 25K examples used in this experiment. As the first point of comparison, this is 14.8 percentage points (pp) better than the 38.46% error obtained using a naive baseline which simply classifies all examples as correctly labelled.

Training-ValueNet outperforms DRAE [22], the state-of-the-art outlier detection method by a substantial 15.3pp. This demonstrates the superior effectiveness of our approach over outlier detection whilst providing support for our suggestion that outlier detection is an inherently unsuitable approach for label cleaning. Our method also outperforms the best performing unsupervised method, the CleanNet unsupervised baseline by 6.9pp. Unfortunately, we do not quite reach the performance of the semi-supervised methods, of which CleanNet fared the best with a 15.77% detection error. We stress, however, that each of these methods benefited from 250 hand-labelled examples *per class* whereas our method required no manual supervision on the training data.

Label Noise Detection Results	
Method	Detection error (%)
Semi-supervised methods:	
MLP	16.09
kNN	17.58
SVM	16.75
Label prop [37]	17.81
Label spread [38]	17.71
CleanNet [4]	15.77
Unsupervised methods:	
Naive baseline	38.46
DRAE [22]	38.95
CleanNet - unsup. base [4]	30.56
Training-ValueNet	**23.66**

Table 4.3: Label noise detection results on Clothing 1M.

If we remove all training examples which we classify as mislabelled, then the mean accuracy of labels for the remaining images increases from 61.7% to 78.7% (17% increase). This is precisely the kind of improvement in label accuracy that we hope will lead to improvements in the final performance of our classifier. This is what we evaluated in experiment 2.

4.4.2 Experiment 2 Results - Image Classification

In this experiment, the baseline ResNet-50 image classifier that was trained on the entire weakly-labelled training set achieves 68.88% accuracy on the test set. This is within 0.06% of the baseline performance reported in [19]. In comparison, training the same ResNet-50 model on a separate set of just 50K hand-labelled yielded a substantially higher 75.19% accuracy. This demonstrates the detrimental impact of the label noise in this dataset.

We proceed to discard 323K training examples that received a negative predicted training-value. We then fine-tuned our baseline ResNet-50 model on this cleaned dataset and observed a 72.03% accuracy on the test set. This represents a +3.15 percentage point increase over the baseline despite having discarded a third of the training examples. Our method outperforms [19] by 2.19pp despite their benefiting from the use of hand-labelled examples which they use to estimate the confusion between classes. Once again, however, we fall slightly short of the performance of CleanNet (2.66pp) but we stress that CleanNet used 250 hand-labelled training examples per class whilst we used none.

We report a further set of results in Table 4.4 for which we perform a final round of fine-tuning on the 50K hand-labelled training examples. It has been shown in several papers that this can yield additional improvements. Whilst this is not in the true spirit of what we a trying to achieve in this work (training without human supervision) we follow suit to provide a full comparison with other methods. We observe a further 6.03pp increase in test accuracy as a result of this fine-tuning. CurriculumNet

[30] achieves the highest performance overall at 81.50%. Unfortunately, the authors of CurriculumNet never reported the performance that their model achieved *without* this additional 50K images meaning we cannot provide a comparison with CurriculumNet in the weakly-supervised setting.

These results clearly demonstrate the effectiveness of our method for improving image classification performance through label cleaning. We have come within 2.19pp of the state-of-the-art in semi-supervised label cleaning despite requiring no supervision.

4.4.3 Experiment 3 Results - Comparison with Fully-Supervised Learning

In this final experiment, we compare weakly-supervised learning (both with and without our method for label cleaning) with the fully-supervised paradigm. Full results are shown in Table 4.5. The baseline model that was trained on our Aircraft-7 dataset achieved 82.4% accuracy on the held-out test set. In comparison, the fully-supervised training set that was trained on the examples from ImageNet recorded a much higher 88.6% accuracy despite containing less training examples.

Image Classification on Clothing 1M					
#	paper	method for noise	init.	training set	accuracy (%)
1	[19]	noisy baseline	ImageNet	1M	68.94
2	ours	noisy baseline	ImageNet	1M	68.88
3	[19]	clean baseline	ImageNet	50K	75.19
Training on noisy 1M only					
4	[18]	loss correct.	ImageNet	1M	69.84
5	[4]	CleanNet	ImageNet	1M	74.69
6	ours	Training-ValueNet	#2	1M	72.03
Training on additional clean 50K					
7	[18]	None	#4	50K	80.38
8	[4]	None	#5	50K	79.90
9	[30]	CurriculumNet	ImageNet	1M + 50K	81.50
10	ours	None	#6	50K	78.06

Table 4.4: Image classification results on Clothing 1M. Results achieved using the additional 50K cleanly labelled images are segregated to avoid confusion with weakly-supervised learning.

We proceeded to discard approximately 20K training examples from the Aircraft-7 dataset that received a negative predicted training-value. We then fine-tuned the weakly-supervised baseline model on the remaining images and observed an 87.0% test accuracy, a significant 4.4pp increase. This final accuracy is just 1.6pp lower than that achieved using several thousand manually labelled examples.

To directly assess the impact our method has had on the amount of label noise in the Aircraft-7 dataset, we manually annotate the label correctness for a random sample of 500 images per class from the cleaned training set. We find that the mean label accuracy across all classes is 85.4% in this cleaned dataset, 15.3pp higher than it was prior to label cleaning. The label accuracy is broken down by class in Table 4.6.

The results of this final experiment demonstrate that by using our proposed method for automated label cleaning, weakly-supervised learning *is* able to achieve performance comparable with fully-supervised learning. This is despite our method requiring *no human supervision on the training set*. The entire process of creating the Aircraft-7 dataset, performing automated label cleaning and training the final classifier took only a matter of hours during which our attention was scarcely required. This is in stark contrast with the many hours of tedious labour it would have taken to label the 75K images manually.

Image Classification on Aircraft-7				
#	Method	init.	Training set	accuracy (%)
1	Weakly supervised baseline	ImageNet	noisy 75K	82.4
2	Supervised baseline	ImageNet	clean ImageNet	88.6
3	Training-ValueNet	#1	noisy 75K	87.0

Table 4.5: Image classification results on Aircraft-7 dataset.

One of the most beneficial aspects of our method which we have yet to mention is that once a Training-ValueNet has been trained for each class they can be readily applied to new example as and when they become available. We only scraped 75K examples for the Aircraft-7 dataset because that was all the time we could afford to spend. If we were to collect a million more images tomorrow, however, we could use our Training-ValueNets to automatically clean these examples at *no additional expense*.

Aircraft-7 Dataset Statistics				
Class Name	Before Label Cleaning		After Label Cleaning	
	# Images	Label acc. (%)	# Images	Label acc. (%)
Airliner	11.0k	87.3	8.7k	99.1
Fighter Jet	8.9k	88.3	7.1k	88.6
Glider	13.1k	53.5	7.6k	77.9
Helicopter	10.8k	74.6	9.5k	85.7
Prop Plane	11.3k	85.1	10.0k	82.9
Sea Plane	14.9k	55.3	8.2k	79.3
Stealth Bomber	4.6k	46.7	2.1k	78.8
Overall	74.6k	70.1	53.2k	85.4

Table 4.6: Label accuracy for each class of the Aircraft-7 dataset before and after label cleaning.

5 Conclusion

In May of this year, renowned reinforcement learning researcher Richard Sutton published a short article entitled 'The bitter lesson' [39]. It begins as follows:

> *"The biggest lesson that we can learn from seventy years of AI research is that general methods that leverage computation [and learning] are ultimately the most effective, and by a large margin. "*

The article goes on to explain how progress within each area of AI research has followed a similar pattern. At first, when data is scarce or computation limited, researchers seek to improve their system by exploiting their own knowledge of the problem. Whilst this almost always helps in the short term, performance inevitably plateaus as the system becomes constrained by the knowledge of its creators. In the end, a breakthrough eventually occurs using a more general-purpose method that scales with increased computation and data.

In the present afterglow of the deep learning revolution, it certainly seems that we may finally be learning this bitter lesson. From feature engineering and selection, to hyperparameter tuning and even model design, we are increasingly removing our knowledge and intuition from the equation and accepting that in order to build intelligent systems, we must enable machines to *learn* as we learn, not know what we know.

And yet, for all the incredible achievements of deep learning in recent years, the vast majority of machine learning being done today still relies on a huge amount of human knowledge in the form of data labelling. For the past year I have been working within the machine learning industry and I have seen several perfectly viable projects grind to a halt because of the time and financial cost of data labelling. The motivation for this work stems from an overwhelming feeling that we can and must do more to remove this barrier.

As we examined existing attempts to automate label cleaning, we discovered that they too are limited by an over-reliance on human knowledge. Semi-supervised methods are effective in *reducing* the amount of labelling required but this can only ever be a partial solution. Outlier detection, on the other hand, requires no human intervention but performs poorly due to the flawed assumptions that researchers have made about the nature of label noise in weakly-supervised datasets. The work we have presented in this thesis serves as a conscious move *away* from this reliance on human knowledge for label cleaning.

The primary contribution of the work is Training-ValueNet, a simple yet effective method which leverages computation and learning to remove label noise from weakly-supervised datasets *without* human supervision. Our results clearly demonstrate that Training-ValueNet is able to accurately identify and remove mislabelled examples and improve the performance of state-of-the-art image classification algorithms as a result. On the Clothing 1M and Aircraft-7 datasets, Training-ValueNet reduced the proportion of mislabelled examples by 17.1 percentage points (pp) and 15.3pp respectively. This led to improvements in test accuracy of 3.15pp for Clothing 1M and 4.4pp for Aircraft-7.

Of equal importance to us, however, is the fact that we have shown Training-ValueNet to be a genuinely useful tool that can be easily integrated into the machine learning pipeline. Using Training-ValueNet we collected and cleaned a weakly-supervised dataset of 75K images in a matter of hours, during which our attention was scarcely required. This is in stark contrast to the many hours of tedious work it would otherwise take to clean these labels manually. What's more, the final classification test accuracy that was achieved using this dataset came within just 1.6pp of the accuracy achieved using thousands of hand labelled images. We have released an easy-to-use implementation [10] of Training-ValueNet in the hope that researchers and practitioners in the field can benefit from this tool.

[10] `https://github.com/lukasmyth96/Training-ValueNet`

Training-ValueNet is not without its limitations. We shall spend the final chapter of this thesis discussing some of the many ways in which our work could be improved and extended in the future. What we like would draw the readers attention to, however, is the performance we were able to achieve *in spite* of these limitations. We feel as confident now as we did when beginning this project that a general purpose method leveraging computation and learning will one day replace the need for burdensome data labelling. We hope that this work may act as a small first step towards achieving this important goal.

6 Future Work

In this final chapter, we present some ideas for how this line of work could be extended in the future. The ideas put forward in sections 6.1 - 6.3 *are* merely ideas and suggestions. In sections 6.4 and 6.5 however, we present findings from some preliminary investigations that we have carried out in order to demonstrate to the reader why we feel these directions are interesting.

6.1 Beyond Image Classification

Whilst the experiments carried out in this work have focused on image classification, the method we have presented here is general purpose and can be readily applied to *any* classification task. Therefore, a natural starting point for future work would be to do just that. There are many classification tasks in the realm of natural language processing (NLP) in particular for which we feel our method would be particularly suitable. Our reasoning for this is twofold. First of all, textual data is so incredibly abundant and accessible on the internet that it is often possible to scrape enormous amounts of data for a text classification task. That being said, the process of labelling these examples remains no less of an issue than it is for image classification. It can, however, be very easy to write labelling functions for text data which makes it a prime candidate for weak supervision. Both of these factors make text classification tasks a prime candidate for our method. We have designed our open-source implementation of Training-ValueNet with these possible applications in mind.

6.2 Improving Computational Efficiency

We do not shy away from the fact that our algorithm is rather computationally expensive. It was after all our intention to shift the burden of label cleaning *away* from humans onto machines that will only become cheaper and more powerful over time. Furthermore, in all the experiments we have conducted, the time and financial cost required to perform label cleaning with Training-ValueNet remains substantially less

than it would have been if label cleaning was performed manually. In addition to this, the absolute run time of our algorithm can be drastically reduced as each episode of the Monte-Carlo estimation phase can be run independently on separate machines.

Nevertheless, there are cases in which Training-ValueNet would currently be prohibitively expensive to run. This is due to the fact that during the Monte-Carlo estimation phase we perform $n_t \cdot K$ training iterations per epoch and compute the change in loss on $n_v \cdot K$ validation examples at each iteration. The complexity of the algorithm is therefore $\mathcal{O}(K^2)$ where K is the number of classes. This means that whilst the run time is independent of the number of examples per class (as n_t can remain fixed), it scales very poorly with the number of classes K. At the present time Training-ValueNet is therefore not well suited for tasks with many classes (e.g. > 100).

An important direction for future work is therefore to find ways of improving the computational efficiency of the algorithm with respect to the number of classes. The most promising way of doing this would be to increase the batch size from one during the Monte-Carlo estimation phase. The difficulty with doing this, of course, is that you encounter a credit assignment problem in trying to determine precisely how each training example in the batch contributed to the change in validation loss. This problem is not insurmountable however and exploring ways to deal with this credit assignment problem should be a top priority for future work.

6.3 Dynamic Training-Value

One of the main limitations of the method we have presented is that training-value, as we have defined it here, provides only a static estimation of impact an example will have on performance. In reality, however, it is likely that the value an example offers will vary throughout the course of training. This is, of course, the entire premise behind curriculum learning [28]. In traditional curriculum learning, however, the default assumption is that we should train on *easy* examples first before progressing to more difficult examples later in training. Whilst this seems reasonable in theory, it is likely a gross over-simplification of how an example's true value changes over time. Recent

works have indeed shown that traditional curriculum learning has very little impact on deep learning models [40].

Taking all this into account, *the* single most interesting line of future work in our opinion is the potential to introduce an additional dependency between the training-value of an example and the state of the classifier during training. What this would do is allow the training-value of an example to vary throughout the course of training in precise accordance with how its impact on performance *actually* changes. It would then be possible for the curriculum to adapt dynamically, with examples being selected for each batch based on the value they offer at that exact point in time.

6.4 Cross-category vs. Cross-domain Label Noise

When reviewing the literature on label noise, we noticed that whilst there have been many investigations into the impact of label noise in general, there has been little attempt to consider how different *types* of label noise might impact performance differently. One of the few mentions we have seen of sub-categories of label noise is from [41]. In this paper, the authors briefly mention the difference between cross-*category* and cross-*domain* label noise. They use the term cross-category noise to describe examples which are mislabelled but actually belong to another class within the classification task. In contrast, cross-domain noise is used to describe examples for which no suitable label exists among the set of classes used in the task. Fig. 6.1 shows a set of example images for each type of noise taken from the 'propeller plane' class of the Aircraft-7 dataset.

Whilst weakly-supervised datasets will generally contain some amount of each type of label noise, it is not necessarily the case that each type should be equally detrimental to performance. If it was to be demonstrated that one type is significantly more detrimental than the other then the entire discussion surrounding the impact of label noise would need to account for this factor. Unfortunately, we have found no existing work where this possibility has been investigated. Therefore, since our entire method

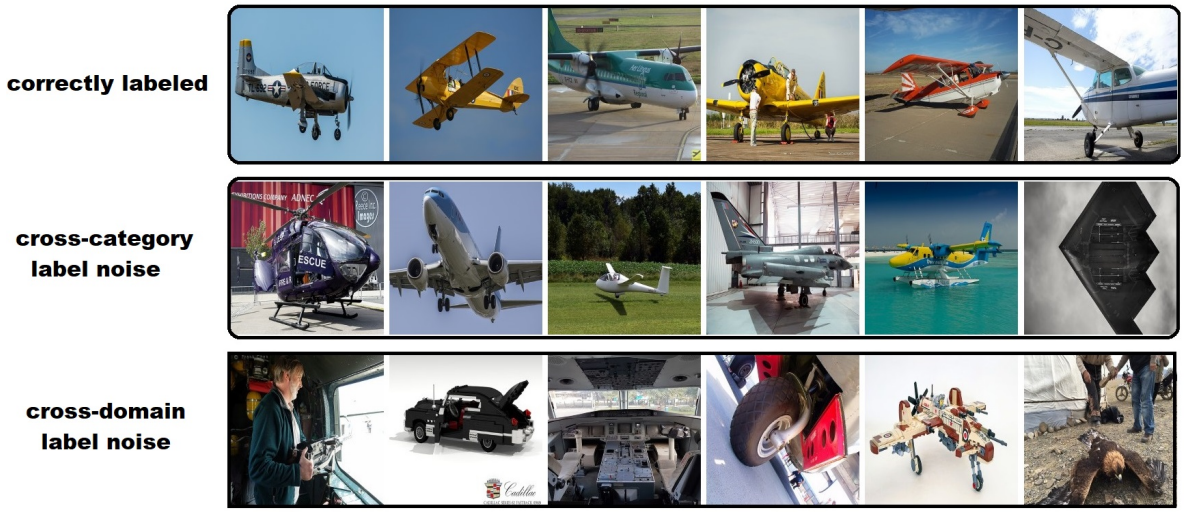

Figure 6.1: Examples of cross-domain and cross-category label noise from the 'propeller plane' class of the Aircraft-7 dataset.

revolves around estimating the impact that each training example has on performance we felt we are well placed to conduct an initial investigation into this matter.

To do so we began by manually classifying a random sample of images from our Aircraft-7 dataset as being either correctly labelled, cross-domain label noise or cross-category label noise. We continued until we had precisely 500 images for each of the three categories. Using the training-values that we obtained for each image during experiment 3 we first calculated the mean training-value for each of the three groups. A box plot of these averages is presented in Fig. 6.2.

As expected, the mean training-value for correctly labelled images is a small positive value. When we compare the two types of label noise however, we find that cross-category label noise is *substantially* more damaging to performance than cross-domain noise ($\approx 7\times$ more). To confirm that this difference is reflected in the actual impact that each type has on classification performance, we perform a small experiment. We begin by training the classifier only on the correctly labelled 500 images. In each of two separate conditions, we then proceed to add 100 mislabelled images at a time over the course of five episodes from *either* the cross-domain or cross-category group. At the end of each episode, we record the maximum accuracy obtained on the ImageNet validation set. This maximum performance is plotted against the label noise ratio (i.e. the ratio of mislabelled to correctly labelled images) in Fig. 6.3.

We can see from Fig. 6.3 that adding cross-domain label noise has a very small negative effect on performance. Cross-category label noise, on the other hand, causes a substantial detriment to performance. By the time all 500 cross-category mislabelled images are added to the training set the validation accuracy decreases from 82.7% (with no label noise) to 67.3% (-15.5pp). This supports the finding that cross-category label noise is substantially more detrimental than cross-domain label noise.

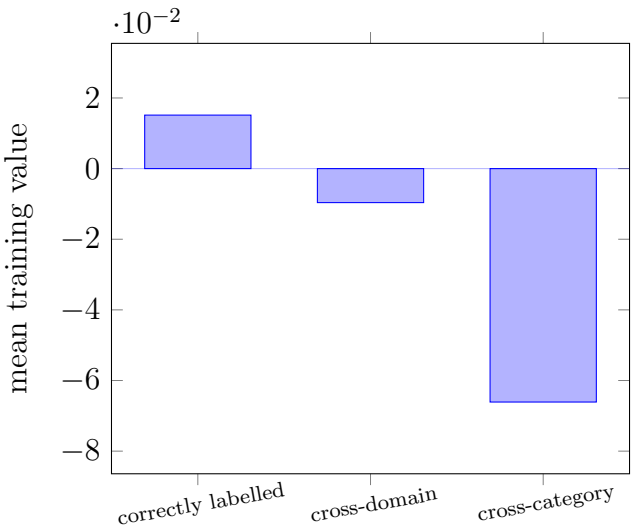

Figure 6.2: Box plot showing the mean training-value for 500 correctly labelled training examples from the Aircraft-7 dataset as well for 500 examples of cross-domain and cross-category noise respectively.

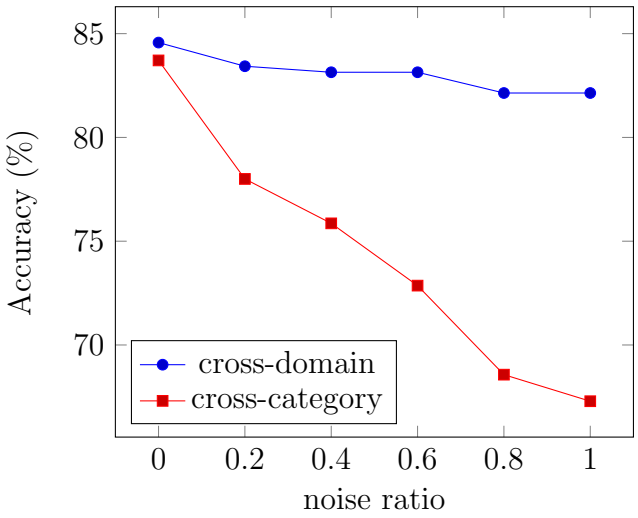

Figure 6.3: Plot showing how classification performance decreases when different types of label noise are added to the training set.

The finding that cross-category noise is more detrimental is quite intuitive. It is understandable why an example that truly belongs to one class and has been mistakenly labelled as another would be more confusing to the classifier than another mislabelled example that truly belongs to no available class. What we have shown here, however, is that the difference in impact between the two types can be quite substantial. This suggests that we should perhaps be less concerned with the absolute amount of label noise in our dataset than how that noise is divided between these types.

So far as this finding concerns our method, we can at least say that the more damaging an example is (i.e. the lower its training-value) the more confident we can be that Training-ValueNet will successfully identify and remove that example from the dataset. We should, therefore, be particularly successful at removing the more damaging cross-category label noise from a dataset.

6.5 Training-ValueNet for Domain Shift

In this thesis, we have focused on the issue of label noise in weakly-supervised datasets. Unfortunately label noise is not the only difficulty that can be encountered when performing weakly-supervised learning. In cases where we have a specific target domain in mind for our model but then collect training data from a more general domain (e.g. by scraping data from the web) we often find that there is a significant *domain shift* between our training and target domains. This means that there is a difference between the underlying statistical distribution of the two sets. Like label noise, domain shift has also been shown to significantly hurt classification performance [42, 43].

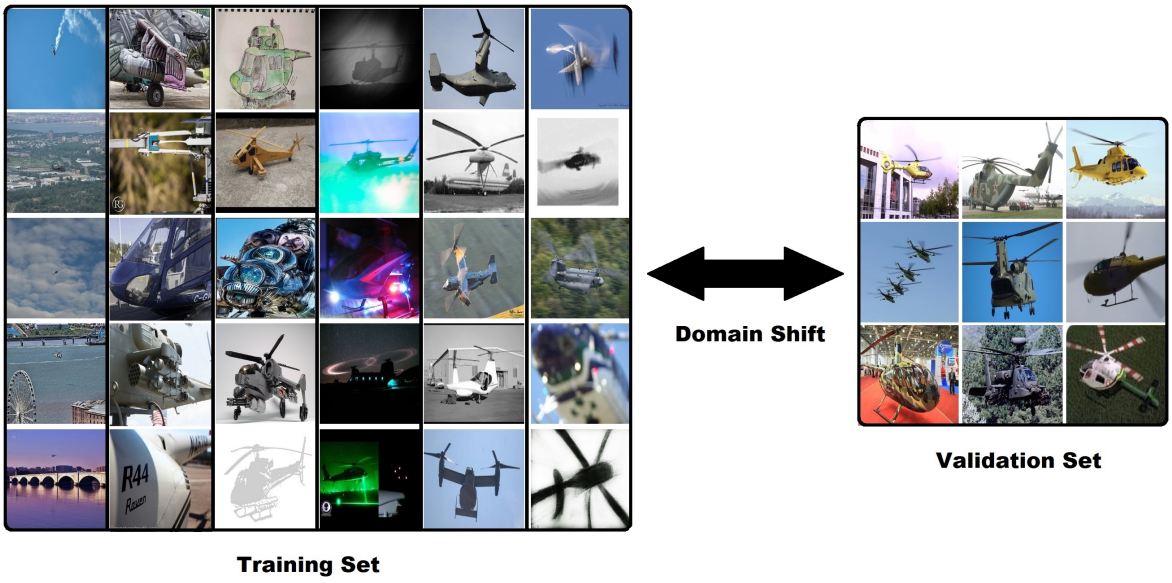

Figure 6.4: Examples of domain shift from the Helicopter class of the Aircraft-7 dataset.

The most common approach for dealing with domain shift is so-called domain adaption (DA) (see [44] for a comprehensive survey). These methods attempt to learn a latent representation that is useful for the classifier but also domain invariant. A fundamental assumption of DA, however, is that the training set represents a single homogeneous domain. As noted in [43] however, this assumption is almost never true for images datasets gathered from the web. Unless images are carefully selected with a clearly identiable domain in mind, weakly-supervised datasets of this kind are more realisti-

cally an agglomeration of several distinct domains. Because of this, we often encounter a situation whereby some images in the training set are representative of the target domain whilst others are not. It is unclear whether traditional DA methods are appropriate in this setting.

When we began evaluating our method for label cleaning on weakly-supervised web images, we soon discovered that it simultaneously tackles domain shift in a very natural way. We observed that the correctly labelled examples which Training-ValueNet wrongly identified as being mislabelled were typically those that were not representative of the target domain. Fig. 6.4 shows a selection of these training examples taken from the helicopter class of the Aircraft-7 dataset. From columns left to right, these images are highly zoomed out, zoomed in, artistic depictions, badly lit, unusual looking, and blurry. It is not surprising that these examples did not benefit performance when they are so different to the more standardised validation examples shown on the right of Fig. 6.4. Whilst removing these examples from the training set will not eliminate the domain shift entirely, it will at least get rid of the worst offending images which actually *hurt* performance. In this way there is a clear potential here for Training-ValueNet to tackle label noise and domain shift *simultaneously*.

A thorough investigation of the utility of our method for dealing with domain shift is left to further work. We do however present an initial demonstration here of the sensitivity of training value to domain shift. To do this we take the Celeb-A face facial recognition dataset [45] and repurpose it for the task of gender classification. We use the binary attributes provided for each image to then divide the images into five mutually exclusive sub-domains. These are; young with blonde hair, young with black hair, old with grey hair, old with black hair, and wearing a hat. Fig. 6.5 shows an image from each sub-domain and class. Each sub-domain is further divided into a training and validation subset.

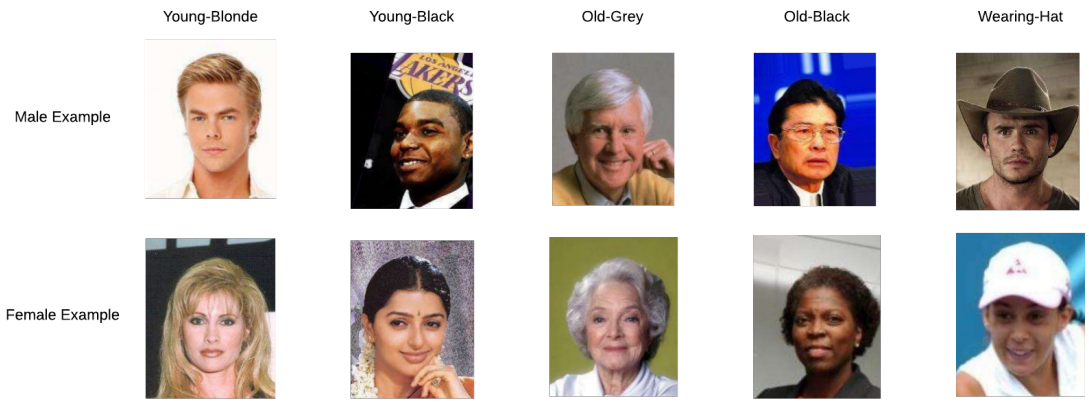

Figure 6.5: Examples of each sub-domain and class from the Celeb-A dataset.

We begin by computing the pairwise performance between all pairs of sub-domains, that is, the performance obtained if we train on sub-domain X and evaluate on sub-domain Y (for all pairs X,Y). These performances are listed in Table 6.1. We can clearly see that the domain shift between sub-domains causes a degradation in performance compared to when we train and test on images from the same domain.

In order to demonstrate the sensitivity of training value to domain shift, we re-combined the sub-domains to form a unified training set. We then selected just one of the sub-domains ('young blonde') to be our target domain (validation set). Under these conditions, we carry out our method and obtain predictions for the training value of each

Test on → / Train on ↓	Young Blonde Hair	Young Black Hair	Old Grey Hair	Old Black Hair	Wearing a Hat
Young Blonde Hair	93.4	88.1	67.0	80.5	68.9
Young Black Hair	91.3	95.2	69.7	81.5	66.6
Old Grey Hair	74.4	86.4	93.9	88.8	78.7
Old Black Hair	87.0	89.2	82.5	92.9	74.0
Wearing a Hat	75.8	87.5	90.3	89.4	84.5

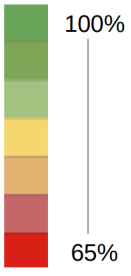

Table 6.1: Pairwise performance (accuracy (%)) obtained by training a classifier on one sub-domain and evaluating on another. Colours indicate the performance relative to the maximum achieved on that sub-domain.

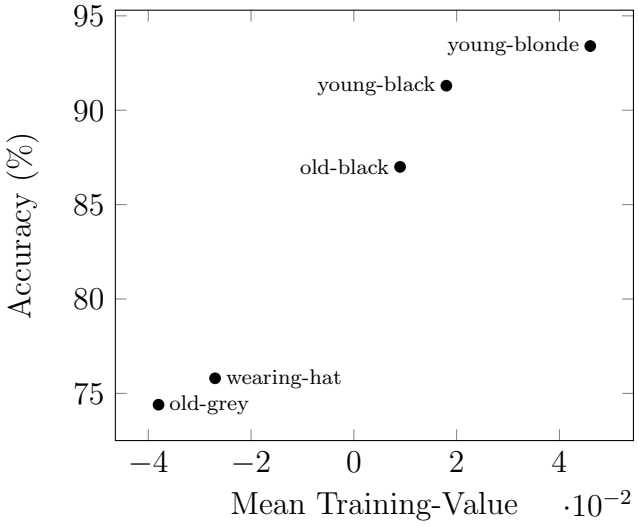

Figure 6.6: Plot showing mean training-value for examples in each sub-domain vs the accuracy achieved when training on that sub-domain and evaluating on the 'young-blonde' sub-domain.

training example. In Fig. 6.6 we plot the mean training value for images in each sub-domain against the pairwise performance that obtained when we trained on examples from that sub-domain and evaluated on 'young blonde'. What we find is a near-perfect correlation between the two demonstrating that the training value is sensitive to the severity of the domain shift between that image and the target domain.

Whilst a more rigorous investigation is still needed, this initial investigation does suggest that our method is capable of dealing with label noise and domain shift *simultaneously*. Whilst an abundance of work has been carried out on each of these issues, we are unaware of any existing method that tackles both simultaneously in this way.

References

[1] L. Smyth, D. Kangin, and N. Pugeault, "Training-valuenet: Data driven label noise cleaning on weakly-supervised web images," in *Proceedings of the Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics*, IEEE, 2019.

[2] R. Socher, J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 248–255, 2009.

[3] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop," *arXiv e-prints*, p. arXiv:1506.03365, June 2015.

[4] K.-h. Lee, X. He, L. Zhang, and L. Yang, "CleanNet: Transfer Learning for Scalable Image Classifier Training with Label Noise," *CVPR, 2016*, 2016.

[5] A. Drory, S. Avidan, and R. Giryes, "On the Resistance of Neural Nets to Label Noise," *CoRR*, pp. 1–19, 2018.

[6] A. J. Ratner, C. M. De Sa, S. Wu, D. Selsam, and C. Ré, "Data programming: Creating large training sets, quickly," in *Advances in neural information processing systems*, pp. 3567–3575, 2016.

[7] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid training data creation with weak supervision," *Proceedings of the VLDB Endowment*, vol. 11, no. 3, pp. 269–282, 2017.

[8] S. H. Bach, D. Rodriguez, Y. Liu, C. Luo, H. Shao, C. Xia, S. Sen, A. Ratner, B. Hancock, H. Alborzi, *et al.*, "Snorkel drybell: A case study in deploying weak supervision at industrial scale," in *Proceedings of the 2019 International Conference on Management of Data*, pp. 362–375, ACM, 2019.

[9] B. Frénay and M. Verleysen, "Classification in the presence of label noise: A survey," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 25, no. 5, pp. 845–869, 2014.

[10] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[11] C. E. Brodley and M. A. Friedl, "Identifying mislabeled training data," *Journal of artificial intelligence research*, vol. 11, pp. 131–167, 1999.

[12] G. L. Libralon, A. C. P. de Leon Ferreira, A. C. Lorena, *et al.*, "Pre-processing for noise detection in gene expression classification data," *Journal of the Brazilian Computer Society*, vol. 15, no. 1, pp. 3–11, 2009.

[13] D. Rolnick, A. Veit, S. Belongie, and N. Shavit, "Deep Learning is Robust to Massive Label Noise," *arXiv e-prints*, p. arXiv:1705.10694, May 2017.

[14] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," tech. rep., Citeseer, 2009.

[15] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[16] D. Flatow and D. Penner, "On the robustness of convnets to training on noisy labels," tech. rep., Stanford University, 2017.

[17] W. Li, L. Wang, W. Li, E. Agustsson, and L. Van Gool, "WebVision Database: Visual Learning and Understanding from Web Data," *arXiv e-prints*, p. arXiv:1708.02862, Aug. 2017.

[18] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 2691–2699, 2015.

[19] G. Patrini, A. Rozza, A. K. Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proc. - 30th IEEE*

Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-Janua, pp. 2233–2241, 2017.

[20] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*, pp. 413–422, IEEE, 2008.

[21] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *ACM sigmod record*, vol. 29, pp. 93–104, ACM, 2000.

[22] Y. Xia, X. Cao, F. Wen, G. Hua, and J. Sun, "Learning discriminative reconstructions for unsupervised outlier removal," in *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2015 Inter, pp. 1511–1519, 2015.

[23] J. S. Kim and C. Scott, "Robust kernel density estimation," in *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, pp. 3381–3384, 2008.

[24] E. Parzen, "On estimation of a probability density function and mode," *The annals of mathematical statistics*, vol. 33, no. 3, pp. 1065–1076, 1962.

[25] M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," *The Annals of Mathematical Statistics*, pp. 832–837, 1956.

[26] W. Liu, G. Hua, and J. R. Smith, "Unsupervised one-class learning for automatic outlier removal," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 3826–3833, 2014.

[27] L. Van Der Maaten and G. Hinton, "Visualizing Data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.

[28] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. 26th Annu. Int. Conf. Mach. Learn. - ICML '09*, vol. 2, (New York, New York, USA), pp. 1–8, ACM Press, 2009.

[29] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and

A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, (Stockholmsmässan, Stockholm Sweden), pp. 2304–2313, PMLR, 10–15 Jul 2018.

[30] S. Guo, W. Huang, H. Zhang, C. Zhuang, D. Dong, M. R. Scott, and D. Huang, "CurriculumNet: Weakly supervised learning from large-scale web images," in *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11214 LNCS, pp. 139–154, 2018.

[31] R. S. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting N," *J. Mach. Learn. Res.*, 2014.

[32] "Bulk Image Downloader." `https://bulkimagedownloader.com/`. Accessed: 2019-09-30.

[33] F. Chollet, "keras." `https://github.com/fchollet/keras`, 2015.

[34] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *CVPR, 2016*, dec 2016.

[36] Nesterov, "A method for solving the convex programming problem with convergence rate $O(1/k^2)$," *Dokl. Akad. Nauk SSSR*, 1983.

[37] Z. G. X. Zhu, "Learning from labeled and un- labeled data with label propagation," *C. CALD-02-107*, 2002.

[38] D. Y. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Scholkopf, "Learning with local and global consistency," *Adv. Neural Inf. Process. Syst. 16*, vol. 16, pp. 321–328, 2004.

[39] R. Sutton, "The Bitter Lesson." `http://www.incompleteideas.net/IncIdeas/BitterLesson.html`.

[40] V. Avramova, *Curriculum Learning with Deep Convolutional Neural Networks*. PhD thesis, KTH Royal Institute of Technology Stockholm, 2015.

[41] J. Krause, B. Sapp, A. Howard, H. Zhou, A. Toshev, T. Duerig, J. Philbin, and L. Fei-Fei, "The Unreasonable Effectiveness of Noisy Data for Fine-Grained Recognition," in *Lect. Notes Comput. Sci.*, 2016.

[42] E. H. Pooch, P. L. Ballester, and R. C. Barros, "Can we trust deep learning models diagnosis? the impact of domain shift in chest radiograph classification," *arXiv preprint arXiv:1909.01940*, 2019.

[43] L. T. Alessandro Bergamo, "Exploiting weakly-labeled Web images to improve object classification: a domain adaptation approach," in *NIPS*, pp. 1–9, 2010.

[44] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, "Visual domain adaptation: A survey of recent advances," *IEEE signal processing magazine*, vol. 32, no. 3, pp. 53–69, 2015.

[45] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.