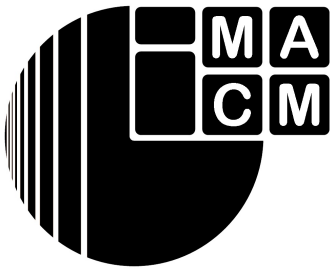

Bergische Universität Wuppertal

Fachbereich Mathematik und Naturwissenschaften

Institute of Mathematical Modelling, Analysis and Computational Mathematics (IMACM)

Preprint BUW-IMACM 18/06

M. Rottmann, K. Kahl and H. Gottschalk

Deep Bayesian Active Semi-Supervised Learning

June 12, 2018

http://www.math.uni-wuppertal.de

Deep Bayesian Active Semi-Supervised Learning

Matthias Rottmann*, Karsten Kahl* and Hanno Gottschalk*

Abstract

In many applications the process of generating label information is expensive and time consuming. We present a new method that combines active and semi-supervised deep learning to achieve high generalization performance from a deep convolutional neural network with as few known labels as possible. In a setting where a small amount of labeled data as well as a large amount of unlabeled data is available, our method first learns the labeled data set. This initialization is followed by an expectation maximization algorithm, where further training reduces classification entropy on the unlabeled data by targeting a low entropy fit which is consistent with the labeled data. In addition the algorithm asks at a specified frequency an oracle for labels of data with entropy above a certain entropy quantile. Using this active learning component we obtain an agile labeling process that achieves high accuracy, but requires only a small amount of known labels. For the MNIST dataset we report an error rate of 2.06% using only 300 labels and 1.06% for 1,000 labels. These results are obtained without employing any special network architecture or data augmentation.

1 Introduction

In recent years deep learning has shown great potential in solving classification and regression tasks of increasing complexity and difficulty. For academic purposes, several labeled data sets with associated tasks are available to support and facilitate research on machine learning. Though in many practical applications (e.g. in industry, medicine and microbiology) where raw data is available in abundance, labeled information is not readily available and the process of generating labels can be time consuming and expensive. Therefore, the development of methods that provide strong predictive models from as few labels as possible is a field of high interest.

The fields of active learning and semi-supervised learning address this issue and provide two approaches to obtain strong predictive models using only few labels, see Gal et al. 2016; Hu et al. 2017; Kingma et al. 2014a; Lee 2013; Pitelis et al. 2014; Rasmus et al. 2015; Rifai et al. 2011; Weston et al. 2012. They both assume a situation where the complete set of data is large, but labels are known only for a small fraction of it.

The field of semi-supervised learning has a long history. Already in Suddarth et al. 1990 unlabeled data had been injected into the training of neural networks in order to improve generalization performance. Most approaches rely on the Expectation Maximization (cf. Dempster et al. 1977, EM) technique which is a clustering algorithm. In the semi-supervised context, EM is used to assign unlabeled data to a finite number of clusters which are initially defined by the small set of labeled samples. That is, an initial model is trained and then, using the resulting model, labels are assigned to unlabeled data, which in turn are used to further train the model. The pseudo-label approach, introduced in Lee 2013, is such an EM technique. It uses the labels predicted by the neural network itself and can be viewed as well as an auxiliary loss in the training phase which is inserted to reduce classification entropy on the unlabeled data. It is well known that the EM strategy works well in presence of low density class separation. Thus it is unclear if and how this approach is able to adequately classify samples with high classification uncertainty. In case a quantitative measure of classification uncertainty can be defined, unlabeled data should only be used for training if their uncertainty is small. However, some samples typically retain high uncertainty in the semi-supervised training cycle. This is where active learning comes into play, in that it is most valuable to acquire ground truth labels for samples with high classification uncertainty and add those to training. In this way active and semi-supervised learning complement each other naturally.

Both learning approaches benefit from good uncertainty quantification mechanisms. With the advent of Monte-Carlo (MC) dropout Gal et al. 2016, we have an instrument at hand that makes it feasible to construct sensitive metrics to monitor classification uncertainty. Bayesian inference has been used in an active deep learning approach introduced

*Bergische Universiatät Wuppertal, Faculty of Mathematics and Natural Sciences, {rottmann,kkahl}@math.uni-wuppertal.de, hanno.gottschalk@uni-wuppertal.de

in Gal et al. 2017.

In recent years, there were also efforts on designing specialized network architectures that incorporate components like denoising auto-encoders, see Rasmus et al. 2015. Also deep generative models were used for semi-supervised learning, see Kingma et al. 2014b.

Combining the active learning and the semi-supervised learning track, a method for synthetic aperture radar image recognition has been published in Gao et al. 2017.

In this paper, we present a deep Bayesian Active Semi-Supervised learning (deepBASS) approach that is based on an EM deep learning approach for classification tasks paired with an active learning component and approximate Bayesian uncertainty. We first train a Convolutional Neural Network (CNN) on a small sample of labeled training data. Afterwards we employ the EM technique, i.e., we iteratively predict classes and assign these as pseudo-labels to the unlabeled data set. Then, we train one epoch on the pseudo-labeled data and the ground-truth-labeled data. While doing so, we make sure that the prediction accuracy on the ground truth remains high. During this process, the algorithm asks an oracle for additional label information where the neural network shows increased classification uncertainty, e.g., high classification entropy. For all predictions and uncertainty estimations we incorporate MC dropout inference.

The remainder of this work is structured as follows: In section 2 we classify our method with respect to existing approaches in the literature. Then we introduce our method in detail in section 3 including all necessary notations. Using a simple toy example in section 4 we motivate the combination of active and semi-supervised learning. Using the MNIST dataset, we compare two settings in section 5 where on one hand all unlabeled data is present in training from the beginning and where on the other hand unlabeled data is added only incrementally. Both settings are combined with two different label acquisition policies. Concluding the experiments we compare our method with other semi-supervised and active learning approaches.

2 Related Work

Our aim is to provide concepts of how to train models with high predictive power with as small labeling effort as possible. In this we combine components from active learning, semi-supervised learning and approximate Bayesian uncertainty quantification and construct a robust method that achieves high accuracy.

A related semi-supervised deep learning method including MC dropout inference has been published in Hyams et al. 2017 which incrementally assigns labels to data with highest predicted class probability above a chosen threshold and adds the respective data to the training data, but does not facilitate an active learning components.

On the other hand, an active deep learning approach making use of Bayesian uncertainty has been introduced in Gal et al. 2017. This work stresses the importance of approximate Bayesian model uncertainty in active learning and shows comparisons with semi-supervised methods. However, this approach does not make use of semi-supervised learning.

In Rasmus et al. 2015, a so-called ladder network with denoising components has been introduced, which achieved outstanding results for the MNIST dataset. This specialized network architecture, however, is not trivially generalizable to more complex tasks, like e.g. object classification/detection and semantic segmentation, where state-of-the-art networks are huge. Deep generative models have been employed successfully in semi-supervised learning, but suffer from scalability issues as well, see Kingma et al. 2014b. The aim of the presented approach is to show that a combination of active and semi-supervised learning techniques is able to achieve similar performance with a much simpler and scalable network architecture.

The active semi-supervised learning approach introduced in Gao et al. 2017, a method for synthetic aperture radar image recognition, accepts in every iteration a chosen number of pseudo-labels with highest confidence and asks an oracle for a chosen number of samples with lowest confidence. Confidence is measured in terms of highest classification probability, approximate Bayesian uncertainty is not employed in this approach. We observe in our tests that compared to the average classification entropy of the available initial ground-truth-labeled data, plenty of unlabeled data have classification entropy below this threshold. Therefore, in the beginning, many thousands of unlabeled samples can be automatically labeled and added to train-

ing while producing only a tiny fraction of false positives, i.e., incorrect labels. Furthermore, we also address the question whether it is necessary to pseudo-label and add unlabeled data incrementally or at once.

3 DeepBASS Learning

In order to introduce the deep Bayesian active semi-supervised learning approach we first review its basic components.

3.1 Expectation Maximization

The Expectation Maximization (Dempster et al. 1977, EM) algorithm is a widely used clustering approach. In the original unsupervised context this clustering algorithm is initialized on a model with a predefined number of classes and random parameters. When doing semi-supervised learning, this random initial model is replaced by a model that is trained on the scarce ground truth labels. The second phase is always unsupervised and applies the model to all unlabeled (later pseudo-labeled) data in order to cluster it. Here, the clustering metric is provided by the neural network itself via classification entropy. This in turn can be viewed as adding an additional term to the loss function. Afterwards the neural network is trained in a self-affirmation manner towards its own predictions, where ground truth labels, unlike for the original EM, are never reassigned by model predictions.

Let us introduce some notation. Let

$$(X, G) := \{(x_j, g_j) \,:\, j = 1, \dots, N\} \subseteq \mathbb{R}^n \times \mathbb{G} \quad (1)$$

denote the collection of input samples, where X denotes the data and G the set of all associated labels from a finite label space $\mathbb{G}$ containing C classes, which in the following are identified with numbers, i.e., $\mathbb{G} := \{1, \dots, C\}$. We denote the deterministic probability distribution on $\mathbb{G}$ by δ_g, i.e., $\delta_g(g) = 1$ and $\delta_g(c) = 0$, $g \neq c$. Further, let $X' = \{x'_j \,:\, j = 1, \dots, N'\}$ with $N' \gg N$ denote input samples where no labels are available, and let $f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{Y}$ denote the neural network function where p is the number of learnable parameters and $\mathbb{Y} = \{y \in [0,1]^C \subseteq \mathbb{R}^C \,:\, \sum_{c=1}^C y_c = 1\}$ the space of classification distributions. For an input $x \in \mathbb{R}^n$ and weights $w \in \mathbb{R}^p$ we denote the softmax output of the neural network by $\hat{y} = f(x, w)$.

The loss function in our approach is the negative maximum likelihood of the softmax classification rule

$$\mathcal{L}_1(g, \hat{y}) = -\sum_{c=1}^{C} \delta_g(c) \log(\hat{y}_c) = -\log(\hat{y}_g)\,. \quad (2)$$

Similarly, the normalized classification entropy $\mathcal{H}$ is defined by

$$\mathcal{H}(\hat{y}) = -\frac{1}{\log(C)} \sum_{c=1}^{C} \hat{y}_c \log(\hat{y}_c) \in [0,1]\,. \quad (3)$$

An auxiliary loss for the unlabeled data can be defined as follows. Let pseudo-labels be defined by

$$\psi(\hat{y}) := \arg\max_c \{\hat{y}_c \,:\, c = 1, \dots, C\}\,, \quad (4)$$

i.e., the index with highest classification probability. Then the auxiliary loss can be expressed using the loss function $\mathcal{L}_1$ from eq. (2) via

$$\mathcal{L}_1(\psi(\hat{y}), \hat{y}) = -\sum_{c=1}^{C} \delta_{\psi(\hat{y})}(c) \log(\hat{y}_c)\,. \quad (5)$$

This term reaches its minimum if $\hat{y} = \delta_{\psi(\hat{y})}$, and indeed we have $\mathcal{L}_1(\psi(\hat{y}), \hat{y}) = \mathcal{H}(\delta_{\psi(\hat{y})}) = 0$. Thus, minimizing eq. (3) is conceptually close to minimizing eq. (5). We can now define a combined loss by

$$\mathcal{L}(g, \hat{y}) = \begin{cases} \mathcal{L}_1(g, \hat{y}) & \text{for } x \in X \\ \mu \mathcal{L}_1(\psi(\hat{y}), \hat{y}) & \text{for } x \in X'\,. \end{cases} \quad (6)$$

We do not consider the entropy based equivalent of eq. (6), as the loss function $\mathcal{L}$ yields additional freedom in the definition of pseudo-labels $\psi(\hat{y})$ (cf. eq. (4)). The choice of the regularization parameter μ and other practical implementation details are discussed in sections 4 and 5. Having ingredients for the prescription of the EM algorithm with pseudo-labels, we can state a generic version of it in algorithm 3.1, which we specify further in the next paragraph.

3.2 Monte-Carlo Dropout Inference

Disregarding the nature of the given data and the prediction task, the practical performance of algorithm 3.1 strongly depends on three factors, the initial accuracy achieved in line 2, the pseudo-label quality which depends on lines 5 and 6,

Algorithm 3.1: Active EM with pseudo-labels

Data: (X, G) labeled data from eq. (1), X' unlabeled data
Result: weights w

1 let $(D, L) := (X, G)$, initialize weights w
2 train w on (D, L) minimizing $\mathcal{L}$
3 **repeat**
4 **forall** $x' \in X'$ **do**
5 infer y' using f, w and x'
6 **if** y' close enough to $\delta_{\psi(y')}$ **then**
7 $(D, L) \leftarrow (D, L) \cup \{(x', \psi(y'))\}$
8 **for** a chosen number of $x' \in X'$ with y' far away from $\delta_{\psi(y')}$
9 ask the oracle for the ground truth g'
10 $(X, G) \leftarrow (X, G) \cup \{(x', g')\}$
11 $x' \leftarrow X' \cup \{x'\}$
12 train w one epoch on (D, L) minimizing $\mathcal{L}$
13 **until** satisfied

cf. e.g. Hyams et al. 2017; Lee 2013, and the acquisition policy in line 8 for demanding additional ground truth, see Gal et al. 2017. In this paragraph we focus on the latter two aspects. It has been proposed in Hyams et al. 2017 to use MC dropout inference for generating pseudo-labels in a semi-supervised setting. In the active learning setting, MC dropout has been used in Gal et al. 2017 to evaluate the uncertainty of a prediction $f(x, w)$ and thus decide which samples to label next by help of an oracle. We combine both approaches as follows.

In line 5 we simply infer using MC dropout with a chosen number T' of forward passes to obtain the average probability outputs

$$\tilde{y} = \tilde{f}_{T'}(x, w) := \frac{1}{T'} \sum_{t=1}^{T'} f^{(t)}(x, w) \text{ for all } x \in X' , \tag{7}$$

where $f^{(t)}$ denotes f with dropout. Note, that $f^{(t)}$ is not deterministic when including dropout, i.e., $f^{(t)}$ is not uniquely defined for a chosen t.

In order to perform line 6 we define a metric that tells us how close $\tilde{y}$ is to $\delta_{\psi(\tilde{y})}$. Clearly, many different metrics could be considered. Our metric of choice is the classification entropy $\mathcal{H}(\tilde{y})$ from eq. (3). For the threshold estimation, we apply MC dropout inference with a chosen number of forward passes T to all available ground truth labeled data X (including data where ground truth is obtained from the oracle during training) and calculate the average classification entropy

$$\theta := \frac{1}{|X|} \sum_{x \in X} \mathcal{H}(\tilde{f}_T(x, w)) . \tag{8}$$

We choose a threshold θ and add in every iteration of algorithm 3.1 all samples with entropy below threshold, i.e., $\mathcal{H}(\tilde{f}_{T'}(x', w)) < \theta$. That is,

$$(D, L) \leftarrow (D, L) \cup \{(x', \psi(\tilde{f}_{T'}(x', w)))\} \text{ for } x' \in X' . \tag{9}$$

For the active learning part in line 8 we use the entropy of averaged classification results under MC dropout, i.e., $\mathcal{H}(\tilde{f}_{T'}(x', w))$ for all $x' \in X'$. For a chosen number of samples $x' \in X'$ with highest entropy we ask the oracle for the ground truth g' and add the labeled data (x', g') to (X, G) while removing x' from X'. Other approaches for acquiring labels are proposed in Gal et al. 2017, where it has been shown that the classification entropy under MC dropout is one of the best choices among the considered acquisition functions.

Parameter values for T and T' are stated in the experiments in section 5.

4 An Illustrative Example

In this section we show some experiments with algorithm 3.1 for illustration and motivation. Our simple problem consists of 2 non-convex distributions, from which we draw 500 samples each. The distributions are generated as follows: Let $\mathcal{N}(m, \sigma^2)$ be a Gaussian normal distribution with mean m and variance σ^2. We draw radius and angle from

$$r \sim \mathcal{N}(1, (1/4)^2) \quad \text{and} \quad \phi \sim \mathcal{N}(1/2, (1/3)^2) . \tag{10}$$

Samples from the red distribution are constructed as

$$red = (1/3, -1/10) + r \cdot (\cos(\phi), \sin(\phi)) \tag{11}$$

and samples from the blue one are constructed as

$$blue = (-1/3, 1/10) + r \cdot (\cos(\phi), -\sin(\phi)) , \tag{12}$$

together they form a Yin-and-Yang type of picture, see figs. 1 and 2.

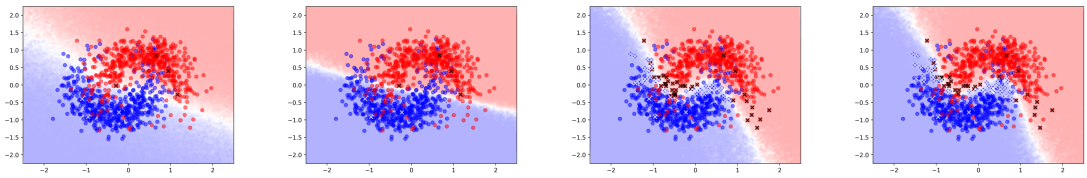

Figure 1: Experiments for algorithm 3.1: *(left)*: training with 8 labels, 4 per class, *(middle left)*: 8 labels + pseudo-labels for the rest, *(middle right)*: active learning with initial 8 labels + 72 over time, *(right)*: active semi-supervised learning with initial 8 labels, pseudo-labels for the rest and 72 labels over time.

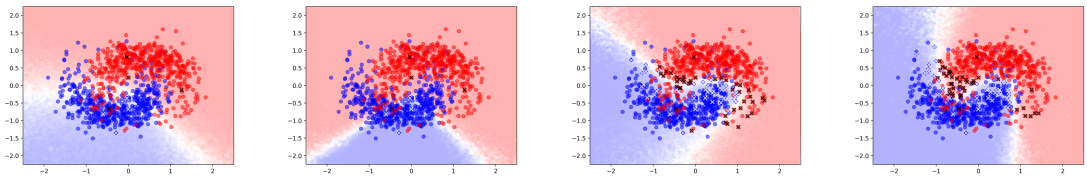

Figure 2: Experiments for algorithm 3.1: same tests as in fig. 1, but a different choice of initial labels.

Network Architecture and Parameters There are two competing objectives at work when considering a suitable network architecture for deepBASS. On one hand, we have to employ strong regularization in order to avoid overfitting when learning the small set of initially labeled samples. On the other hand, the initial model has to re-adjust to new ground truth obtained from the oracle during the EM iteration, which requires flexibility.

Thus in addition to strong regularization our model needs to be equipped with enough learnable parameters. Hence, for all tests shown in this section we use a fully connected neural network with 2 input neurons, 3 hidden layers with 50 neurons each and 2 output neurons. After each hidden layer we use the LeakyReLU activation function, i.e.,

$$LeakyReLU(x) = \begin{cases} x & \text{for } x > 0 \\ 0.1x & \text{else,} \end{cases} \tag{13}$$

followed by dropout Srivastava et al. 2014 with 33% dropout rate. All models are trained and evaluated using Keras Chollet et al. 2015 with the Tensorflow backend Abadi et al. 2015. All layers are L^2 regularized with regularization parameter $\lambda = 10^{-3}$. To fit the weights we use Adam Kingma et al. 2014a with default parameters. The mini-batch size is 256.

For MC dropout inference on unlabeled and pseudo-labeled data X' we use $T' = 10$ forward passes with dropout, and on the data X with known ground truth we perform $T = 100$ forward passes. In every iteration of algorithm 3.1 unlabeled data from X' that is not in D, yet, is added if its classification entropy is below the threshold θ, cf. eq. (8).

Intentional Overfitting of Ground Truth The choice of the regularization parameter μ in the loss function in eq. (6) plays an important role. Chosen too small, the use of unlabeled data will barely have any effect, chosen too large, the classification accuracy (rate of correctly predicted classes) on the ground truth labeled data will decrease while the iteration in algorithm 3.1 proceeds. Thus we choose to rather overfit the scarce ground truth data in order to allow algorithm 3.1 to find a clustering of the unlabeled data X' consistent with the labeled data X.

In our Keras implementation, we implemented this balance by "upsampling" X in D, i.e., a sample from X, with known ground truth, is contained 20 times in D while a sample from X' will be contained (at most) once in D. E.g. for 8 labeled samples and 992 unlabeled ones, we can expect about 14% of the data in a mini-batch to be labeled with ground truth. In our tests we observe that this leads to

overfitting the ground truth, however it prevents the neural network from forgetting the crucial information.

Experiments. In all four panels in figs. 1 and 2 we start with 4 labels per class, the depicted data points represent all data available for training, i.e., $X \cup X'$, the data points in X, where ground truth is available, are crossed out. The background color gradients depict where the neural network predicts the red or the blue class, respectively. The classification boundary is white and represents the region where the classification uncertainty is high. Figures 1 and 2 only differ in the choice of the 8 labels. While the choice in fig. 1 is easy to handle, the choice in fig. 2 represents a rather ill-posed case.

For the left panel in figs. 1 and 2 we train only on the available ground truth until the classification accuracy on the training set stagnates. For each figure, all other panels share the left panel as initial model.

In the middle left panel we continue with semi-supervised training, not adding any further ground truth labels. The resulting model in both cases is more sure about its decision, this is indicated by background colors that are more saturated. However in fig. 2 the classification boundary got worse compared to the left panel. Both figures also show that it can happen that semi-supervised learning does not perform well, especially when a low density distribution at the class boundaries is not present.

In the center right panel we use active learning. Every second iteration, we demand two ground truth labels. We keep iterating until 80 samples are labeled, i.e., 72 iterations. In the right hand panel we use both, active and semi-supervised learning. While in fig. 1 active learning performs just as well as active semi-supervised learning, the latter clearly is superior in fig. 2. The interpretation of these results is that the semi-supervised learning component has a regularizing effect on active learning.

Method	panel in figs. 1 and 2	val. acc.
initial model	(left)	83.27%
semi-supervised	(middle left)	84.33%
active	(middle right)	90.19%
active semi-supervised	(right)	90.33%
80 labels		88.65%
1,000 labels		91.37%

Table 1: Average classification accuracies for the tests performed in figs. 1 and 2.

Summarizing these tests, we state classification accuracies averaged over 10 runs for all four tests in table 1 and complement these with results for purely supervised learning using 80 and 1,000 labels, respectively. The corresponding models are trained until validation accuracy stagnates.

5 Experiments with MNIST

For our experiments, we use the MNIST dataset LeCun et al. 1998 of handwritten digits, given as tiny 28×28 gray scale images with the pre-defined data split of 60,000 training and validation 10,000 images. Again, all models are trained and evaluated using Keras with Tensorflow backend. For the CNN architecture we use a generic building block containing the following components:

- convolutional layer with 16 filters of size 3×3,
- LeakyReLU activation function, eq. (13),
- dropout with 33% dropout rate.

We stack four of these building blocks, after the second and the fourth layer we apply 2×2 max pooling. This results in a $7 \times 7 \times 16$ tensor, followed by a dense layer with 10 outputs and a final softmax activation. The resulting network is equipped with 14,970 learnable parameters. All convolutional layers are trained with L^2 regularization and a regularization parameter $\lambda = 10^{-3}$. We again use Adam with default parameters for training.

Parameters. Throughout our experiments we use the following parameters. For MC dropout inference on unlabeled and pseudo-labeled data X' we use $T' = 10$ forward passes of dropout, and on data X with known ground truth we perform $T = 100$ forward passes. In each test we perform 200 iterations of algorithm 3.1 and we perform each test 10 times while re-sampling the initial 100 samples. The presented results are averages of these 10 runs, the ground truth up-sampling factor is 20.

The initial neural network is trained on a balanced data set containing the same number of samples for each class. By default we start with 100 labeled samples, i.e., 10 per class. By presenting the ground truth labeled data 2,000 times we obtain a training accuracy of roughly a 99–100%. During

the 200 iterations in algorithm 3.1 we track the performance by monitoring validation accuracy. When we perform active learning, we acquire 10 labels once every 10 iterations. Note, that the added labels are not necessarily class-balanced.

Experiments with Entropy Thresholding and Label Acquisition Policy. Figure 3 shows the behavior of algorithm 3.1 over the course of 200 iterations, averaged over 10 runs. In the left panel we study the influence of the threshold θ, i.e., we compare the case where all unlabeled data is used in training right from the start ($\theta = 1$, short hand: *all data*) with the strategy where pseudo-labeled data is added step-wise according to the threshold θ from eq. (8) (short hand: *step-wise*). This comparison is made while using two different label acquisition policies. On one hand we ask for labels of unlabeled samples $x' \in X'$ with maximum entropy as explained in section 3.2 (short hand: *max. entropy*), on the other hand we try a slightly more careful policy where we only demand labels for samples $x' \in X'$ randomly drawn from all data in X' with entropy above average (short hand: *above avg.*), i.e.,

$$\mathcal{H}(\tilde{f}_{T'}(x', w)) > \frac{1}{|X| + |X'|} \sum_{x \in X \cup X'} \mathcal{H}(\tilde{f}_{T'}(x, w))\,. \tag{14}$$

The latter strategy is motivated by the fact that it might happen that exclusively acquiring data with high classification entropy could result in overfitting (wiggly decision boundaries) of data from non-separable distributions, consequently slowing down the convergence of algorithm 3.1.

In our tests with 100 initial ground truth labels, all four approaches share the same initial models in each run, the average validation accuracy is 82.05% after training with 100 labeled samples evenly distributed over all classes. The left panel of fig. 3 shows that the *all data + max. entropy* approach is slightly superior to the same acquisition policy where unlabeled data is added *step-wise*. We believe that the reason for this is in part the well-behaved nature of the MNIST dataset. The *above avg.* acquisition policy is slightly inferior, but works better when all available unlabeled data is used from the beginning. Summarizing, *all data* combined with *max. entropy* acquisition policy reaches 97.92% accuracy on average. This result is the average of all 10 runs stopping after 200 iterations and measuring the accuracy of the model after the last iteration. The best result in a single run is 98.33%. In contrast to this, the most careful approach, *step-wise + above avg.*, ends up with 97.43% which is still good.

Threshold	policy	ground truth	val. acc.(stddev.) %
–	–	100	82.03(±1.95)
–	–	300	91.91(±0.75)
–	–	600	94.92(±0.32)
–	–	1,000	96.24(±0.39)
–	–	3,000	97.89(±0.25)
step-wise	max. entropy	300	97.67(±0.34)
step-wise	above average	300	97.43(±0.19)
all data	max. entropy	300	97.92(±0.19)
all data	above average	300	97.65(±0.26)
all data	–	100	96.08(±1.49)
all data	–	300	97.10(±0.40)
all data	–	600	97.39(±0.35)
all data	–	1,000	97.64(±0.23)
all data	–	3,000	98.14(±0.26)
–	–	60,000	99.09(±0.07)

Table 2: Summary of average classification accuracies for the tests performed in fig. 3. For all tests with label acquisition policy, 100 ground truth labels are used initially, 200 are added over time.

In order to understand how much we benefit from combining active learning and semi-supervised learning, we compare the best approach from the left panel of fig. 3 with algorithm 3.1, but without the active learning component. That is, we use 100 and 300 labels from the beginning and perform semi-supervised learning without adding any further ground truth labels. The results are depicted in the right hand panel of fig. 3 and they show that active learning is indeed beneficial when using algorithm 3.1 for semi-supervised learning. The pure semi-supervised approach with 300 labels ends up with an average accuracy of 97.10% which is 0.82% less than for active semi-supervised learning. Note, that we achieve 96.08% with semi-supervised learning and 100 labels. All results from this section are summarized in table 2, complemented with result for supervised and semi-supervised learning with different numbers of labels. Compared to the *all data + max. entropy* approach, pure supervised learning with a random sample of labeled data requires about 10 times as many labels. The samples standard deviation reveals that our approach is robust under data resampling.

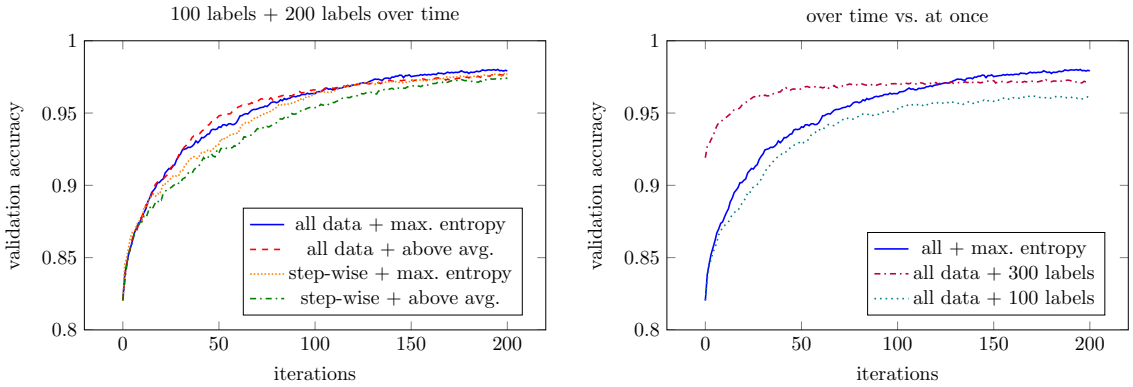

Figure 3: Experiments for algorithm 3.1 with two different thresholds for adding data, each with two different label acquisition policies.

Data Augmentation. Except for this paragraph, all tests in this work are performed without data augmentation. However in a practical setting it might make sense to use data augmentation as well. For the MNIST dataset, when using data augmentation on the ground truth labeled data with slight rotations of less than 10 degrees and slight image scaling of up to 5% in height and width, we observe that 50 labels are enough to achieve competitive initial validation accuracies of around 85%.

Comparison with Other Methods. In this section, we provide an overview of methods for semi-supervised deep learning and active deep learning where tests with the MNIST dataset have been performed. Most of the referred works provide numbers for 1,000 labels, results for 300 labels are scarce. We compare these results with the *all data + max. entropy* Deep Bayesian Active Semi-Supervised learning approach. For comparison we run our method 10 times until 1,000 samples are ground truth labeled and average over all validation accuracies, achieving 98.94% validation accuracy. A comprehensive comparison is stated in table 3. Clearly our approach, using 1,000 labeled samples is competitive at the upper end of the spectrum of reported results. Though one of the main advantages of it, as reported in the previous sections, is its ability to yield high accuracies even with as few as only 300 labeled samples. Note, that the full ladder net model from Rasmus et al. 2015 is a very sophisticated model incorporating denoising auto-encoder structures which might lack scalability and portability.

The semi-supervised part of our method with only 100 labels reaches a validation accuracy of 96.08%, a similar approach without MC dropout, see Lee 2013, only achieved 89.51%. We observed similar results in our tests without MC dropout inference which reveals its impact.

Method	test error
Semi-Supervised:	
Weston et al. 2012: Semi-Supervised Embedding	5.73%
Weston et al. 2012: Transductive SVM	5.38%
Pitelis et al. 2014: AtlasRBF	3.68%
Rifai et al. 2011: Manifold Tangent Classifier	3.64%
Lee 2013: Pseudo-label	3.46%
Hyams et al. 2017: Self training + Dyn. conf.	3.42%
Kingma et al. 2014b: Deep Generative Models	2.40%
Rasmus et al. 2015: Ladder, Γ-model	1.53%
Hu et al. 2017: Virtual Adversarial	1.32%
Rasmus et al. 2015: Ladder, full	0.84%
Active:	
Gal et al. 2017: Bald	1.80%
Gal et al. 2017: Max Entropy	1.74%
Gal et al. 2017: Var Ratios	1.64%
Active + Semi-Supervised:	
DeepBASS (*all data + max. entropy*):	1.06%

Table 3: Comparison with other approaches for a 1,000 labels. We term our method

6 Conclusion & Outlook

We have introduced a general active semi-supervised deep learning method with a wide field of possible applications that shows great performance in first results for the MNIST dataset. While

we use only simple classification entropy based uncertainty quantification, the presence of approximate Bayesian inference as well as the combination of semi-supervised learning and active learning constitute to the strength of our method as it outperforms state-of-the-art general approaches which do not use advanced network architectures.

If validation data is available, our approach can be further tuned with respect to thresholding and acquisition policy. This fact implies, that additional meta-learning extensions could be developed. A minor concern might be, that data which is added in the active part of the approach is prone to overfitting. A clean restart with the final data splitting and further tuning could additionally improve the performance of our method.

We plan to produce results for this approach in different applications and provide our source code on GitHub, cf. https://github.com/mrottmann/DeepBASS.

Acknowledgements. We would like to thank Fabian Hüger and Peter Schlicht from Volkswagen Group Research for discussion and remarks on this work.

References

Abadi, M., A. Agarwal, P. Barham, et al. (2015). *TensorFlow: large-scale machine learning on heterogeneous systems*. Software available from tensorflow.org.

Chollet, F. et al. (2015). *Keras*. https://github.com/fchollet/keras.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). "Maximum likelihood from incomplete data via the em algorithm". *Journal of the royal statistical society, series b* 39.1, pp. 1–38.

Gal, Y. and Z. Ghahramani (2016). "Dropout as a bayesian approximation: representing model uncertainty in deep learning ". *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. New York, NY, USA: JMLR.org, pp. 1050–1059.

Gal, Y., R. Islam, and Z. Ghahramani (2017). "Deep bayesian active learning with image data". *Corr* abs/1703.02910.

Gao, F. et al. (2017). "A novel active semisupervised convolutional neural network algorithm for SAR image recognition". *Comp. int. and neurosc.* 2017, 3105053:1–3105053:8.

Hu, W. et al. (2017). "Learning discrete representations via information maximizing self-augmented training ". *ICML*. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 1558–1567.

Hyams, G., D. Greenfeld, and D. Bank (2017). "Improved training for self-training". *Corr* abs/1710.00209.

Kingma, D. P. and J. Ba (2014a). "Adam: A method for stochastic optimization". *Corr* abs/1412.6980.

Kingma, D. P. et al. (2014b). "Semi-supervised learning with deep generative models". *Corr* abs/1406.5298.

LeCun, Y. et al. (1998). "Gradient-based learning applied to document recognition". *Proceedings of the ieee* 86.11, pp. 2278–2324.

Lee, D.-h. (2013). *Pseudo-label: the simple and efficient semi-supervised learning method for deep neural networks*.

Pitelis, N., C. Russell, and L. Agapito (2014). "Semi-supervised learning using an unsupervised atlas ". *Machine Learning and Knowledge Discovery in Databases*. Ed. by T. Calders et al. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 565–580.

Rasmus, A. et al. (2015). "Semi-supervised learning with ladder network". *Corr* abs/1507.02672.

Rifai, S. et al. (2011). "The manifold tangent classifier ". In: *Advances in Neural Information Processing Systems 24*. Ed. by J. Shawe-Taylor et al. Curran Associates, Inc., pp. 2294–2302.

Srivastava, N. et al. (2014). "Dropout: a simple way to prevent neural networks from overfitting". *Journal of machine learning research* 15, pp. 1929–1958.

Suddarth, S. C. and Y. L. Kergosien (1990). "Rule-injection hints as a means of improving network performance and learning time ". *Neural Networks*. Ed. by L. B. Almeida and C. J. Wellekens. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 120–129.

Weston, J. et al. (2012). "Deep learning via semi-supervised embedding ". In: *Neural Networks: Tricks of the Trade: Second Edition*. Ed. by G. Montavon, G. B. Orr, and K.-R. Müller. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 639–655.