INTRODUCING ACTIVE LEARNING FOR CNN UNDER THE LIGHT OF VARIATIONAL INFERENCE

Melanie Ducoffe & Frederic Precioso
Université Côte dAzur
CNRS, I3S, France
{ducoffe, precioso}@i3s.unice.fr

ABSTRACT

One main concern of the deep learning community is to increase the capacity of representation of deep networks by increasing their depth. This requires to scale up the size of the training database accordingly. Indeed a major intuition lies in the fact that the depth of the network and the size of the training set are strongly correlated. However recent works tend to show that deep learning may be handled with smaller dataset as long as the training samples are carefully selected (let us mention for instance *curriculum learning*). In this context we introduce a scalable and efficient active learning method that can be applied to most neural networks, especially Convolutional Neural Networks (CNN). To the best of our knowledge, this paper is the first of its kind to design an active learning selection scheme based on a variational inference for neural networks. We also deduced a formulation of the posterior and prior distributions of the weights using statistical knowledge on the Maximum Likelihood Estimator.

We describe our strategy to come up with our active learning criterion. We assess its consistency by checking the accuracy obtained by successive active learning steps on two benchmark datasets MNIST and USPS. We also demonstrate its scalability towards increasing training set size.

1 INTRODUCTION

A daunting challenge in many contexts is to gather annotated data. This can be a long and tedious process, which often slows down the development of a framework and may jeopardize its economic prospects.

We refer to active learning Cohn (1994) as the field of machine learning which targets building iteratively the annotated training set with the help of an oracle.

In this setting and in a context of pool-based active learning[1], a model is trained on a small amount of data (i.e. the initial training set) and a scoring function discriminates samples which should be labeled by the oracle from the ones which do not hold new information for the model. The queried samples are then submitted to an oracle (which can be another decision algorithm for instance in co-training context, or a human expert in interactive learning context) to be labeled. They are then added to the current training set. Finally the model is retrained from scratch. This process is repeated recursively to grow the training set.

Although active learning and deep learning represent two important pillars of machine learning, they have mainly coexisted into independent stream of works owing to the complexity of combining them. The main issues are the scalability and the adaptability of common active learning schemes when considering architectures with a huge number of parameters such as deep networks. Another issue lies in the overall number of training iterations since training a deep architecture remains a computationally expensive process despite all the optimizations through GPU processing. This specificity has prevented deep learning from being prevalent within active learning. Indeed seminal active learning frameworks Cohn (1994) have mainly focused on adding one sample at-a-time. When it comes to selecting a batch of queries, the most intuitive solution is to select top scoring samples.

[1]Other settings exist but are not considered here for the sake of clarity, we refer the reader to Settles (2012)

Such a solution is immediate in the process but fails to model the correlations between samples. Labeling one sample at-a-time may therefore lead to the labeling of another sample totally useless.

In our work, batches of actively selected samples are added at each training iteration. We propose a batch active learning framework designed for deep architectures, especially Deep Convolutional Neural Networks (CNN).

Batch active learning is highly suitable for deep networks which are trained on minibatches of data at each iterations. Indeed training with minibatches help the training of deep networks, and we empirically noticed that the size of the minibatch is a major hyperparameter. Thus it makes sense to query a batch of unlabelled data whose size would be proportionnal to the size of a minibatch. In our work, batches have the same size as the minibatches but they could be decorrelated by considering *importance sampling techniques.*

Our model focuses on log loss which is involved in the training process of most neural networks. To achieve the required scalability of active learning for deep architectures, we step away from traditional active learning methods and focus our attention on a more general setting: Maximum Likelihood Estimation (MLE) and Bayesian inference. Provided certain assumptions, our active selection relies on a criterion which is based on Fisher information and is obtained from the minimization of a stochastic method for variational inference. Our active selection relies on the Fisher matrices on the unlabeled data and on the data selected by the active learning step. An approximation of Fisher information, based on a diagonal Kronecker-block decomposition makes our criterion computationally affordable for an active greedy selection scheme.

Variational methods have been previously explored like in Graves (2011) as a tractable approximation to Bayesian inference for Artificial Neural Networks (ANNs). One advantage of such a representation is that it leads to a two-term active learning criterion: one related to the prediction accuracy on the observed data and the second term expressing the model complexity. Such a two-fold criterion is, to the best of our knowledge, the first of the kind scalable for active learning. Such an expression may help both to analyze and to optimize ANNs, not only in an active learning framework but also for curriculum learning.

We dedicate section *Related works* to the presentation of active learning literature. Section *Covering* presents the theoretical aspects of our active learning framework, while section *Active learning as a greedy selection scheme* details our greedy algorithm. We then assess our active learning method through experiments on MNIST and USPS benchmarks. We discuss possible improvements of our approach and connections with previous MLE-based active learning methods before concluding.

2 RELATED WORKS

Active learning is a framework to automatize the selection of instances to be labeled in a learning process. Active learning offers a variety of strategies where the learner actively selects which samples seem “optimal” to annotate, so as to reduce the size of the labeled training set required to achieve equivalent performance.

We consider the context of pool-based active learning where the learner selects its queries among a given unlabeled data set. For other variants (*query synthesis, (stream-based) selective sampling*) we refer the reader to Settles (2012).

2.1 POOL-BASED ACTIVE LEARNING

When it comes to pool-based active learning, the most intuitive approaches focus on minimizing some error on the target classifier. *Uncertainty sampling* minimizes the training error by querying unlabeled data on which the current classifier (i.e. from previous training iteration) is assigning labels with the weakest confidence. This method, *uncertainty sampling*, while being the least computational consuming among all active learning techniques has the main drawback of ignoring much of the output distribution classes, and is prone to querying outliers. Thanks to its low cost and easy setup, uncertainty has been adapted to deep architectures for sentiment classification Zhou et al. (2010). However, deep architectures are subject to adversarial examples, a type of noise we suspect uncertainty selection to be highly sensitive to Szegedy et al. (2014); Goodfellow et al. (2015). Other strategies (*expected error reduction*, *expected output variance reduction*) directly minimize the error on a validation set after querying a new unlabeled sample. However they are computationally expensive especially when considering neural networks.

Traditional active learning techniques handle selection of one sample at-a-time. One of the main drawbacks of the aforementioned active learning techniques is that is does not pay attention to the information held in unlabeled data besides considering it as potential queries. Hence once the strategy for selecting samples to be labeled and added to the training set is defined, the question on the impact of the possible correlation between successive selected samples remains.

To that end, one recent class of methods deals with the selection of a batch of samples during the active process, *batch mode active learning*. Batch mode active learning selects a set of most informative unlabeled sample instead of a unique sample. Such a strategy is highly suitable when retraining the model is not immediate and require to restart the training from scratch at each iteration as it is the case for neural networks. A simple strategy (*whose has also been used for previous deep active learning strategy Zhou et al. (2010)*) is to select a batch of top scoring instances. However that strategy fails to consider the correlation among pairs of samples. The redundancy between the so-selected samples may therefore hinder the learning process.

In the context of a deep learning scenario, if several elements related to the same direction of the gradient are in the same minibatch, the gradient descent in the next learning step may lead at once too close to a local minimum, diverting the process away from the global minimum.

While one sample at-a-time can prevent from being misled that way, it gets prohibitive when considering big data because the number of iterations is equal to the number of learning samples, unlike the batch-based strategies.

Recently some solutions have been proposed for choosing an appropriate subset of samples so as to minimize any significant loss in performance. Those methods consider the minimization of the Kullback-Leibler (KL) divergence between the resampled distribution of any possible subset selected for the active process and the whole unlabeled data set distribution. A lower bound of the negative of this KL divergence is then defined and rewritten as a submodular function. The minimization of the initial KL divergence becomes then a problem of submodular maximization Hoi et al. (2006). In Wei et al. (2015), Wei et al have designed several submodular functions so as to answer at best the need of specific classifiers (Naive Bayes Classifiers, Logistic Regression Classifier, Nearest Neighbor Classifier). However, their approach is hardly scalable to handle all the information from non-shallow classifiers such as deep networks.

Another solution to minimize the correlation among a set of queries is to perform bayesian inference on the weights of a neural network. In a bayesian context, a neural network is considered as a parametric model which assigns a conditional probability on the observed labeled data $\mathcal{A}$ given a set of weights $\mathbf{w}$. The weights follow some prior distribution $P(\alpha)$ depending on the parameter α and the posterior distribution of the weights $P(\mathbf{w} \mid \mathcal{A}, \alpha)$ is deduced. The goal is thus to maximize the posterior probability of the weights on the observed data $\mathcal{A}$. Indeed bayesian inference expresses the uncertainty of the weights which consequently leads to a relevant exploration of the underlying distribution of the input data $\mathbf{X}$. When it comes to active learning, the learner needs not only to estimate the posterior given the observed data $\mathcal{A}$ but also to consider the impact of new data on that posterior Golovin et al. (2010). In our context, bayesian inference is intractable, partially due to the high number of weights involved in a deep network. To solve this issue, Graves (2011) introduced a variational approximation to perform bayesian inference on neural networks. Specifically he approximates the posterior distribution $P(\mathbf{w} \mid \mathcal{A}, \alpha)$ with a tractable distribution of his choice $\mathcal{Q}(\mathbf{w} \mid \beta)$ depending on a new parameter β. The quality of the approximation $\mathcal{Q}(\mathbf{w} \mid \beta)$ compared to the true posterior $P(\mathbf{w} \mid \mathcal{A}, \alpha)$ is measured by the *variational free energy* $\mathcal{F}$ with respect to the parameters α and β. $\mathcal{F}$ has no upper bound but gets closer to zero as both distributions become more and more similar.

$$\mathcal{F}(\alpha, \beta) = -\mathbb{E}_{\mathbf{w}\sim\mathcal{Q}(\beta)}\left(ln\left(\frac{P(\mathcal{A} \mid \mathbf{w})P(\mathbf{w} \mid \alpha)}{\mathcal{Q}(\mathbf{w} \mid \beta)} \right)\right) \tag{1}$$

Finally in Graves (2011), $\mathcal{F}$ is then expressed as a minimum description loss function on α and β:

$$\mathcal{F}(\alpha, \beta) = \mathbb{E}_{\mathbf{w}\sim\mathcal{Q}(\beta)}(\mathcal{L}(\mathcal{A}; \mathbf{w})) + KL(\mathcal{Q}(\mathbf{w}) \mid\mid \mathcal{P}(\alpha)) \tag{2}$$

where KL is the Kullback Leibler divergence between $\mathcal{Q}(\beta)$ and $\mathcal{P}(\alpha)$.

Under certain assumptions on the family distribution for the posterior and prior of the weights (diagonal gaussian ...), Graves proposed a backpropagation compatible algorithm to train an ensemble of networks, whose weights are sampled from a shared probability distribution.

The primary purpose of the variational free energy is to propose a new training objectif for neural network by learning α and β. In an active learning context, the main drawback of variational free energy based method is that it requires to update by backpropagation the parameters for each unlabeled data submitted as a query. However we know from statistical assumption on the maximum likelihood the posterior and prior distribution of trained weights given the current labeled training set: if and only if we consider trained networks, we know how to build α and β in a unique iteration without backpropagation. This knowledge helps us to extend Graves first objectives to the use of variational free energy to measure how new observations affect the posterior.

3 COVERING

3.1 Variational Inference on Neural Network with Gaussian Posterior weight distribution

As done for the majority of neural networks, we measure the error of the weights $\mathbf{w}$ by the negative log likelihood on an observed set of annotated data $\mathcal{A}$:

$$\mathcal{L}(\mathcal{A};\mathbf{w}) = -\sum_{(x,y)\in\mathcal{A}} ln(P(y \mid x, \mathbf{w})) \tag{3}$$

We consider the Maximum Likelihood Estimator (MLE) $\mathcal{W}$ as the value which makes the data observed $\mathcal{A}$ as likely as possible for a fixed architecture. Note than even for a fixed $\mathcal{A}$ in the case of neural network, $\mathcal{W}$ may not be unique.

$$\mathcal{W} = argmin_{\mathbf{w}} \mathcal{L}(\mathcal{A};\mathbf{w}) \tag{4}$$

When assuming that an arbitrary parameter $\mathcal{W}^*$ is governing the data generation process, we know that the expected negative log likelihood is lower bounded by the expected negative log likelihood of the true parameter $\mathcal{W}^*$ governing the data generation process. What it means is that no distribution describes the data as well as the true distribution that generated it. It turns out that, under certain assumptions, we can prove using the central limit theorem that the MLE is asymptotically normal with a mean equal to the true parameter value and variance equal to the inverse of the expected Fisher information evaluated at the true parameter.

If we denote by $\mathbf{X}$ the underlying data distribution, $\mathcal{W}_{\mathbf{X}}$ the MLE and $\mathcal{W}^*_{\mathbf{X}}$ the true parameter, we know that $\mathcal{W}_{\mathbf{X}}$ is a sample from a multivariate gaussian distribution parametrized by $\mathcal{W}^*_X$. Note that in this context we assume the Fisher matrices are invertible.

$$\mathcal{W}_{\mathbf{X}} \sim \mathcal{N}(\mathcal{W}^*_X, \mathbb{I}^{-1}_{\mathbf{X}}(\mathcal{W}^*_X)) \tag{5}$$

However the expected Fisher information on the underlying distribution is intractable. Eventually, using the law of large numbers, we know that the observed Fisher information converges to the expected Fisher information as the sample size increases. Another theoretical limitation is that the true parameter is unknown but we can approximate its observed Fisher information with the observed Fisher information at the MLE because of the consistency of the MLE. For a sake of simplicity we keep the same notation for observed and expected Fisher matrix.

Let denote by $\mathbf{Y}$ the random variable after resampling the underlying distribution $\mathbf{X}$ using an active learning strategy. $\mathcal{W}^*_{\mathbf{X}}, \mathcal{W}^*_{\mathbf{Y}}$ are the true parameters with respect to their respective data distributions and their respective MLE variables $\mathcal{W}_{\mathbf{X}}, \mathcal{W}_{\mathbf{Y}}$, then the following relations hold:

$$\begin{aligned} \mathcal{W}_{\mathbf{X}} &\sim \mathcal{N}(\mathcal{W}^*_{\mathbf{X}}, \mathbb{I}^{-1}_{\mathbf{X}}(\mathcal{W}^*_{\mathbf{X}})) \\ \mathcal{W}_{\mathbf{Y}} &\sim \mathcal{N}(\mathcal{W}^*_{\mathbf{Y}}, \mathbb{I}^{-1}_{\mathbf{Y}}(\mathcal{W}^*_{\mathbf{Y}})) \end{aligned} \tag{6}$$

We thus notice than in an active learning context, the learner is trained on data uniformly sampled from $\mathbf{Y}$, while the optimal solution would be when training on data uniformly sampled from $\mathbf{X}$.

The asymptotic behaviour provides us with a prior distribution of the weights based on the data distribution $\mathbf{X}$. In our context of active learning, we approximate the posterior distribution with the MLE distribution induced by the resampling $\mathbf{Y}$. Hence we define a prior and posterior distribution which did not require to be learnt by backpropagation directly but depend on the two data distribution $\mathbf{X}$ and $\mathbf{Y}$.

$$\begin{aligned} P(\alpha) &\equiv P(\alpha_{\mathbf{X}}) = \mathcal{N}(\mathcal{W}^*_{\mathbf{X}}, \mathbb{I}^{-1}_{\mathbf{X}}(\mathcal{W}^*_{\mathbf{X}})) \\ \mathcal{Q}(\beta) &\equiv \mathcal{Q}(\beta_{\mathbf{Y}}) = \mathcal{N}(\mathcal{W}^*_{\mathbf{Y}}, \mathbb{I}^{-1}_{\mathbf{Y}}(\mathcal{W}^*_{\mathbf{Y}})) \end{aligned} \tag{7}$$

Our active learning scheme relies on the selection of input data whose induced MLE distribution $\mathcal{Q}(\beta_{\mathbf{Y}})$ is minimizing the variational free energy.

$$\mathbf{Y} = \ argmin_{\mathbf{Y}}\ \mathcal{F}(\alpha_{\mathbf{X}}, \beta_{\mathbf{Y}}) \tag{8}$$

It consists in the minimization of the sum of two terms which we denote respectively by the *training factor* $\mathbb{E}_{\mathcal{W}\sim\mathcal{Q}(\beta)}(\mathcal{L}(\mathcal{A};\mathcal{W}))$ and the *generalization factor* $KL(\mathcal{Q}(\beta)\ ||\ \mathcal{P}(\alpha))$. It is possible to analyze both terms independently and explain their role into the minimization:

- **Training factor**: Ideally, the Cramer Rao bound implies that the minimum on the *training factor* is reached when $\mathcal{Q}(\beta)$ matches the asymptotically most efficient estimator of the optimal parameter on the error loss on the observed data. Hence the *training factor* corresponds to the minimization of the error on the observed data $\mathcal{A}$.

- **Generalization factor**: Empirical results Choromanska et al. (2015) tend to show that the variance of the accuracy diminishes as the depth of the classifier increases. So our ultimate goal would be to converge to any set of parameters of the MLE distribution as their effectiveness is similar. The goal of the *generalization factor* is to converge to the asymptotic distribution on the whole input distribution $\mathbf{X}$ and to minimize the error of prediction of new input data.

If the expression of the *variational free energy* provides us a theoretical context to work on, the usage of Fisher matrices of deep networks renders it computationally unaffordable. Indeed the Fisher matrix is a quadratic matrix in terms of the number of parameters of the deep networks. Because of the huge number of parameters involved, such matrices takes a lot of memory and processing them costs a lot of ressources, especially if the operations may be repeated often in the framework (*as it would be the case for every possible query processed by an active learning scheme.*)

The next section 3.2 explains the different approximation proposed to deduce a more friendly user criterion.

3.2 Approximations

3.2.1 KRONECKER FACTORED APPROXIMATION OF THE FISHER INFORMATION OF A CNN

Recently an approximation of the Fisher information for deep architectures has been proposed first for fully connected layer in Martens & Grosse (2015), and then for convolutional layer as well in Grosse & Martens (2016). The block kronecker decomposition content (ψ, τ) is explained in Martens & Grosse (2015); Grosse & Martens (2016)

Based on their decomposition definition, we define the evaluation of blocks of the Fisher information at a certain point $x_i(\psi_{x_i,l}, \tau_{x_i,l})$ and an empirical estimation of the Fisher matrix on a set of data $\mathcal{A}$. A sum up of their decomposition is presented in Eq. (9) while the exact content of the kronecker blocks ψ and τ is left as undescribed in this paper for the sake of concision.

$$\begin{aligned}
\mathbb{I}_{\mathcal{A}}(\mathcal{W}) &= diag([\psi_{\mathcal{A},l}(\mathcal{W}) \otimes \tau_{\mathcal{A},l}(\mathcal{W})]_{l=1}^{L}) \\
\psi_{\mathcal{A},l}(\mathcal{W}) &= \frac{1}{\mid \mathcal{A} \mid} \sum_{x_i \in \mathcal{A}} \psi_{x_i,l}(\mathcal{W}) \\
\tau_{\mathcal{A},l}(\mathcal{W}) &= \frac{1}{\mid \mathcal{A} \mid} \sum_{x_i \in \mathcal{A}} \tau_{x_i,l}(\mathcal{W})
\end{aligned} \tag{9}$$

The strength of this decomposition lies in the properties of block diagonal combined with those of the kronecker product. ψ and τ are respectively related to the covariance matrix of the activation and the covariance of the derivative given the input of a layer. Recent deep architectures tend to prevail the depth over the width (*i.e. the number of input and output neurons*) so this expression becomes really suitable and tractable.

3.3 Approximation of the training factor

Despite the block kronecker product approximation of the Fisher matrix, sampling on $\mathcal{Q}(\beta)$ requires to compute the inverse. Because the kronecker blocks may still have an important number of parameters involved (especially the first fully connected layer suceeding to a convolutional layer), the inverse of the blocks may be still too computationally expensive. To approximate the *training factor*, we opt for a second order approximation of the log likelihood for parameters $\mathcal{W}$ close to the mean parameter $\mathcal{W}_{\mathbf{Y}}^{*}$ of $\mathcal{Q}(\beta)$.

$$\mathcal{L}(\mathcal{A};\mathcal{W}) \approx \mathcal{L}(\mathcal{A};\mathcal{W}_{\mathbf{Y}}^{*}) + \frac{\partial \mathcal{L}(\mathcal{A};\mathcal{W}_{\mathbf{Y}}^{*})}{\partial \mathcal{W}} + (\mathcal{W} - \mathcal{W}_{\mathbf{Y}}^{*})^{T} \frac{\partial^2 \mathcal{L}(\mathcal{A};\mathcal{W}_{\mathbf{Y}}^{*})}{\partial \mathcal{W}' \partial \mathcal{W}} (\mathcal{W} - \mathcal{W}_{\mathbf{Y}}^{*}) \tag{10}$$

Our first approximation consists in assuming that the MLE parameter $\hat{w}$ of the currently trained network is a good approximator of $\mathcal{W}_{\mathbf{Y}}^{*}$. Because the network has converged on the current set of observed data $\mathcal{A}$ the first derivative of the log likelihood is also set to zero. Hence Eq. (10) thus becomes:

$$\mathcal{L}(\mathcal{A};\mathcal{W}) \approx \mathcal{L}(\mathcal{A};\hat{w}) + (\mathcal{W} - \hat{w})^{T} \mathbb{I}_{\mathcal{A}}^{-1}(\hat{w})(\mathcal{W} - \hat{w}) \tag{11}$$

To compute the expectation over the range of weights sampled from $\mathcal{Q}(\beta)$ we need to upperbound the expectation of the dot product of $\mathcal{W}$ given the Fisher matrix. Because we assume our Fisher matrices invertible, and because a covariance matrix is at least semi-definite, our Fisher matrices are positive definite matrix. Hence every eigenvalue is positive and the trace of the Fisher matrix is greater than its maximum eigenvalue. From basic properties of the variance covariance matrix, if we denote by N the number of parameters in the network we obtain the following upperbound for the *training factor*:

$$\mathbb{E}_{\mathcal{W} \sim \mathcal{Q}(\beta)}(\mathcal{L}(\mathcal{A};\mathcal{W})) \leq \mathcal{L}(\mathcal{A};\hat{w}) + \frac{N}{\sqrt{\pi}} Tr(\mathbb{I}_{\mathbf{Y}}^{-1}(\hat{w})^{T} \mathbb{I}_{\mathcal{A}}^{-1}(\hat{w}) \mathbb{I}_{\mathbf{Y}}^{-1}(\hat{w})) \tag{12}$$

When it comes to the trace of the inverse, we approximate it by the closest lower bound with the inverse of the trace like in Wei et al. (2015).

$$\mathbb{E}_{\mathcal{W} \sim \mathcal{Q}(\beta)}(\mathcal{L}(\mathcal{A};\mathcal{W})) \lessapprox \mathcal{L}(\mathcal{A};\hat{w}) + \frac{N}{\sqrt{\pi}} \frac{N^2}{Tr(\mathbb{I}_{\mathbf{Y}}(\hat{w}) \mathbb{I}_{\mathcal{A}}(\hat{w}) \mathbb{I}_{\mathbf{Y}}(\hat{w})^{T})} \tag{13}$$

3.4 Approximation of the generalization factor

Our *generalization factor* corresponds to the KL divergence between the approximation of our posterior $\mathcal{Q}(\beta)$ and the prior $\mathcal{P}(\alpha)$. Because both distributions are multivariate gaussians, we have a direct formulation of the KL which is always definite since the Fisher matrices are invertible:

$$KL(\mathcal{Q}(\beta) \mid\mid \mathcal{P}(\alpha)) = \frac{1}{2}\left(ln\left(\frac{det(\mathbb{I}_{\mathbf{X}}^{-1}(\mathcal{W}_{\mathbf{X}}^{*}))}{det(\mathbb{I}_{\mathbf{Y}}^{-1}(\mathcal{W}_{\mathbf{Y}}^{*}))}\right) - N + Tr(\mathbb{I}_{\mathbf{X}}(\mathcal{W}_{\mathbf{X}}^{*}) \mathbb{I}_{\mathbf{Y}}^{-1}(\mathcal{W}_{\mathbf{Y}}^{*})) + (\mathcal{W}_{\mathbf{X}}^{*} - \mathcal{W}_{\mathbf{Y}}^{*})^{T} \mathbb{I}_{\mathbf{X}}(\mathcal{W}_{\mathbf{X}}^{*})(\mathcal{W}_{\mathbf{X}}^{*} - \mathcal{W}_{\mathbf{Y}}^{*}) \right) \tag{14}$$

Our first approximation consists in assuming that the MLE parameter $\hat{w}$ of the currently trained network is a good approximator of both optimal parameters $\mathcal{W}^*_{\mathbf{X}}, \mathcal{W}^*_{\mathbf{Y}}$ like in Zhang & Oles (2000). We also upper bound the determinant with a function of the trace and the number N of parameters. When it comes to the trace of the inverse, we approximate it again by the closest lower bound with the inverse of the trace.

$$KL(\mathcal{Q}(\beta) \mid\mid \mathcal{P}(\alpha)) \underset{\sim}{\propto} N\left(ln\Big(Tr(\mathbb{I}_{\mathbf{Y}}(\hat{w})\mathbb{I}_{\mathbf{X}}^{-1}(\hat{w}))\Big) + \frac{N}{Tr(\mathbb{I}_{\mathbf{Y}}(\hat{w})\mathbb{I}_{\mathbf{X}}^{-1}(\hat{w}))}\right) \tag{15}$$

3.5 Approximation of the variational free energy

In the previous subsections, we proposed independent approximations of both our sub-criteria: the *training factor* and the *generalization factor*. However the scale of our approximations may not be balanced so we sum up our criterion with an hyperparameter factor γ which counterparts the difference of scale between the factors:

$$\mathcal{F} \underset{\sim}{\propto} \gamma\left(\frac{N}{\sqrt{\pi}}\frac{N^2}{Tr(\mathbb{I}_{\mathbf{Y}}(\hat{w})\mathbb{I}_{\mathcal{A}}(\hat{w})\mathbb{I}_{\mathbf{Y}}(\hat{w})^T)}\right) + N\left(ln\left(Tr\left(\mathbb{I}_{\mathbf{Y}}(\hat{w})\mathbb{I}_{\mathbf{X}}^{-1}(\hat{w})\right)\right) + \frac{N}{Tr(\mathbb{I}_{\mathbf{Y}}(\hat{w})\mathbb{I}_{\mathbf{X}}^{-1}(\hat{w}))}\right) \tag{16}$$

We approximate the expected Fisher matrices on the underlying distribution $\mathbf{Y}$ and $\mathbf{X}$ by the observed Fisher matrices on a set of data sampled from those distributions. This approximation is relevant due to the consistenty of the MLE.

As we are in a pool-based selection case, we dispose at first of two sets of data: $\mathcal{A}$ and $\mathcal{U}$ which denote respectively the *annotated observed data* and *unlabeled data*. Note that the derivatives in the Fisher matrix computation implies to know the label of the samples. Thus at each active learning step, an unknown label is approximated by its prediction from the current trained network. We denote by $\mathcal{S}$ the subset of data to be queried by an oracle. The size of $\mathcal{S}$ is fixed with $\mid \mathcal{S} \mid = K$. $\mathcal{S}$ is the subset sampled from $\mathbf{Y}$ while $\mathcal{U}$ is sampled from $\mathbf{X}$. Finally an approximation of $\mathcal{F}$ will be:

$$\mathcal{F} \underset{\sim}{\propto} \gamma\left(\frac{N}{\sqrt{\pi}}\frac{N^2}{Tr(\mathbb{I}_{\mathcal{S}}(\hat{w})\mathbb{I}_{\mathcal{A}}(\hat{w})\mathbb{I}_{\mathcal{S}}(\hat{w})^T)}\right) + N\left(ln\left(Tr\left(\mathbb{I}_{\mathcal{S}}(\hat{w})\mathbb{I}_{\mathcal{U}}^{-1}(\hat{w})\right)\right) + \frac{N}{Tr(\mathbb{I}_{\mathcal{S}}(\hat{w})\mathbb{I}_{\mathcal{U}}^{-1}(\hat{w}))}\right) \tag{17}$$

Now we express the trace based on the approximation of the Fisher matrix: we consider that every Fisher matrix for CNN is a L diagonal block matrix, with L the number of layers of the CNN. Every block is made of a kronecker product of two terms ψ and τ. We rely on the properties involved by the choice of this specific matrix topology to obtain a more computationally compliant approximation of $\mathcal{F}$ in Eq. (18):

$$\begin{aligned}\mathcal{F} \underset{\sim}{\propto} &\gamma\left(\frac{N}{\sqrt{\pi}}\frac{N^2}{\sum_{l\in(1,L)} Tr(\psi_{\mathcal{S},l}(\hat{w})\psi_{\mathcal{A},l}^{-1}(\hat{w})\psi_{\mathcal{S},l}(\hat{w})^T)Tr(\tau_{\mathcal{S},l}(\hat{w})\tau_{\mathcal{A},l}^{-1}(\hat{w})\tau_{\mathcal{S},l}(\hat{w})^T)}\right)\\ &+ N\left(ln\left(\sum_{l\in(1,L)} Tr\big(\psi_{\mathcal{S},l}(\hat{w})\psi_{\mathcal{U},l}^{-1}(\hat{w})\big)Tr(\tau_{\mathcal{S},l}(\hat{w})\tau_{\mathcal{U},l}^{-1}(\hat{w}))\right)\right.\\ &\left.+ \frac{N}{\sum_{l\in(1,L)} Tr(\psi_{\mathcal{S},l}(\hat{w})\psi_{\mathcal{U},l}^{-1}(\hat{w}))Tr(\tau_{\mathcal{S},l}(\hat{w})\tau_{\mathcal{U},l}^{-1}(\hat{w}))}\right)\end{aligned} \tag{18}$$

4 Active learning as a greedy selection scheme on the variational free energy

The selected subset $\mathcal{S}$ selected at one step of active learning is only involved through the kronecker product of the Fisher matrix $\mathbb{I}_{\mathcal{S}}(\hat{w})$. We express our approximation of the free energy by a criterion on the subset $\mathcal{S}$ in Eq. (19):

$$C(\mathcal{S};\mathcal{A},\mathcal{U}) = \gamma\Bigg(\frac{N}{\sqrt{\pi}}\frac{N^2}{\sum_{l\in(1,L)} Tr(\psi_{\mathcal{S},l}(\hat{w})\psi_{\mathcal{A},l}(\hat{w})\psi_{\mathcal{S},l}(\hat{w})^T)Tr(\tau_{\mathcal{S},l}(\hat{w})\tau_{\mathcal{A},l}(\hat{w})\tau_{\mathcal{S},l}(\hat{w})^T)}\Bigg)$$
$$+ N\Bigg(ln\Bigg(\sum_{l\in(1,L)} Tr(\psi_{\mathcal{S},l}(\hat{w})\psi^{-1}_{\mathcal{U},l}(\hat{w}))Tr(\tau_{\mathcal{S},l}(\hat{w})\tau^{-1}_{\mathcal{U},l}(\hat{w}))\Bigg)$$
$$+ \frac{N}{\sum_{l\in(1,L)} Tr(\psi_{\mathcal{S},l}(\hat{w})\psi^{-1}_{\mathcal{U},l}(\hat{w}))Tr(\tau_{\mathcal{S},l}(\hat{w})\tau^{-1}_{\mathcal{U},l}(\hat{w}))}\Bigg) \tag{19}$$

Finally we estimate our subset $\mathcal{S}$ by a greedy procedure: to be more robust to outliers and for reasons of computational efficiency, we select first a pool of samples $\mathcal{D} \subset \mathcal{U}$ which we will use as the set of possible queries. We recursively build $\mathcal{S} \subset \mathcal{D}$ by picking the next sample $x_i \in \mathcal{D}$ which minimizes $C(\mathcal{S} \cup \{x_i\}; \mathcal{A},\mathcal{U})$ among all remaining samples in $\mathcal{D}$. When it comes to the *training factor* coefficient, we notice that it is a quadratic term in $\mathbb{I}_{\mathcal{S}}(\hat{w})$ which increases the complexity in a greedy selection scheme. Our choice is to estimate the trace in the following way:

$$Tr(\psi_{\mathcal{S}\cup\{x\},l}(\hat{w})\psi_{\mathcal{A},l}(\hat{w})\psi_{\mathcal{S}\cup\{x\},l}(\hat{w})^T) \approx Tr(\psi_{\mathcal{S},l}(\hat{w})\psi_{\mathcal{A},l}(\hat{w})\psi_{\mathcal{S},l}(\hat{w})^T) + Tr(\psi_{\{x\},l}(\hat{w})\psi_{\mathcal{A},l}(\hat{w})\psi_{\{x\},l}(\hat{w})^T)$$

Pseudo-code and illustration of the algorithm are provided in table 1 in appendix.

5 EXPERIMENTS

We demonstrate the validity of our approach on two datasets: MNIST (28-by-28 pictures, 50.000 training samples, 10.0000 validation samples and 10.000 test samples) and USPS (16-by-16 pictures, 4185 training samples, 464 validation samples and 4649 testing samples) both gray scaled digits image datasets. We describe the CNN configuration and the hyperparameters settings in table 2 in appendix. Note that we do not optimize the hyperparameters specifically for the size of the current annotated training set $\mathcal{A}$. We picked those two similar datasets to judge of the robustness of our method against different size of unlabeled datasets, as expected our method is efficient on both small and large databases.

5.1 TEST ERROR

We run 10 runs of experiments and average the error on the test set of the best validation error before a pass of active learning. We start from an annotated training set of the size of one minibatch selected randomly. We stop both set of experiments after 30% of the training set has been selected (15.000 image for MNIST, 1255 for USPS). We compare the lowest test error achieved so far by our MLE based method against naive baselines: uncertainty sampling, curriculum sampling and a random selection of a minibatch of examples. We measure both uncertainty and curriculum scores based on the log likelihood of a sample using as label its prediction on the full network. While uncertainty selects samples with the highest log likelihood, our version of curriculum does the exact contrary. We select randomly the set of possible queries $\mathcal{D}$ among the unlabeled training data. Its size is set to 30 times the minibatch size. We present the results in two phases for the sake of clarity in figure 1 for MNIST and figure 2 for USPS: the first rounds of active learning when the annotated training set is almost empty, and the second round which is more stable in the evolution of the error. In both phases and for both databases we observe a clear difference between the test error achieved by our MLE method with the test error obtained by selecting randomly the data to be queried. Moreover the error achieved by our method on 30 % is close (*even equivalent in the case of USPS*), to the error achieved using the standard full training sets defined for both datasets (this error rate is defined as yellow line groundtruth on the figures). The experiments made appear that curriculum learning is not a good active learning strategy for both tested datasets. As for the uncertainty selection, it works really well on MNIST while it fails on USPS. While MNIST is a pretty clean database, USPS contains more outliers and noisy samples rendering it more difficult in terms of accuracy even though both databases are designed to assess digit classification. As other works we mentioned in the related work section, we are led to explain uncertainty selection to select useless samples with the amount of outliers and noisy samples in USPS.

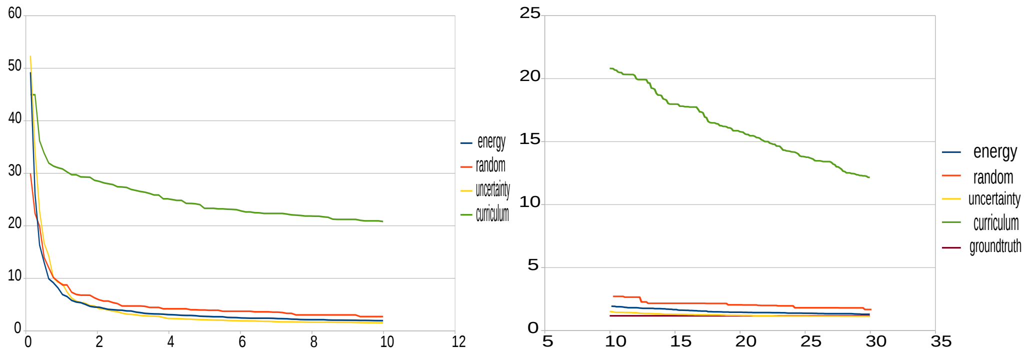

Figure 1: Error on the test set splits in two figures : the first rounds of active learning and the second round which is more stable in the evolution of the error (MNIST)

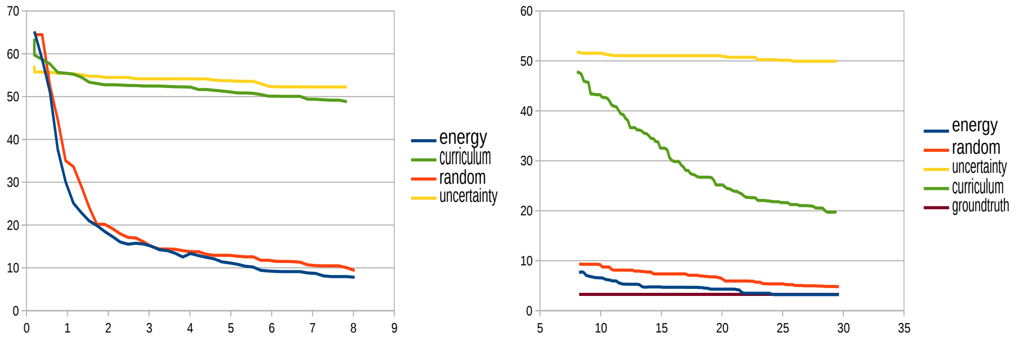

Figure 2: Error on the test set splits in two figures : the first rounds of active learning and the second round which is more stable in the evolution of the error (USPS)

5.2 Time complexity

To validate our method in terms of scalability and time complexity, we measured in seconds, the current processor time for one pass of active learning. We repeated this evaluation for different size of query (8 to 128 unlabeled samples added to the current training set). For this experiments we used a laptop with a Titan-X (GTX 980 M) with 8 GB RAM GPU memory. Metrics were reported in figure 3. Our criterions takes few seconds to select a batch of query of hundreds of unlabeled data. Moreover the evolution of the time given the size of the query is less than linear.

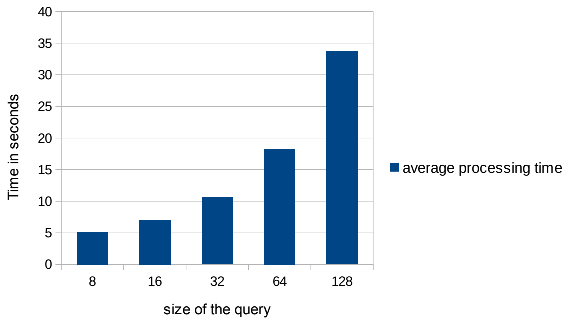

Figure 3: Average processing time for one pass of our active learning strategy vs the size of the selected samples for annotation (USPS).

6 DISCUSSION

We believe our method is a proof of concept for the use of variational inference for active learning on deep neural networks. However our approximations are subject to improvements which may lead to faster convergence and lower generalization error.

The first point to raise is that our approximation of the posterior is an asymptotic distribution which may be unstable on a small subset of observed data, as it is the case for active learning. Such a distribution may be regularized by taking the probability provided by the central limit theorem about how well our data fits to the asymptotic gaussian distribution. When it comes to the KFAC approximation, it suffers from the same issue and could be regularized when evaluating on small subset. A refinement of the approximations, especially for the *generalization factor*, following the approaches of submodular functions may be investigated.

Finally, an interesting observation is that our formulation of the variational free energy finds similarities with other MLE based active learning criteria previously proposed in the litterature. Indeed, in Zhang & Oles (2000) the authors study active learning by looking among the possible resampling of the input distribution. They formulate their criterion as the minimization of the trace of the inverse Fisher of the resampled distribution multiplied by the Fisher matrix on the input distribution: $\min_{\mathcal{S}} Tr(\mathbb{I}_{\mathcal{S}}^{-1}(\hat{w})\mathbb{I}_{\mathcal{U}}(\hat{w}))$

7 CONCLUSION

In a nutshell, we proposed a scalable batch active learning framework for deep networks relying on a variational approximation to perform bayesian inference. We deduced a formulation of the posterior and prior distributions of the weights using statistical knowledge on the Maximum Likelihood Estimator. Those assumptions combined with an existing approximation of the Fisher information for neural network, lead us to a backpropagation free active criterion. Eventually we used our own approximations to obtain a greedy active selection scheme.

Our criterion is the first of the kind to scale batch active learning to deep networks, especially Convolutional Neural Networks. On different databases, it achieves better test accuracy than random sampling, and is scalable with increasing size of queries. It achieves near optimal error on the test set using a limited percentage (30%) of the annotated training set on larger and more reduced dataset. Our works demonstrated the validity of batch mode active learning for deep networks and the promise of the KFAC approximations for deep Fisher matrices for the active learning community. Such a solution is also interesting as a new technique for curriculum learning approach.

REFERENCES

Choromanska, Anna, Henaff, Mikael, Mathieu, Michael, Arous, Gérard Ben, and LeCun, Yann. The loss surfaces of multilayer networks. In *AISTATS*, 2015.

Cohn, David A. Neural network exploration using optimal experiment design. 1994.

Golovin, Daniel, Krause, Andreas, and Ray, Debajyoti. Near-optimal bayesian active learning with noisy observations. In *Advances in Neural Information Processing Systems*, pp. 766–774, 2010.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and Harnessing Adversarial Examples. *ICLR 2015*, December 2015.

Graves, Alex. Practical variational inference for neural networks. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, NIPS'11, pp. 2348–2356, USA, 2011. Curran Associates Inc. ISBN 978-1-61839-599-3. URL `http://dl.acm.org/citation.cfm?id=2986459.2986721`.

Grosse, Roger and Martens, James. A kronecker-factored approximate fisher matrix for convolution layers. *arXiv preprint arXiv:1602.01407*, 2016.

Hoi, Steven C. H., Jin, Rong, Zhu, Jianke, and Lyu, Michael R. Batch mode active learning and its application to medical image classification. ICML '06, pp. 417–424, New York, NY, USA, 2006.

ACM. ISBN 1-59593-383-2. doi: 10.1145/1143844.1143897. URL `http://doi.acm.org/10.1145/1143844.1143897`.

Martens, James and Grosse, Roger. Optimizing neural networks with kronecker-factored approximate curvature. *arXiv preprint arXiv:1503.05671*, 2015.

Settles, Burr. *Active Learning*, volume 6 of *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan & Claypool Publishers, 2012.

Szegedy, Christian, Zaremba, Wojciech, Sutskever, Ilya, Bruna, Joan, Erhan, Dumitru, Goodfellow, Ian, and Fergus, Rob. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL `http://arxiv.org/abs/1312.6199`.

Talwalkar, Ameet. *Matrix Approximation for Large-scale Learning*. PhD thesis, New York, NY, USA, 2010.

Wei, Kai, Iyer, Rishabh, and Bilmes, Jeff. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, pp. 1954–1963, 2015.

Zhang, Tong and Oles, F. The value of unlabeled data for classification problems. In *Proceedings of the Seventeenth International Conference on Machine Learning,(Langley, P., ed.)*, pp. 1191–1198. Citeseer, 2000.

Zhou, Shusen, Chen, Qingcai, and Wang, Xiaolong. Active deep networks for semi-supervised sentiment classification. In *ACL International Conference on Computational Linguistics*, pp. 1515–1523, 2010.

Algorithm 1: Greedy selection of the final query $\mathcal{S}$

```
Require :$\mathcal{A}$ set of initial annotated training examples
Require :$\mathcal{U}$ set of initial unlabeled training examples
Require :$\mathcal{D}$ set of possible queries, $\mathcal{D} \subset \mathcal{U}$
Require :N number of parameters
Require :L number of layers
Require :K number of samples to query
Require :$\gamma$ scaling hyperparameter
$\mathcal{S}^0 = \{\};\ \mathcal{D}^0 = \mathcal{D}$
for k $[| 1, L |]$ do
    # init the coefficients of each layer with 0 value:
    for $x \in \mathcal{D}^k$ do
        for l in $[| 1, L |]$ do
            $V_l^k = [0, 0, 0, 0]$
        end
    end
    # Compute the inverse Fisher information on the whole input distribution
    for l in $[| 1, L |]$ do
        $a_l = \psi_l(\mathcal{A})^{-1}; b_l = \tau_l(\mathcal{A})^{-1}$
        $c_l = \psi_l(\mathcal{U})^{-1}; d_l = \tau_l(\mathcal{U})^{-1}$ [a]
    end
    # coefficient for the greedy selection for $x \in \mathcal{D}^k$ do
        for l in $[| 1, L |]$ do
            $V_{l,0}(x) = Tr(\psi_{x,l} a_l \psi_{x,l}^T)$
            $V_{l,1}(x) = Tr(\tau_{x,l} b_l \tau_{x,l}^T)$
            $V_{l,2}(x) = Tr(\psi_{x,l} c_l)$
            $V_{l,3}(x) = Tr(\tau_{x,l} d_l)$
        end
        # compute the trace
        $Z_0(x) = (\frac{1}{k+1})^2 \sum_{l \in (1,L)} (V_{l,0}^k + V_{l,0})(V_{l,1}^k + V_{l,1})$
        $Z_1(x) = (\frac{1}{k+1})^2 \sum_{l \in (1,L)} (V_{l,2}^k + V_{l,2})(V_{l,3}^k + V_{l,3})$
    end
    # select the best sample in $\mathcal{D}^k$ based on the criterion $C$:
      $x_k = argmax_x\ \gamma \frac{1}{Z_0(x)} + N(ln(Z_1(x)) + \frac{N}{Z_1(x)})$
    $\mathcal{S}^{k+1} \leftarrow \mathcal{S}^k \cup \{x_k\}$
    $\mathcal{D}^{k+1} \leftarrow \mathcal{D}^k \setminus \{x_k\}$
end
$\mathcal{S} \leftarrow \mathcal{S}^K$
```

[a] If blocks are too big, an approximation of the inverse by a Woodburry-Nystrom is processed Talwalkar (2010)

Table 1: Pseudo code for the greedy selection given the variatiational free energy

For reasons of computational efficiency; we first sample a larger subset $\mathcal{D} \subset \mathcal{U}$ from which we select the queries. To assert that this subset is not detrimental for uncertainty selection, we present a one shot experiments on USPS with $\mathcal{D} \equiv \mathcal{U}$ in fig **??**.

hyper parameters	**MNIST**	**USPS**
# filters	[20, 20]	[20, 20]
filter size	[(3,3), (3,3)]	[(3,3), (3,3)]
pooling size (no stride)	[(2,2), (2,2)]	[None, (2,2)]
activation	Rectifier	Rectifier
# neurons in full layers	[200, 200, 50, 10]	[300, 50, 10]
# batch size	64	8

Table 2: Set of hyperparameters used respectively on the CNN for MNIST and SVHN

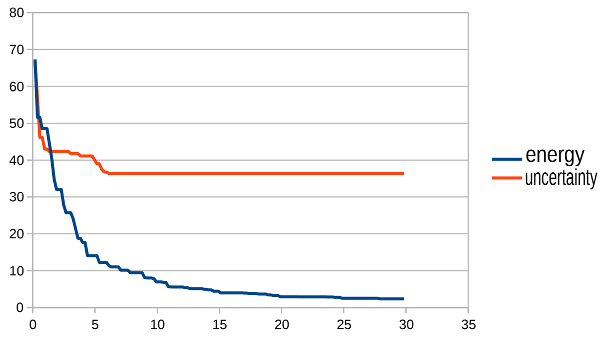

Figure 4: Error on the test set when we do not sample first a larger subset where to pick the queries ($\mathcal{D} \equiv \mathcal{U}$, USPS)