NEURAL NETWORK TRAINING VARIATIONS IN SPEECH AND SUBSEQUENT PERFORMANCE EVALUATION

Ewout van den Berg, Bhuvana Ramabhadran & Michael Picheny
IBM Watson Group
1101 Kitchawan Rd.
Yorktown Heights, NY 10598, USA
{evandenberg,bhuvana,picheny}@us.ibm.com

ABSTRACT

In this work we study variance in the results of neural network training on a wide variety of configurations in automatic speech recognition. Although this variance itself is well known, this is, to the best of our knowledge, the first paper that performs an extensive empirical study on its effects in speech recognition. We view training as sampling from a distribution and show that these distributions can have a substantial variance. These observations have important implications on way results in the literature are reported and interpreted.

1 INTRODUCTION

In automatic speech recognition (ASR), the goal is to develop a combination of language and acoustic models that together minimize the decoding word-error rate (WER) on predefined tasks. Recently, deep neural networks and variations, such as convolution neural networks, have been used with great success in ASR, and motivated by these results as well as those in machine vision, there is now very active research in network architectures, training algorithms, feature representations, and data augmentation. The performance of a newly proposed approach is typically evaluated by comparing it against the performance of a baseline system. This process typically involves extensive tuning of the hyperparameters of the new system followed by re-training and evaluating the performance of the system. Once an instance of the approach is found that substantially improves the WER (in the case of ASR), it is concluded that the method, architecture, or training algorithm is successful.

2 NEURAL NETWORK TRAINING AS SAMPLING

For the vast majority of conventional neural networks, training involves minimizing a cost function that is highly non-convex. Given that optimization is done with techniques that were designed for convex problems, such as stochastic gradient descent (SGD), it should come as no surprise that the solution of a training run depends on the initial starting point, as well as various factors that affect the optimization process, such as the mini-batch randomization (see also (Choromanska et al., 2014; Pascanu et al., 2014; Pinto et al., 2009)). Although this phenomenon is rarely discussed explicitly in the literature, it is common knowledge to the practitioner, and indeed it was leveraged in work by Hinton et al. (2015) to generate ensembles. Generally, the outcome of the neural network training is deterministic given the parameters and the state of the pseudo-random number generators (PRNG); we here exclude asynchronous algorithm, in which the exact timing and network condition may affect the outcome. Unless controlled explicitly, the PRNG state can be viewed as a hidden random variable and it then follows that training of a neural network is equivalent to sampling a result from some distribution that is conditioned on the tunable parameters. Namely, training the network starting from all finitely many PRNG states gives a mapping from states to results, and actual training of the network gives the result corresponding to the random initial PRNG state. In the next section we perform extensive experiments that empirically show what these distributions look like for practical ASR tasks. We discuss the implications on the evaluation of new methodologies in Section 4.

3 NUMERICAL RESULTS

3.1 EXPERIMENT SETUP

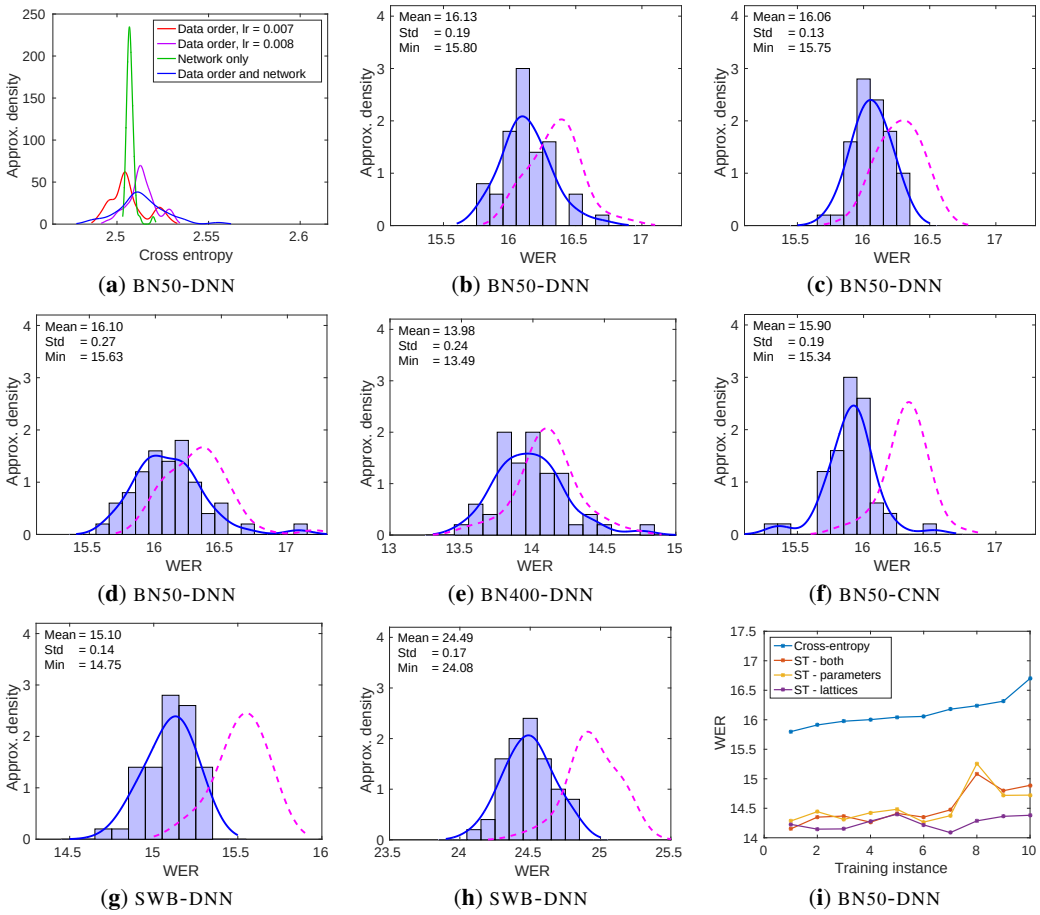
For the training and evaluation of the systems we use three datasets. The first two datasets are based on the English Broadcast News (BN) training corpus (Fiscus et al., 1998) which we pre-process to obtain 40-dimensional logmel features with speaker-dependent mel filters chosen from a set of 21 possible filters. The first dataset (BN400) contains the complete 400-hour BN training set and a 30-hour hold-out set. The second dataset (BN50) uses only a subset of the data and defines a 45-hour training set and a 5-hour hold out set (Kingsbury, 2009). For evaluation we use EARS Dev-04f, as described by Kingsbury (2009). we generate logmel features based on the 300h Switchboard corpus (Godfrey & Holliman, 1997; Liu et al., 1995). Evaluation is done on the Hub5-2000 and CallHome corpora Saon et al. (2015). For all decodes we use the trigram language model as described in (Kingsbury, 2009).

For the acoustic model we consider both DNN and CNN architectures, each with a softmax output layer mapping to 5999 tri-phone states for BN and 9300 for Switchboard. The DNN has an input layer that takes the logmel features with ± 4 temporal context for BN and ± 5 for SWB. This input is then transformed through five hidden linear layers, each with a sigmoid activation function and an output dimension of 1024 for BN and 2048 for SWB, before reaching a softmax output layer. The CNN consists of two convolution layers (with max pooling and respectively 128 filters of size $9\times9\times3$ and 25 of size $3\times4\times128$), two hidden linear layers with sigmoid activation of size 1024, and a final output layer. The input features are logmel with ± 4 frames context and Δ and Δ^2 information.

Most of the training is done by minimizing a cross-entropy loss function using SGD in combination with the newbob schedule in which the learning rate is halved whenever there is an insufficient decrease in the held-out loss. In addition, we reset the network weights to those from the previous epoch whenever the held-out loss increases. For the DNN we apply greedy layerwise pre-training, with one epoch per layer (Sainath et al., 2012). For the BN50-DNN task we also apply Hessian-free sequence training Kingsbury (2009); Martens (2010); Kingsbury et al. (2012).

3.2 RESULTS

By controlling the random seeds for the data order and network initialization we can get an empirical distribution of the neural network training results. Unless otherwise noted, we vary both the parameter and data order seeds. We first consider BN50h DNN training with different configurations: (1) same seed for the parameters, different seeds for the data order, initial learning rate 0.07; (2) same as (1) but with learning rate 0.08; (3) different seeds for parameters, same seed for the data order; and (4) different seeds for parameters and data order. For each configuration we train 50 models with different initial seed values over 30 epochs. The kernel density estimation for the final network, with Gaussian kernels and manually chosen standard deviation, are plotted in Figure 1(a). The largest variance in results is obtained for setting (4) in which both seeds are varied. Keeping the initial parameter seed fixed, as done in settings (1,2) reduces the training variance. The variance for setting (3) in which we only vary the seed for the initial parameters is very small. This suggests, at least in this case, that the data order affects the final outcome much more than the initial parameters. Figure 1(b)–(d) respectively show the WER density estimation for settings (2)–(4). The blue and dashed magenta line indicate estimations obtained for decodings with different acoustic weights. For the results in blue, the mean is fairly constant. The standard deviation is again largest (0.27) when both seeds are varied, and smallest (0.13) when the data order is kept fixed and only the initial parameters differ. Increasing the amount of data often leads to substantial improvement in the WER. This is reflected in the BN400-DNN results shown in Figure 1(e). We expected the standard deviation in the results to decrease accordingly, but only found it to decrease from 0.27 to 0.24. The distribution for CNN training is shown in Figure 1(f). Compared to the equivalent DNN training (3) we see a slightly lower minimum and mean WER, as well as a smaller standard deviation. The WER for the SWB systems evaluated on the Hub5-2000 and CallHome test sets are summarized in Figures 1(g) and (h), respectively. For sequence training we took ten networks from setting (2) and generated enumerator and denominator lattices using a unigram language model. Labeling the networks n_1 through n_{10} in increasing WER, and the corresponding lattices ℓ_1 through ℓ_{10} we considered three ST settings: (1) initial network n_i, lattices ℓ_i; (2) n_i and ℓ_1; and (3) n_1 and ℓ_i; for $i = 1 \dots 10$. The resulting WERs are plotted in Figure 1(i) and connected by lines for clarity. All ST results can be



(**a**) BN50-DNN (**b**) BN50-DNN (**c**) BN50-DNN

(**d**) BN50-DNN (**e**) BN400-DNN (**f**) BN50-CNN

(**g**) SWB-DNN (**h**) SWB-DNN (**i**) BN50-DNN

Figure 1: Result obtained for different datasets and neural network types, as indicated next to the labels. Plot (a) gives the kernel density estimation for the cross-entropy loos; (b–h) give the density estimates for the word-error rate obtained with SGD; and (i) summarizes the results obtained with sequence training.

seen to improve substantially over the cross-entropy results. Interestingly, the starting point seems to be much more important than the network quality used to generate the lattices.

4 Discussion

The results in Figure 1 show that by merely changing the random seeds we obtain a wide range of word error rates across network settings and data sets. For example in BN50-DNN we obtain 15.6 to 17.0, which far exceeds the impact of many typical algorithm enhancements. In fact, even the standard deviation (0.27) nearly reaches the 0.3 absolute improvement in WER commonly reported in the literature. This raises serious questions on how meaningful such results really are. For the BN50-DNN example, we could sample a baseline from the distribution and then repeatedly sample other instances and with high probability conclude that the method improves upon itself! Although this may seem far fetched, extensive fine tuning does essentially correspond to sampling from potentially very similar distributions, thereby increasing the chance of sampling a fortuitous outlier. Of course, improvements of 0.3 can certainly be meaningful (as shown for example by the decoding results with different acoustic weights), but based on individual WERs we simply cannot tell. On the other hand, we concede that extensively sampling the distributions as done in Figure 1 is practically infeasible. The challenge therefore will be to find a metric for comparing distributions that requires a minimal number of samples. Until then, we at hope that the results presented in this paper will raise awareness of this important issue.

REFERENCES

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. arXiv:1412.0233, 2014.

Jonathan Fiscus, John Garofolo, Mark Przybocki, William Fisher, and David Pallett. 1997 English Broadcast News Speech (HUB4) LDC98S71, 1998. Linguistic Data Consortium.

John Godfrey and Edward Holliman. Switchboard-1 Release 2 LDC97S62, 1997. Linguistic Data Consortium, Philadelphia.

Geoffrey Hinton, Oriol Vinyals, and Jeff dean. Distilling the knowledge in a neural network. arXiv:1503.02531, 2015.

Brian Kingsbury. Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling. In *Proceedings of ICASSP*, pp. 3761–3764, 2009.

Brian Kingsbury, Tara Sainath, and Hagen Soltau. Scalable minimum Bayes risk training of deep neural network acoustic models using distributed Hessian-free optimization. In *Proceedings of Interspeech*, 2012.

Fu-Hua Liu, Mike Monkowski, Mirek Novak, Mukund Padmanabhan, Michael Picheny, and Patibandla Srinivasa Rao. Performance of the IBM LVCSR system on the Switchboard corpus. In *Speech Research Syposium*, pp. 189, 1995.

James Martens. Deep learning via Hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010.

Razvan Pascanu, Yann N. Dauphin, Surya Ganguli, and Yoshua Bengio. On the saddle point problem for non-convex optimization. arXiv:1405.4604, 2014.

Nicolas Pinto, David Doukhan, James J. DiCarlo, and David D. Cox. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLos Comput Biol*, 5(11):e100579, 2009.

Tara N. Sainath, Brian Kingsbury, and Bhuvana Ramabhadran. Improving training time of deep belief networks through hybrid pre-training and larger batch sizes. In *Proceedings of the NIPS Workshop on Log-linear Models*, 2012.

George Saon, Hong-Kwang J. Kuo, Steven Rennie, and Michael Picheny. The IBM 2015 English conversational telephone speech recognition system. arXiv:1505.05899, 2015.