Interspeech 2018
2-6 September 2018, Hyderabad

Inference-Invariant Transformation of Batch Normalization for Domain Adaptation of Acoustic Models

Masayuki Suzuki, Tohru Nagano, Gakuto Kurata, Samuel Thomas

IBM Research AI

{szuk, tohru3, gakuto}@jp.ibm.com, sthomas@us.ibm.com

Abstract

Batch normalization, or batchnorm, is a popular technique often used to accelerate and improve training of deep neural networks. When existing models that use this technique via batchnorm layers, are used as initial models for domain adaptation or transfer learning, the novel input feature distributions of the adapted domains, considerably change the batchnorm transformations learnt in the training mode from those which are applied in the inference mode. We empirically find that this mismatch can degrade the performance of domain adaptation for acoustic modeling. To mitigate this degradation, we propose an inference-invariant transformation of batch normalization, a method which reduces the mismatch between training mode and inference mode transformations without changing the inference results. This invariance property is achieved by adjusting the weight and bias terms of the batchnorm to compensate for differences in the mean and variance terms when using the adaptation data. Experimental results show that our proposed method performs the best on several acoustic model adaptation tasks with up to 5% relative improvement in recognition performances in both supervised and unsupervised domain adaptation settings.

Index Terms: Batch normalization, domain adaptation, transfer learning

1. Introduction

Batch normalization (batchnorm) [1] is a useful technique to accelerate and improve the training of deep neural networks. This technique normalizes activations to reduce the effect of internal covariate shift, which reduces the efficiency of neural network training. Batchnorm makes it possible to use significantly higher learning rates, and reduces the sensitivity to initialization. It has been used in many state-of-the-art deep neural networks, including convolutional neural networks such as very deep convolutional neural networks (VGG) [2, 3] and deep residual networks [4, 5].

Many state-of-the-art neural network models have batchnorm layers and are often used as initial models for domain adaptation and transfer learning. For example, a general purpose acoustic model trained using large amounts of data from various sources is a good initial point to train an adapted acoustic model for a specific call center with a small amount of target domain adaptation data [6, 7, 8, 9, 10]. In a low-resource language scenario, a model trained with rich-resource language data or multiple low-resource language data is a good starting point to train an acoustic model for a target low-resource language by replacing and randomly initializing only the top softmax layer [11, 12].

Batchnorm applies different transformations in training and inference. In training, statistics of activations in the mini-batch are used to normalize activations. In inference, the averages of these statistics are used for the normalization. If the mini-batch size is large enough and input feature distribution is i.i.d., the parameter update of the neural network dominates the difference between these two transformations [13]. In such cases, batchnorm effectively reduces the negative effect of internal covariance shift.

When domain adaptation or transfer learning starts, the change in the input feature distribution increases the difference between training and inference. In inference with a base model, batchnorm uses averaged statistics of the original domain data. In training, however, batchnorm uses statistics of the target domain data in the mini-batch. The distribution of their activations are different because the input feature distributions have changed. We empirically observe that this mismatch can often degrade the performance of domain adaptation for acoustic modeling. Although the simplest way to compensate for this mismatch would be to freeze the parameters of batchnorm (batchnorm freezing), such a modification is likely to nullify the advantages of using this technique.

We propose inference-invariant transformation (IIT) of batch normalization to solve this problem. We use averages of the statistics of the target domain adaptation data instead of the original domain data for normalization in batchnorm. While this change can reduce the mismatch, it can also change the inference results. To compensate for the change in inference results, we numerically adjust the weight and bias parameters of batchnorm in advance so that the inference results are invariant. Experimental results on several acoustic model adaptation tasks show that our proposed method performs better than or similar to the performance of basic adaptation or batchnorm freezing.

The rest of this paper is organized as follows. Section 2 describes the batchnorm technique. In Section 3 we introduce the IIT of batchnorm and summarize the proposed changes in Fig. 1. After reviewing related work in Section 4, we evaluate various techniques in Section 5 in both supervised and unsupervised settings. Our proposed technique performs well and provides up to 5% relative improvement over baseline system performances. The paper concludes with a summary in Section 6.

2. Batch Normalization

In the training mode, batchnorm normalizes input activations in a mini-batch $\{x_{1\ldots m}\}$ using the mean μ_B and variance σ_B^2 of the mini-batch as follows.

$$\text{batchnorm}(x_i) \quad = \quad \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta, \qquad (1)$$

where γ and β are trainable parameters and ϵ is a low value, such as 10e-4, to stabilize the transformation. #1 in Fig. 1 uses (1) as batchnorm. Depending on the input activations in the mini-batch, μ_B and σ_B^2 change, and so do their transformations.

10.21437/Interspeech.2018-1563

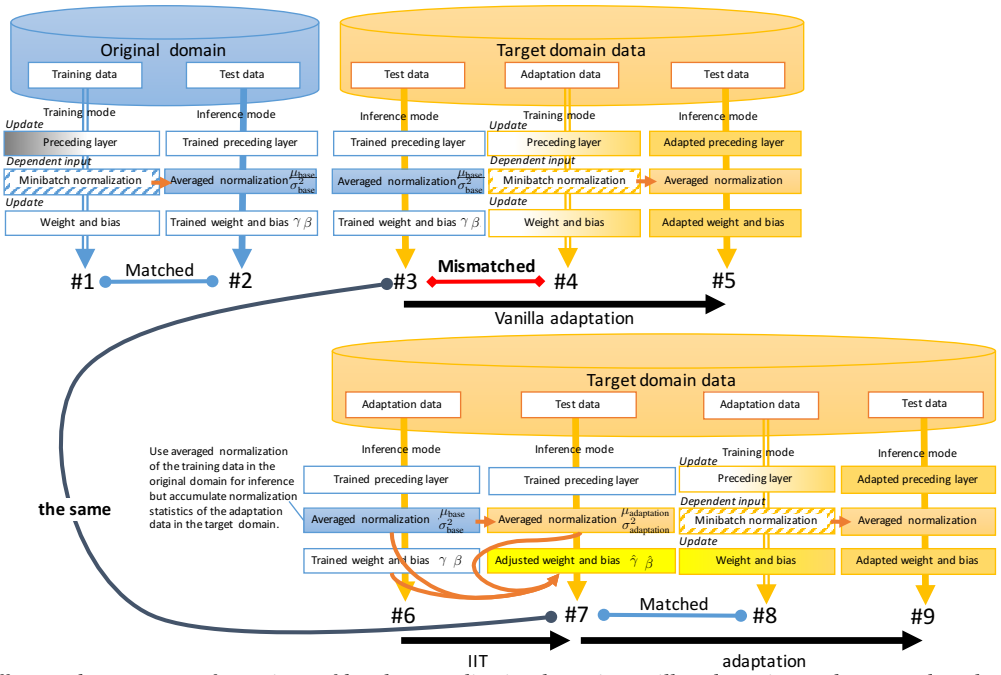

Figure 1: *Difference between transformations of batch normalization layer in vanilla adaptation and proposed method. #1 and #2 represent typical steps for the training of a base model and evaluation. #3 should be a initial point for domain adaptation. #4 and #5 represent steps for adaptation and evaluation in a target domain. When vanilla adaptation starts, there is a mismatch because the mean and variance of the mini-batch in #4 are considerably different from μ_{base} and σ_{base}. With these mismatches #3 is clearly not an appropriate starting point for vanilla adaptation. Our proposed method first accumulates $\mu_{adaptation}$ and $\sigma_{adaptation}$ by fixing the other parameters as in #6. Then, it calculates $\hat{\gamma}$ and $\hat{\beta}$ so that #7 and #3 has the same inference result. Our proposed method uses #7, which has an equivalent inference result to #3, as an initial point for domain adaptation. #8 and #9 represent typical steps for adaptation and evaluation in a target domain. Unlike vanilla adaptation, there is no mismatch between #7 and #8 when adaptation starts. Our proposed method can hence properly leverage #3 as a starting point for domain adaptation.*

In the inference mode, batchnorm does not use μ_B and σ_B. Instead, the averaged mean and variance of the training data of a base model μ_{base} and σ_{base} are used as follows.

$$\text{batchnorm}_{\text{inference}}(x_i) = \gamma \frac{x_i - \mu_{\text{base}}}{\sqrt{\sigma^2_{\text{base}} + \epsilon}} + \beta. \qquad (2)$$

μ_{base} and σ_{base} are often estimated as the final value of the running mean and variance during training. #2 in Fig. 1 represents this transformation.

If the mini-batch size m is large enough and the input feature distribution is i.i.d., μ_B and σ_B are similar to μ_{base} and σ_{base}, respectively [13]. Therefore, after training has finished, #1 and #2, as in Fig. 1, both have matched inference results.

Equation (2) is used for inference with any test data for various input activations. As in #3 of Fig. 1, when the test data now comes from a domain with *different* characteristics than the train data, the new test data is still transformed using the *original* transformations estimated in a different data setting. This clearly introduces mismatches that result in performance degradation. If however an inference mode adapation (vanilla adaptation) were to be done, to obtain #5 as shown in Fig. 1, #3 would be a good initial point for domain adaptation.

When domain adaptation using domain specific data is employed to update the acoustic model, the mini-batch statistics of the target domain data are used for normalization, instead of μ_{base} and σ_{base}. In #4 of Fig. 1, the target domain data is transformed using (1), where μ_B and σ_B are calculated using activations from mini-batches of target domain data. Because the input feature distribution of a target domain is usually different from that of the original domain, the statistics of activations in inference μ_B and σ_B are considerably different from μ_{base} and σ_{base}, respectively. In other words, there is a mismatch between #3 and #4 in Fig. 1.

3. Proposed Methods

3.1. Batchnorm freezing

A simple way to remove the mismatch is freezing the parameters of batchnorm as $\gamma, \beta, \mu_{\text{base}}, \sigma^2_{\text{base}}$. In this case, (2) is used both for inference and training. Since (2) is an affine transformation of activations, it can be folded down into previous affine transformations of the neural network. The resulting neural network will however now not have any batchnorm layer. Batchnorm freezing can therefore be viewed as a technique that removes batchnorm layers without changing inference results. This in turn removes mismatches but nullifies the benefits of batchnorm.

3.2. Inference-invariant transformation of batchnorm

We propose inference-invariant transformation (IIT) of batchnorm to remove the mismatch without changing inference results and freezing batchnorm layers. IIT enables the use of #3 in Fig. 1 as an initial point of domain adaptation for acoustic modeling.

To perform the propose inference-invariant transform, we

begin by estimating $\mu_{\text{adaptation}}$ and $\sigma_{\text{adaptation}}$ of the adaptation data in #6 of Fig. 1. These statistics are estimated off the adaptation data, similar to the estimation of μ_{base} and σ_{base}, after fixing all the parameters for inference. $\mu_{\text{adaptation}}$ and $\sigma_{\text{adaptation}}$ are similar to μ_B and σ_B for a mini-batch of the target domain data when domain adaptation starts.

IIT then adjusts the weight and bias parameters in batchnorm from γ and β to $\hat{\gamma}$ and $\hat{\beta}$ so that the inference results do not change when statistics for normalization are changed from μ_{base} and σ_{base} to $\mu_{\text{adaptation}}$ and $\sigma_{\text{adaptation}}$. Specifically, we defined $\hat{\gamma}$ and $\hat{\beta}$ so that they satisfy the follow equality

$$\gamma \frac{x_i - \mu_{\text{base}}}{\sqrt{\sigma^2_{\text{base}} + \epsilon}} + \beta = \hat{\gamma} \frac{x_i - \mu_{\text{adaptation}}}{\sqrt{\hat{\sigma}^2_{\text{adaptation}} + \epsilon}} + \hat{\beta}. \tag{3}$$

The closed forms of $\hat{\gamma}$ and $\hat{\beta}$ are then calculated as

$$\hat{\gamma} = \gamma \frac{\sqrt{\sigma^2_{\text{adaptation}} + \epsilon}}{\sqrt{\sigma^2_{\text{base}} + \epsilon}}, \tag{4}$$

$$\hat{\beta} = \gamma \frac{\mu_{\text{adaptation}} - \mu_{\text{base}}}{\sqrt{\sigma^2_{\text{base}} + \epsilon}} + \beta. \tag{5}$$

#7 in Fig. 1 shows a transformation using this equation. #7 is the similar to #3 and can be an initial point for domain adaptation.

We call this method inference-invariant transformation (IIT). When domain adaptation starts after IIT,

$$\text{batchnorm}_{\text{IIT}}(x_i) = \hat{\gamma} \frac{x_i - \mu_B}{\sqrt{\sigma^2_B + \epsilon}} + \hat{\beta} \tag{6}$$

is used for batchnorm in the training mode instead of (1). #8 in Fig. 1 represents this transformation. It matches #7, which is the same as #3, because the statistics of the mini-batch in the adaptation data are similar to $\mu_{\text{adaptation}}$ and $\sigma_{\text{adaptation}}$. Therefore, IIT makes it possible to use #3 as the initial point of domain adaptation.

4. Related Works

In batch re-normalization [13], statistics of the entire training data are leveraged to estimate batchnorm transformations in the training mode. Although this method provides good gains when the mini-batch sizes is small or non-i.i.d, it is less effective when an input feature distribution is changed, as is in the case in domain adaptation tasks.

In a similar technique called AdaBN [14], the averaged normalization used in the inference mode is replaced using statistics estimated off the adaptation data. AdaBN changes the inference results but does not require an adaptation step using backpropagation after the technique has been applied. In [10], AdaBN has been shown to produce good results for domain adaptation of image classification tasks. IIT on the other hand, does not change the inference results but assumes that adaptation using backpropagation is performed after it has been applied.

5. Experiments

To evaluate the usefulness of our proposed technique, we performed domain adaptation experiments in both supervised and unsupervised settings.

5.1. Supervised adaptation

For our first evaluation of the proposed technique in a supervised setting we used a narrowband (8kHz sampling rate) Japanese neural network based acoustic model trained on about 2K hours of narrowband speech from various sources. The adaptation data for this task was about 10 hours of manually transcribed telephone conversations from a specific call center collected outside of the training set. 0.5 hours of conversations from the same adaptation data setting was used as a test set for this experiment.

The acoustic model used in our experiments was a VGG based deep neural network operating on blocks of 48 consecutive 40-dimensional log-mel frames augmented with first and second derivatives. The logmels were globally variance normalized and mean normalized per utterance. The VGG had 13 convolutional layers and 5 fully connected layers. Batchnorm layers were inserted after every convolutional and fully connected layers except the last layer. We applied normalization for each frequency and feature map for batchnorm layers following convolutional layers. We inserted ReLU activation after every batchnorm layer. The VGG was trained with the training data by using the cross entropy and stochastic gradient descent with a manually tuned learning rate and decay schedule. A standard word n-gram model was used in common for all evaluation as the language model.

We tuned the learning rate and decay schedule so that vanilla adaptation achieved the best performance for AM domain adaptation. We applied the same learning rate and schedule that was tuned for the vanilla adaptation in the other experiments.

Table 1 shows the experimental results. "JP-1" column of Table 1 shows the character error rate (CER) of various adaptation methods. We used the CER as the evaluation metric because Japanese has ambiguous for word segmentation. Vanilla adaptation outperforms the baseline, but its gain was relatively small. When we froze the parameters of the batchnorm during adaptation, additional gain was obtained. We assumed that this gain was obtained because the mismatch between inference and training when the domain adaptation started was removed. However, it also removed the good effects of batchnorm. Using IIT enabled further gain compared to batchnorm to batchnorm freezing. We assumed that the gain was obtained by reducing the mismatch and leveraging batchnorm's good effects. Overall our proposed technique provides up to 5% relative improvement over the baseline system performance in this setting. Although AdaBN was shown to have a good effect for domain adaptation for some image classification tasks [14], we found it was worse than the baseline for this task.

5.2. Analysis

Fig. 2 shows the negative log likelihood of the validation data during supervised adaptation. The left most data points for both vanila adaptation and the proposed method are the same because IIT does not affect the inference mode. When adaptation started, vanilla adaptation suddenly increased in errors. We assumed that this was caused by the mismatch between inference and training. However, when IIT was applied, the loss was consistently reduced.

For quantitative evaluation of the mismatch between inference and training of batchnorm, we calculated the Kullback-Leibler (KL) divergence between distributions of batchnorm outputs in the inference mode and the training mode. Table 2 shows averages of the KL divergence between #3 – #4, #7 – #8

Table 1: *Results of AM domain adaptation with various methods, language, and data. Numbers in parenthesis mean WERs of intermediate models using first 10K training samples.*

Method	JP-1	US-1	US-2	US-3	SWB-dev04f
Baseline (#3 in Fig. 1)	42.2	26.4	19.5	19.9	14.7
Vanilla adaptation (#5 in Fig. 1)	41.5	26.3	18.7	21.0	(15.1) 14.0
Batchnorm freezing + adaptation	40.9	26.0	18.9	20.2	(14.2) 14.0
IIT + adaptation (#9 in Fig. 1)	40.1	26.0	18.8	19.9	(14.2) 14.0
AdaBN [14]	56.1	27.6	21.1	30.1	17.1

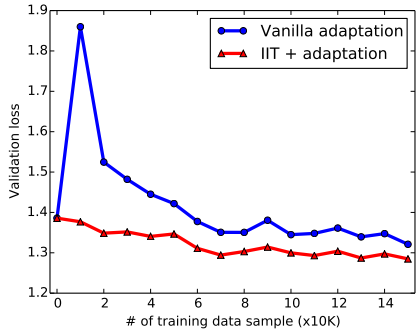

Figure 2: *Negative log likelihood curves of validation data. IIT is an acronym of inference invariant transformation. Vanilla adaptation and IIT + adaptation correspond to #4 and #8 in Fig. 1, respectively.*

Table 2: *Averages of KL divergence between distributions of batchnorm outputs.*

	#3 – #4	#7 – #8
First batchnorm	3.63	0.13
Middle (5th) batchnorm	26.90	0.35
Last (17th) batchnorm	15.46	0.13

in Fig. 1 of the first, middle (5th), and last (17th) batchnorm layers. The parameters of the preceding layer and batch normalization of #4 and #8 are frozen. As anticipated, the KL divergence between #3 and #4 is larger than that of between #7 and #8 for all batchnorm layers' output.

5.3. Unsupervised adaptation

To evaluate our proposed method's performance in an unsupervised setting also we conducted a second set of experiments using an English narrow band model. The training data for the base model was 2K hours of Switchboard and Callhome data and 500 hours of in-house call center data. We evaluate the performance of this model against three test sets, from three specific call centers. These 1 hour test sets were also used as unsupervised adaptation data. The language model in these experiments was a standard 4-gram. All other experimental settings are similar to those described in Section 5.1.

"US-1,2,3" columns in Table 1 show the word error rate (WER) of various test sets and various adaptation methods. As for US-1, it had a similar tendency as that of JP-1, except that batchnorm freezing performed comparably to our proposed method. For US-2, vanilla adaptation had the best performance, but the difference between its performance and that of our proposed method was small. For US-3, vanilla adaptation degraded performance from the baseline. However, our proposed method did not hurt the WER. In this setting too the proposed technique performs well and provides up to 4% relative improvement over the baseline system.

5.4. Unsupervised adaptation without significant domain mismatch

To further understand the usefulness of our proposed technique we conducted an unsupervised experiment with test data from a domain without any considerable mismatch to the training data. With sufficient amount of training data and minimal mismatch between train and test statistics, we hypothesize our proposed technique will only perform at par with other techniques.

The training data for the base model was 2K hours of Switchboard and Callhome data. We evaluated the performance of this model against downsampled version of the DARPA EARS dev04f set. The 2.2 hour dev04f set was also used as unsupervised adaptation data. All other experimental settings are similar to those described in Section 5.3. The SWB-dev04f column in Table 1 shows the WER of various adaptation methods. If we compare the performance of intermediate models using just the first 10K training samples, batchnorm freezing and IIT have better results than vanilla adaptation - see results in parenthesis. However, vanilla adaptation, batchnorm freezing, and the proposed method all have equal WERs when we run more iterations using all of the data. This experiment confirms our hypothesis that when the mismatch is minimal, with sufficient amount of adaptation data, vanilla adaptation can perform as well as our proposed technique. We observe up to 5% relative improvement over the baseline system performance in this setting as well.

In both unsupervised settings (experiment sets 2 and 3) we observe that the proposed technique has gains over the baseline system but the improvements are not as significant as in the supervised setting (experiment set 1). We also observe that the amount of adaptation data also plays an important role in the final performance of systems adapted in unsupervised settings.

6. Conclusion

In this paper we have proposed an adaptation strategy based on batchnorm, a popular technique to improve the training of deep neural networks. The adaptation strategy is an inference-invariant transformation of batchnorm and reduces the mismatch between inference and training when the domain adaptation strategy starts without changing the inference results. We have demonstrated the usefulness of the proposed approach in both supervised and unsupervised settings on various languages. We observe up to 5% relative improvement over the baseline system performance in all these various settings. In future work, we will conduct analysis to determine when our proposed method performs better than vanilla adaptation. We also plan to apply our proposed methods to the other transfer learning and domain adaptation tasks such as image classification.

7. References

[1] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[3] T. Sercu, C. Puhrsch, B. Kingsbury, and Y. LeCun, "Very deep multilingual convolutional neural networks for lvcsr," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4955–4959.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[5] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. C. Cui, B. Ramabhadran, M. Picheny, L.-L. Lim, B. R. Roomi, and P. Hall, "English conversational telephone speech recognition by humans and machines," *arXiv preprint arXiv:1703.02136*, 2017.

[6] M. Suzuki, R. Tachibana, S. Thomas, B. Ramabhadran, and G. Saon, "Domain adaptation of cnn based acoustic models under limited resource settings," *Interspeech 2016*, pp. 1588–1592, 2016.

[7] H. Liao, "Speaker adaptation of context dependent deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7947–7951.

[8] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, "Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7893–7897.

[9] J. Xue, J. Li, D. Yu, M. Seltzer, and Y. Gong, "Singular value decomposition based low-footprint speaker adaptation and personalization for deep neural network," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6359–6363.

[10] C. Liu, Y. Wang, K. Kumar, and Y. Gong, "Investigations on speaker adaptation of lstm rnn models for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5020–5024.

[11] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 806–813.

[12] J. Cui, B. Kingsbury, B. Ramabhadran, A. Sethy, K. Audhkhasi, X. Cui, E. Kislal, L. Mangu, M. Nussbaum-Thom, M. Picheny *et al.*, "Multilingual representations for low resource speech recognition and keyword search," in *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 2015, pp. 259–266.

[13] S. Ioffe, "Batch renormalization: Towards reducing minibatch dependence in batch-normalized models," *arXiv preprint arXiv:1702.03275*, 2017.

[14] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou, "Revisiting batch normalization for practical domain adaptation," *arXiv preprint arXiv:1603.04779*, 2016.