We thank all the reviewers for their insightful comments. We will polish the wording accordingly and will add additional references as suggested into the next version. We will also release a sample source code example demonstrating the training and computation of AFVs. Below we address the major concerns, and for comments we do not respond to explicitly we assume that we agree with the reviewer and will address properly in the revision.

1. Training protocol [**Reviewer #1**]: We apologize for the confusion in the training protocol. The simple answer is that the EBM view does not necessarily change the way the model is trained. As a matter of fact, in the majority of our experiments, we adopt the standard training procedure of GANs, except for cases where we explicitly test the effectiveness of the MCMC-inspired objective (see Sec 5.4).

A more detailed explanation is as follows. First, we agree that Equation 3 is not a concrete optimization procedure. However, it does indicate that, in order to train the EBM (D) with the variational trick, G needs to trained until convergence to tighten the lower bound on the NLL before updating D. This is in contrast to what is suggested by Equation 1 (following the convergence analysis of GAN theory), where D needs to be trained until convergence before updating G. In practice, we approximate both of the max and min optimize problems in Equation 1&3 with a few steps of mini-batch SGD, as is done in a standard GAN training procedure. As a result, the mini-batch SGD optimization algorithm can be interpreted as an approximation to either of the two objectives. We will add proper clarifications to the manuscript.

2. Scalability [**Reviewer #2,4**]: Scalability is a practical limitation of our work, because of two reasons. First, computing Fisher Vectors amounts to taking the gradient of each example w.r.t. all model parameters, which is hard to parallelize in modern deep learning frameworks. Second, the dimensionality of the induced Fisher Vectors is usually extremely high, posing a heavy memory demand on GPUs. We made a few initial attempts, one of which is to modify the training procedure such that we compute only the aggregated Fisher Vector representation for a mini-batch, and accordingly modify the ground-truth to be the averaged labels of the batch. This resembles an extreme version of mixup, and works reasonably well when batch size is small, but suffers from performance drop when batch size increases. We suspect that smarter ways of constructing the batch and averaging the labels might lead to further improvements. For the second fact, we also tried sampling the parameters to reduce the dimensionality of AFVs, but experienced minor performance drop in classification accuracy. We believe that the full solution to the scalability issue deserves an independent contribution and will leave it as future work.

3. Loss function [**Reviewer #2,4**]: Our default loss function is least square loss as in LSGAN, with sigmoid activation on the output of D, except the experiment in Sec 5.4 where we explicitly test the MCMC objective in a new setting. The reason for such a choice is that it provides the best numerical stability w.r.t. the outputs of D by preventing unnecessary shifts of D's outputs. We have also tested the hinge loss as done in, e.g., BigGAN, which works equally well w.r.t. the sampling quality and induced AFV representations, but weakened the smoothness of the Fisher Distance as a monitoring metric. We will add proper clarifications and discussions.

4. The MCMC objective [**Reviewer #2**]: Your understanding is correct: MCMC is never actually performed. We refer to MCMC because it offers an interpretation of the generator update as approximating one step MCMC. As a result, in practice we can directly adopt the standard G update rule assuming that each G update is small, mimicking an MCMC update. In cases where the local update of G is violated, it is useful to explicitly incorporate the proposed MCMC inspired objective in Equation 7 as a regularizer, as shown in Sec. 5.4. We will make this clear in the paper.

5. Additional baselines for the classification experiment [**Reviewer #2,4**]: Per Rev 4's request, during the rebuttal period we tested the supervised learning performance using the discriminator architecture, by changing the output dimension of the last layer to 10. With only this change, the supervised learning test accuracy is 86.1%, which is worse than our AFV + SVM's 89.1%. We then replaced all the Spectral Normalization layers with Batch Normalization and repeated the experiment, and got a 92.7% accuracy, which exceeds our AFV result. We additionally have conducted the same experiment on CIFAR100, where AFV+SVM achieves a test accuracy of 67.8%, compared to the supervised training (with BN) performance 70.3%. Note that 67.8% is also the best result we can find under the pretraining + linear classifier training setup on CIFAR100. For example, the Deep InfoMax paper reports an accuracy of 49.74%, which is significantly worse than our result. We will report these experimental results in the paper.

6. Approximating Equation 4 [**Reviewer #4**]: The expectation term in Equation 4 is w.r.t. the model distribution p_θ. It is most natural to use the generated samples to approximate it because, according to the EBM view, the generator is exactly trying to match the model distribution. Empirically, we also found that using the generated examples works slightly better than using real examples, but the margin is small. We will clarify in the manuscript.