Interpolating Latent Representations to Increase Label Noise Tolerance

Mason Wang,[1†] Yi Ding[2]
[1]*Saratoga High School, 20300 Herriman Ave, Saratoga, CA 95070*
[2]*Department of Computer Science, University of California, Santa Barbara, CA 93106*
[†]*corresponding author: masonwang025@gmail.com*

The success of deep learning relies heavily on the availability of large, annotated datasets. However, label noise is a universal problem for real-world datasets because of practical challenges in gathering cleanly, manually annotated data. Large deep learning models are so powerful that they are capable of memorizing noisy labels, which negatively impacts model performance. Although many label noise tolerant techniques have been developed for unimodal image datasets, little research on label noise techniques in other modalities and multimodal settings exists. In this paper, we propose the use of a modality-agnostic label noise tolerant technique involving training on interpolated latent representations along with their labels, encouraging neural networks to predict less confidently on these in-between examples. Our technique reduces the memorization of corrupt labels, which can alleviate the issue of noisy labels. We also explore the significance of selecting latent representations at deeper layers for interpolation.

Keywords: learning with noisy labels, regularization, label noise, feature embeddings

I. INTRODUCTION

Deep neural networks (DNNs) have seen widespread use for machine learning applications because of their capability to model complicated patterns [1]. The remarkable success of deep learning relies heavily on the availability of large, labeled datasets, which are often hard to obtain due to practical challenges. Instead, large datasets are often gathered via crowdsourcing, using non-expert labelers, or crawled from the internet, using implied labels (e.g., assuming the keyword is an accurate label when querying search engines for instances of the keyword). Both approaches result in noisy samples, which are samples with incorrect labels [2].

Label noise is a particularly important issue because large DNNs have the capacity to memorize corrupt labels and can even memorize entire datasets with all wrongly labeled examples relatively easily [3]. DNNs that overfit to noisy data fail to generalize accurately to real-world cases, thus performing extremely poorly [4]. Algorithms capable of learning from noisy datasets without such overfitting are also inherently valuable as they can significantly decrease the costs of gathering training data [2].

Aiming to improve generalization, many existing methods for learning with noisy labels (LNL) take a regularization-based approach [5], [6]. A particularly successful method is *mixup*, a simple learning principle that involves training networks on linear combinations of pairs of samples and their labels. This effectively encourages DNNs to predict less confidently on the in-between training samples [7]. *mixup* has shown considerable success in improving algorithms' label noise tolerance [8], [9].

The majority of LNL methods, including *mixup*, are strictly unimodal and typically designed for only image classification tasks. Multimodal models, unlike unimodal ones, utilize information from multiple streams of information such as acoustic, visual, and language modalities. Their ability to use cross-modal interactions is an important advantage in machine learning tasks [10]. This is especially true for inherently multimodal data, such as human interaction, which consists of facial gestures, natural language, and acoustic behaviors [11].

Despite label noise being a prevalent issue in multimodal datasets, as indicated by unacceptably low agreement scores across annotators in multimodal datasets [10], current techniques do not address LNL in a multimodal setting. The development of multimodal LNL methods could lower annotation costs, and, more importantly, allow for even more powerful multimodal models.

In light of this issue and *mixup*'s success, we seek to develop an LNL technique for multimodal settings by addressing the limitations of existing work. *mixup*'s current implementations may struggle to extend beyond image data, as combining raw input in other modalities in a meaningful way is often not possible (e.g., combining sentences and text). Thus, we are motivated to instead interpolate latent representations—which are learned by the neural network—of training examples rather than the raw

input. Manifold Mixup, a simple regularizer based on *mixup*, first introduces this technique and shows that it can improve hidden representations and decision boundaries of neural networks at multiple layers [12].

However, Manifold Mixup does not investigate this technique's potential for LNL, so its effects on even unimodal datasets with label noise is unknown. Manifold Mixup also does not test its use in multimodal settings or on data of non-image modalities. There is also very limited understanding of the internal hidden representations of neural networks [13], so we are interested in the effects of choosing to include or exclude layer representations from being selected for *mixup*.

In this paper, we propose learning with interpolated latent representations as a modality-agnostic label noise tolerant technique. Our key contributions are as follows:

1. We empirically show that training on interpolated latent representations increases label noise tolerance by applying it in an existing LNL approach.
2. We find that selecting deeper layers for hidden state interpolation results in better performance and faster convergence when there are low levels of label noise present. For high label noise levels, there is only an initial increase in performance before resulting in overfitting to the noisy labels.

II. METHODOLOGY

To show the effectiveness of our proposed technique of training on interpolated latent representations for learning with label noise, we i) show that it successfully increases noise tolerance, ii) analyze the use of certain layers for hidden state interpolation.

A. Latent Representation Interpolation

We rely heavily on Manifold Mixup's implementation of training on hidden states. The theory behind Manifold Mixup is detailed in [12], which applies *mixup* on the hidden states, or latent representations, rather than on the raw input itself. Manifold Mixup trains neural networks on linear interpolations of pairs of hidden states and their labels. Hidden states consist of features learned by the neural networks and thus are also referred to as feature embeddings.

Figure 1 depicts training a neural network with Manifold Mixup and the *mixup* equation. The authors of [12] show that training on interpolated feature embeddings improves both the hidden representations and decision boundaries of neural networks at multiple layers.

The process of training a deep neural network with Manifold Mixup is as follows (for details, see [12]):

1. Select a random layer k from a set of eligible layers S.
2. Process two random minibatches (x, y) and (x', y') as usual until layer k. This results in two intermediate minibatches $(g_k(x), y)$ and $(g_k(x'), y')$, where g_k denotes the part of the neural network mapping the input data to the hidden representation at layer k.
3. Perform *mixup* on these intermediate minibatches, producing the mixed minibatch $(\tilde{g}_k', \tilde{y}')$. See Figure 1 for more details.
4. Continue the forward pass in the network from layer k until the output using the mixed minibatch $(\tilde{g}_k', \tilde{y}')$.
5. Use the output to compute the loss value and gradients to update all the parameters of the neural network.

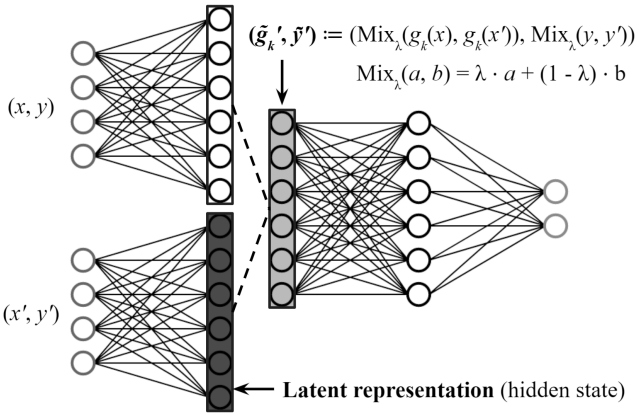

FIG. 1. Training a deep neural network $f(x) = f_k(g_k(x))$ with Manifold Mixup. Two mini-batches (x, y) and (x', y') are processed as usual until reaching layer k. *mixup* is performed on the hidden states (latent representations) of layer k [12].

B. Label Noise Injection

To evaluate our technique for LNL, we train our models on synthetically noisy datasets. For label noise addition, we randomly re-assign labels for a certain percentage of the training data using all possible labels (i.e., the true label could be maintained after the re-assignment).

C. Latent Representation Mixup for LNL

We first set out to empirically show that training on interpolated latent representations succeeds as a label noise tolerant technique. To do so, we implement this technique in a successful LNL approach proposed in [8] and test it on synthetically noisy datasets.

Existing LNL methods are based on the loss correction approach [14], such as the bootstrapping loss, which introduces a term in the loss function that assigns a weight to the current prediction to compensate for the erroneous guiding of noisy samples [15].

The authors of [8] propose a loss correction approach that uses an unsupervised label noise model to correct each sample loss, thus preventing overfitting to label noise. The label noise model relies heavily on the essential observation that random labels take longer to learn than clean labels. This approach is combined with *mixup* data augmentation to outperform the state-of-the-art LNL methods.

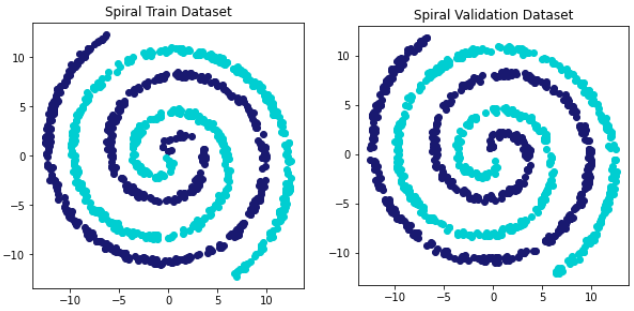

FIG. 2. The train and validation sets for a generated two-spiral dataset, each with 1,000 samples.

Instead of applying *mixup* on the input, we apply it on learned feature embeddings and compare its performance on noisy datasets with other approaches covered in [8]. For every batch, we randomly select a hidden representation at a layer between layers 0-2 (layer 0 is the input layer, and bounds are inclusive).

We validate our approach on CIFAR-10, a well-known image classification dataset with 50,000 32x32 color images for training and 10,000 images for validation. We train a PreAct ResNet-18 in PyTorch from scratch using SGD and use dynamic hard bootstrapping, as it achieved the best performance in [8]. Experiment details are listed in Appendix A.

D. Interpolating Deeper Latent Representations for LNL

Because it is often not possible to meaningfully combine the raw input for most non-image modalities, we further evaluate our approach using *mixup* only on layers 1-4, leaving the rest of the parameters of our previous experiments the same.

To understand the effects of using deeper feature embeddings, we also conduct the same experiment but with selecting from layers 0-4 instead of only 0-2.

We further analyze these effects by running a separate set of experiments on the two-spiral dataset shown in Figure 2. We train a simple neural network on all possible ranges of *mixup* layers. There are at least two layers in each range, with the exception of input *mixup* (only layer 0, the input layer), because it is generally difficult to know the single best layer for *mixup* in more complicated models and tasks.

For each sample, along with the x-coordinate and y-coordinate, we also provide spherical features by transforming the coordinates to spherical coordinates. This allows for our neural network to fit to the spiral data more accurately.

We choose to not include the bootstrapping loss technique used in our previous experiments, as label noise modeling does not work well on the simple binary classification dataset.

Details of our experiment and hyperparameters are listed in Appendix B.

Alg./Noise		0%	20%	50%	80%
CE [8]	Best	94.7	86.8	79.8	63.3
	Last	94.6	82.9	58.4	26.3
M [7]	Best	**95.3**	**95.6**	87.1	71.6
	Last	**95.2**	92.3	77.6	46.7
M-DYR-S [8]	Best	93.3	93.5	89.7	77.3
	Last	93.0	93.1	89.3	74.1
M-DYR-H [8]	Best	93.6	94.0	92.0	86.8
	Last	93.4	93.8	**91.9**	**86.6**
LR-M-DYR-H	Best	94.5	95.1	**92.1**	**87.5**
	Last	84.6	**94.7**	91.5	86.5

TABLE 1. Validation accuracy on CIFAR-10 for joint *mixup* and bootstrapping. Key: CE (cross-entropy), M (*mixup*), DYR (dynamic bootstrapping + regularization), S (soft), H (hard), and LR (latent representation). Bold indicates best performance.

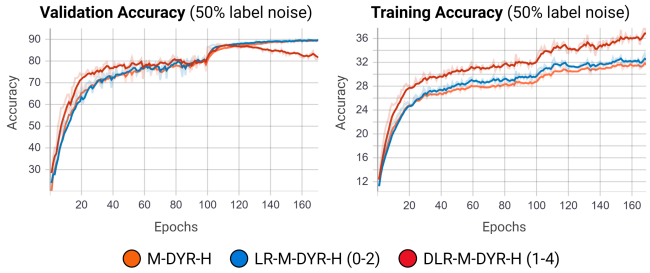

FIG. 3. Validation and training accuracy of model trained on a 50% noisy CIFAR-10 dataset. Shown models use hard bootstrapping (DYR-H) with *mixup* (M), latent representation (LR) *mixup* on layers 0-2, and deep latent representation (DLR) *mixup* on layers 1-4.

III. RESULTS AND DISCUSSION

Table 1 presents the results of our first set of experiments with latent representation (LR) mixup (M). We find that interpolating on feature embeddings instead of the raw input results in a higher best performance across all noise levels. It also achieves the highest best performance out of all algorithms for datasets with 50% and 80% label noise. The overall results demonstrate that our technique can alleviate the issue of label noise.

Figure 3 shows an additional experiment using DLR-M, which excludes layer 0 from mixup and includes deeper layers 3 and 4. We find that although the model initially performs far better and converges faster, the change in eligible layers for *mixup* results in overfitting to label noise. Figure 4 and Figure 5 shows the use of LR-M with layers 0-4 across noise label levels 0%, 20%, and 80%. Figure 4 shows that when there exists low or no label noise, the

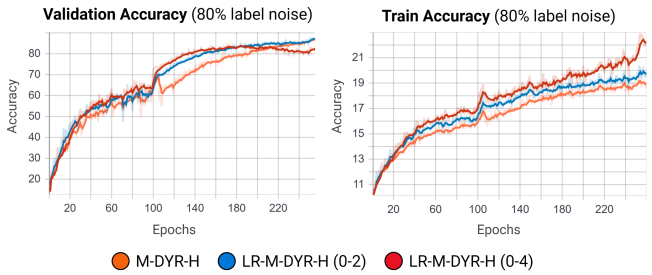

FIG. 4. Validation and training accuracy of a model trained on a 80% noisy CIFAR-10 dataset. Shown models use hard bootstrapping (DYR-H) with *mixup* (M), latent representation (LR) *mixup* on layers 0-2, and latent representation (LR) *mixup* on layers 0-4.

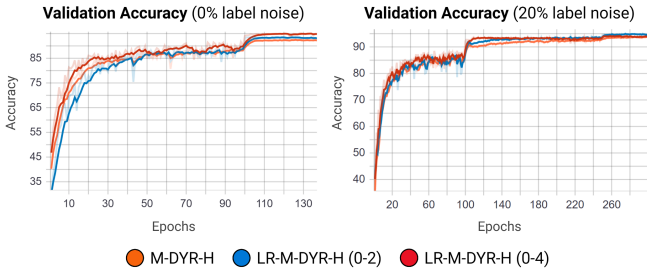

FIG. 5. Validation accuracy on 0% and 20% noisy CIFAR-10 datasets. Shown models use hard bootstrapping (DYR-H) with *mixup* (M), latent representation (LR) *mixup* on layers 0-2, and latent representation (LR) *mixup* on layers 0-4.

inclusion of the deeper layers for *mixup* results in faster convergence and better overall performance. On 0% label noise, our model achieves the highest accuracy (95.4%) after just 138 epochs when compared to all other models (trained for 300 epochs) in Table 1.

This set of additional experiments suggests that the inclusion of the deeper embeddings for *mixup* results in better initial performance for all label noise levels, but also results in overfitting to noisy labels when there are high amounts of label noise.

Table 2 presents the results of our set of experiments on the two-spiral dataset. There is a clear trend of higher accuracy when interpolating deeper representations, and this is consistent across all tested noise levels.

For four out of five of the noise levels, the highest performance is achieved by a model that includes the deepest layer for interpolation. Furthermore, for each group of models that have the same lower *mixup* layer bound, the highest performing one typically includes the deeper layers. This further supports our previous findings, as these spiral datasets contain low amounts of label noise (from 0% to 40% label noise).

Across all label noise levels, the best models which do not include the input layer as a possible *mixup* layer outperform the best models which do, showing that our technique has high potential in other modalities and

Alg./Noise	0%	10%	20%	30%	40%
CE	93.7	90.1	93.6	91.0	91.3
M (0-0)	77.9	78.6	75.1	75.6	72.0
LR-M (0-1)	89.8	86.9	83.9	86.0	86.1
LR-M (0-2)	93.2	90.9	87.1	84.9	87.8
LR-M (0-3)	93.9	89.3	91.6	**88.8**	87.5
LR-M (0-4)	**96.0**	91.9	**93.1**	88.7	**88.2**
LR-M (0-5)	95.1	**93.0**	90.4	88.2	87.3
LR-M (1-2)	96.4	95.4	91.9	89.7	88.7
LR-M (1-3)	96.5	**96.5**	92.0	90.7	88.3
LR-M (1-4)	**96.7**	95.5	93.3	90.3	88.8
LR-M (1-5)	96.6	96.2	**97.6**	**91.5**	**89.0**
LR-M (2-3)	98.4	**97.3**	**93.9**	93.3	90.2
LR-M (2-4)	**96.9**	96.8	93.7	90.8	**91.3**
LR-M (2-5)	96.3	**97.3**	93.3	**94.1**	90.9
LR-M (3-4)	**98.1**	**98.7**	94.0	**93.5**	90.7
LR-M (3-5)	**98.1**	96.8	**94.8**	92.2	**91.1**
LR-M (4-5)	**96.6**	**97.8**	**94.1**	**91.9**	**92.0**

TABLE 2. Best validation accuracies on spiral dataset. Eligible layers for *mixup* in parentheses (inclusive). Key: CE (cross-entropy), M (*mixup*), LR (latent representation). Bold indicates best performance for models with the same lower *mixup* bound. Underline indicates best performance for its noise level.

multimodal settings, where meaningful interpolation of the raw inputs is often not possible.

IV. CONCLUSIONS

In this paper, we propose training on interpolated latent representations as a modality-agnostic label noise tolerant technique. We examine its effectiveness in an existing LNL approach and find it is successful as an LNL method. Next, through testing all different ranges of layers to perform *mixup* on, we find that interpolating on deeper hidden layers consistently increases performance for datasets with low amounts of label noise. It also allows for improvements in initial performance and convergence for datasets with high amounts of label noise, but ultimately overfits to noisy labels.

These findings show that training on interpolated feature embeddings is very promising as a multimodal label noise tolerant technique. Future work primarily involves i) understanding why using deeper layers for *mixup* results in overfitting to label noise, ii) testing our technique in other modalities to ensure it truly is modality-agnostic, and, most importantly, iii) investigating its use in alleviating multimodal label noise.

ACKNOWLEDGEMENTS

This work is affiliated with UC Santa Barabara's Research Mentorship Program. First author would like to thank Raphael Ruschel dos Santos for his guidance in creating this paper. First author would also like to acknowledge Tobias Höllerer, Pradeep Sen, and the rest of the FourEyes Lab at UCSB for all of their feedback.

REFERENCES

[1] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to Reweight Examples for Robust Deep Learning," in *International Conference on Machine Learning*, Jul. 2018, pp. 4334–4343. Accessed: Jul. 10, 2021. [Online]. Available: http://proceedings.mlr.press/v80/ren18a.html

[2] G. Patrini, A. Rozza, A. K. Menon, R. Nock, and L. Qu, "Making Deep Neural Networks Robust to Label Noise: A Loss Correction Approach," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, Jul. 2017, pp. 2233–2241. doi: 10.1109/CVPR.2017.240.

[3] Y. Lu and J. Lu, "A Universal Approximation Theorem of Deep Neural Networks for Expressing Probability Distributions," *arXiv:2004.08867 [cs, math, stat]*, Nov. 2020, Accessed: Jul. 11, 2021. [Online]. Available: http://arxiv.org/abs/2004.08867

[4] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Commun. ACM*, vol. 64, no. 3, pp. 107–115, Feb. 2021, doi: 10.1145/3446776.

[5] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa, "Joint Optimization Framework for Learning with Noisy Labels," *arXiv:1803.11364 [cs, stat]*, Mar. 2018, Accessed: Jul. 11, 2021. [Online]. Available: http://arxiv.org/abs/1803.11364

[6] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, Dec. 2017, pp. 1195–1204.

[7] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization," *arXiv:1710.09412 [cs, stat]*, Apr. 2018, Accessed: Jul. 11, 2021. [Online]. Available: http://arxiv.org/abs/1710.09412

[8] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Unsupervised Label Noise Modeling and Loss Correction," *arXiv:1904.11238 [cs]*, Jun. 2019, Accessed: Jul. 11, 2021. [Online]. Available: http://arxiv.org/abs/1904.11238

[9] J. Li, R. Socher, and S. C. H. Hoi, "DivideMix: Learning with Noisy Labels as Semi-supervised Learning," *arXiv:2002.07394 [cs]*, Feb. 2020, Accessed: Jul. 11, 2021. [Online]. Available: http://arxiv.org/abs/2002.07394

[10] P. P. Liang and R. Salakhutdinov, "Computational Modeling of Human Multimodal Language: The MOSEI Dataset and Interpretable Dynamic Fusion," 2018. https://www.semanticscholar.org/paper/Computational-Modeling-of-Human-Multimodal-The-and-Liang-Salakhutdinov/90ffde2da9f855f96deb025247a860ede90a80dc (accessed Jul. 11, 2021).

[11] Y.-H. H. Tsai, S. Bai, P. P. Liang, J. Z. Kolter, L.-P. Morency, and R. Salakhutdinov, "Multimodal Transformer for Unaligned Multimodal Language Sequences," *arXiv:1906.00295 [cs]*, Jun. 2019, Accessed: Jul. 11, 2021. [Online]. Available: http://arxiv.org/abs/1906.00295

[12] V. Verma *et al.*, "Manifold Mixup: Better Representations by Interpolating Hidden States," *arXiv:1806.05236 [cs, stat]*, May 2019, Accessed: Jul. 11, 2021. [Online]. Available: http://arxiv.org/abs/1806.05236

[13] A. Ghosh and D. Kandasamy, "Interpretable Artificial Intelligence: Why and When," *American Journal of Roentgenology*, vol. 214, no. 5, pp. 1137–1138, May 2020, doi: 10.2214/AJR.19.22145.

[14] D. Hendrycks, M. Mazeika, D. Wilson, and K. Gimpel, "Using trusted data to train deep networks on labels corrupted by severe noise," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, Dec. 2018, pp. 10477–10486.

[15] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training Deep Neural Networks on Noisy Labels with Bootstrapping," *arXiv:1412.6596 [cs]*, Apr. 2015, Accessed: Jul. 18, 2021. [Online]. Available: http://arxiv.org/abs/1412.6596

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Identity Mappings in Deep Residual Networks," Oct. 2016, vol. 9908, pp. 630–645. doi: 10.1007/978-3-319-46493-0_38.

APPENDIX

A. LNL Approach Experiment Details

We use the same hyperparameters and training process outlined in [8]. Specifically, we train a PreAct ResNet-18 [16] in PyTorch from scratch using SGD with momentum of 0.9, weight decay of 10^{-4}, and batch size of 128.

For preprocessing, images are normalized and augmented by random horizontal flipping. We use 32x32 random crops after zero padding with 4 pixels on each side.

We train for 300 epochs and reduce the initial learning rate of 0.1 by a factor of 10 after 100 and 250 epochs.

Bootstrapping starts after 105 epochs of warm-up, and we use dynamic hard bootstrapping, as it achieved the best performance in [8].

B. Spiral Dataset Experiment Details

We train a simple neural network (hyperparameters listed below) in PyTorch from scratch using SGD with momentum of 0.9 and a batch size of 24. We train for 300 epochs and reduce the initial learning rate of 0.03 by a factor of 10 after 100 and 250 epochs.

Each layer of our neural network is a Linear layer (applies linear transformation $y = xA^T + b$) followed by a Tanh activation function. The numbers of input and output features for each Linear layer are as follows:

1. 4 input features, 16 output features
2. 16 input features, 24 output features
3. 24 input features, 24 output features
4. 24 input features, 24 output features
5. 24 input features, 16 output features
6. 16 input features, 2 output features

Layer 6 was not included as an eligible layer for *mixup* in any of our experiments as it is the output layer.