Towards Robustness to Label Noise in Text Classification via Noise Modeling

Siddhant Garg[*][†]
Amazon Alexa AI Search
Manhattan Beach, CA, USA
sidgarg@amazon.com

Goutham Ramakrishnan[*][†]
Health at Scale Corporation
San Jose, CA, USA
goutham7r@gmail.com

Varun Thumbe [*][†]
KLA Corporation
Ann Arbor, MI , USA
thumbevarun@gmail.com

Abstract

Large datasets in NLP suffer from noisy labels, due to erroneous annotation procedures. We study the problem of text classification with label noise, and aim to capture this noise through an auxiliary noise model over the classifier. We first assign a probability score to each training sample of having a noisy label, through a beta mixture model fitted on the losses at an early epoch of training. Then, we use this score to selectively guide the learning of the noise model and classifier. Our empirical evaluation on two text classification tasks shows that our approach can improve over the baseline accuracy, and prevent over-fitting to the noise.

1 Introduction

Training modern ML models requires access to large accurately labeled datasets, which are difficult to obtain due to errors in automatic or human annotation techniques Wang et al. (2018); Zlateski et al. (2018). Recent studies Zhang et al. (2016) have shown that neural models like CNNs can over-fit on noisy labels and thereby not generalize well.

Human annotations for language tasks have been popularly drawn through crowd sourcing platforms like Amazon Mechanical Turk Ipeirotis et al. (2010). The label annotations may be noisy due to a number of reasons: ambiguity of the correct label Zhan et al. (2019), annotation speed, human error, inexperience of annotator, etc. While learning with noisy labels has been extensively studied in computer vision Reed et al. (2015); Zhang et al. (2018); Thulasidasan et al. (2019), the corresponding progress in NLP has been rather limited. With the increasing size of modern NLP datasets, the problem of noisy labels is likely to affect several practical applications Agarwal et al. (2007).

In this paper, we consider the problem of text classification, and capture the label noise through an auxiliary noise model over the classifier model (See Fig. 1). We leverage the finding of learning on clean labels being easier than on noisy labels Arazo et al. (2019), and first fit a 2-component beta-mixture model(BMM) on the losses of the training samples at an early stage of training. Using this, we assign a probability score to every training sample of having a clean or noisy label. Then, using these scores we jointly train the classifier and the noise model by selectively guiding the former's prediction for samples with low scores of being noisy. Our formulation constrains the label noise within the noise model, and drives the classifier to learn from the clean training samples.

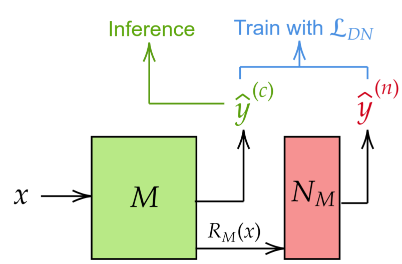
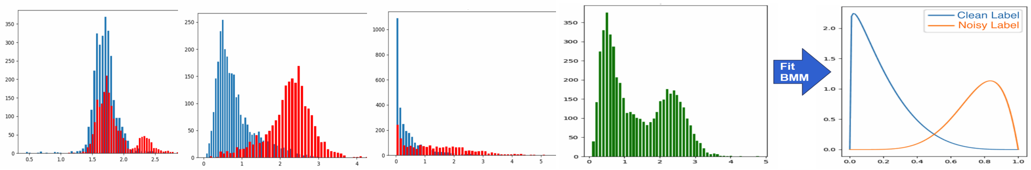


Figure 1: In our approach, we jointly train an auxiliary noise model N_M on top of the classifier M using a *de-noising* loss $\mathcal{L}_{DN}$, and use the clean label prediction $\hat{y}^{(c)}$ during inference.

Existing works on learning with noisy labels make a simplifying assumption that the label noise is independent of the input and only conditional on the true label. Text annotation complexity has been shown to depend on the lexical, syntactic and semantic input features Joshi et al. (2014) and not be conditional solely on the true label. The noise model in our formulation can capture an arbitrary noise function, which may depend on both the input and the original label, taking as

[*] Equal contribution by authors
[†] Work completed as graduate students at the University of Wisconsin–Madison



(a) Epoch 1 (b) Epoch 9 (c) Epoch 30 (d) Fitting a BMM at Epoch 9

Figure 2: (a), (b) and (c) show the histogram of training losses (Clean, Noisy) at different epochs (word-LSTM on TREC with 40% random noise). (d) shows the fitting of a beta-mixture model.

input a contextualized input representation from the classifier. While de-noising the classifier for sophisticated noise functions is a challenging problem, we take the first step towards capturing a real world setting.

We evaluate our approach on two popular text classification datasets, at different levels of random and input-conditional noise. Across two model architectures, our approach shows improvement over the baseline of learning directly with the noisy labels. We show that our de-noising approach prevents the model from over-fitting the label noise.

2 Related Work

Several other research works have studied the problem of combating label noise in computer vision Frénay & Verleysen (2014); Jiang et al. (2018; 2019) through techniques like bootstrapping Reed et al. (2015), mixup Zhang et al. (2018), etc. In NLP, Ardehaly & Culotta (2018) study social media text classification using label proportion (LLP) models, and Malik & Bhardwaj (2011) automatically validate noisy labels using high-quality class labels. Jindal et al. (2019) capture random label noise via a ℓ_2-regularized matrix learned on the classifier logits. Our work differs from this as we i) use a neural network noise model over contextualized embeddings from the classifier, with (ii) a new de-noising loss to explicitly guide learning. It is difficult to draw a distinction between noisy labels, and outliers which are hard to learn from. While several works perform outlier detection Goodman et al. (2016); Larson et al. (2019) to discard these samples while learning the classifier, we utilise the noisy data in addition to the clean data for improving performance.

3 Methodology

Problem Setting Let $(X, Y^{(c)}){=}\{(x_1, y_1^{(c)}), \dots, (x_N, y_N^{(c)})\}$ denote clean training samples from a distribution $\mathcal{D}{=}\mathcal{X}{\times}\mathcal{Y}$. We assume a function $\mathcal{F} : \mathcal{X}{\times}\mathcal{Y} \rightarrow \mathcal{Y}$ that introduces noise in labels $Y^{(c)}$. We apply $\mathcal{F}$ on $(X, Y^{(c)})$ to obtain the noisy training data $(X, Y^{(n)}){=}\{(x_1, y_1^{(n)}), \dots, (x_N, y_N^{(n)})\}$. $(X, Y^{(n)})$ contains a combination of clean samples (whose original label is retained $y^{(n)}{=}y^{(c)}$) and noisy samples (whose original label is corrupted $y^{(n)}{\neq}y^{(c)}$). Let (X_T, Y_T) be a test set sampled from the clean distribution $\mathcal{D}$. Our goal is to learn a classifier model $\mathcal{M} : \mathcal{X}{\rightarrow}\mathcal{Y}$ trained on the noisy data $(X, Y^{(n)})$, which generalizes well on (X_T, Y_T). Note that we do not have access to the clean labels $Y^{(c)}$ at any point during training.

Modeling Noise Function $\mathcal{F}$ We propose to capture $\mathcal{F}$ using an auxiliary *noise model* N_M on top of the classifier model M, as shown in Fig. 1. For an input x, a representation $R_M(x)$, derived from M, is fed to N_M. $R_M(x)$ can typically be the contextualized input embedding from the penultimate layer of M. We denote the predictions from M and N_M to be $\hat{y}^{(c)}$(clean prediction) and $\hat{y}^{(n)}$(noisy prediction) respectively. The clean prediction $\hat{y}^{(c)}$ is used for inference.

Unsupervised Learning of Clean Samples It has been empirically observed that classifiers that capture input semantics do not fit the noise before significantly learning from the clean samples Arazo et al. (2019). For a classifier trained using a cross entropy loss($\mathcal{L}_{CE}$) on the noisy dataset, this can be exploited to cluster the input samples as being clean/noisy in an unsupervised manner. Initially the training loss on both clean and noisy samples is large, and after a few training epochs, the loss of majority of the clean samples reduces. Since the loss of the noisy samples is still large, this segregates the samples into two clusters with different loss values. On further training, the model over-fits on the noisy samples and the training loss on both samples reduces. We illustrate this in Fig. 2(a)$-$(c). We cluster the training losses after a few training epochs using a two-component beta-mixture model $\mathcal{B}$ in line with Arazo et al. (2019). Thus for an input x, we denote $\mathcal{B}(x)$ to be the posterior BMM probability that x has a clean label. $\mathcal{B}$ learnt from Fig. 2b is shown in Fig. 2d. We describe the BMM more formally with details in the appendix.

Learning M, N_M For a clean input (x, y), we want $\hat{y}^{(c)}{=}\hat{y}^{(n)}{=}y$ and for a noisy input (x, y), we want $\hat{y}^{(n)}{=}y$ and $\hat{y}^{(c)}$ to be the clean label for x. We jointly train M, N_M using the de-noising loss:

$$\mathcal{L}_{DN}{=}\ \mathcal{L}_{CE}(\hat{y}^{(n)}, y)\ {+}\beta{\cdot}\mathcal{B}(x){\cdot}\mathcal{L}_{CE}(\hat{y}^{(c)}, y) \tag{1}$$

The first term trains the $M{-}N_M$ cascade jointly using a cross entropy loss between $\hat{y}^{(n)}$ and y. The second term trains M to predict $\hat{y}^{(c)}$ correctly for samples believed to be clean, weighted by $\mathcal{B}(x)$. Here β is a weighting parameter that controls the trade-off between the two terms.

By jointly training M and N_M with $\mathcal{L}_{DN}$, we implicitly constrain the label noise in N_M. We use an alternative formulation for $\mathcal{L}_{DN}$ by replacing the Bernoulli R.V. $\mathcal{B}(x)$ with the indicator $\mathbb{1}[\mathcal{B}(x){>}0.5]$. For ease of notation, we refer the former (using $\mathcal{B}(x)$) as the soft de-noising loss $\mathcal{L}_{DN-S}$ and the latter as the hard de-noising loss $\mathcal{L}_{DN-H}$. Thus we use the following 3-step approach to learn M and N_M:

1. **Warmup:** Train M using $\mathcal{L}_{CE}(\hat{y}^{(c)}, y)$.
2. **Fitting BMM:** Fit a 2-component BMM $\mathcal{B}$ on the $\mathcal{L}_{CE}(\hat{y}^{(c)}, y)$ for all $(x, y) \in (X, Y^{(n)})$.
3. **Training with $\mathcal{L}_{DN}$:** Jointly train M and N_M end-to-end using $\mathcal{L}_{DN-S/H}$.

We formally summarize our methodology in Algorithm 1, when using the $\mathcal{L}_{DN-H}$ loss.

Algorithm 1 Training using $\mathcal{L}_{DN-H}$

Input: Train data $(x_i, y_i^{(n)})_{i=1}^N$, warmup epochs T_0, total epochs T, parameter β, classifier M, noise model N_M
for epoch in $\{1, \ldots, T_0\}$ **do**
  $\hat{y_i}^{(c)} \leftarrow M(x_i)\ \forall\, i \in [N]$
  Train M with $\sum_i \mathcal{L}_{CE}(\hat{y_i}^{(c)}, y_i^{(n)})$
end for
Fit a 2-mixture BMM $\mathcal{B}$ on $\{\mathcal{L}_{CE}(\hat{y_i}^{(c)}, y_i^{(n)})\}_{i=1}^N$
for epoch in $\{T_0 + 1, \ldots, T\}$ **do**
  $\hat{y_i}^{(c)} \leftarrow M(x_i)$,
  $\hat{y_i}^{(n)} \leftarrow N_M(R_M(x_i))\ \ \forall\, i \in [N]$
  Train M, N_M with $\mathcal{L}_{DN-H} = \sum_i \big(\mathcal{L}_{CE}(\hat{y_i}^{(n)}, y_i^{(n)}) + \beta \cdot \mathbb{1}[\mathcal{B}(x){>}0.5] \cdot \mathcal{L}_{CE}(\hat{y_i}^{(c)}, y_i^{(n)})\big)$
end for
Return: Trained classifier model M

4 Evaluation

Datasets We experiment with two popular text classification datasets: (i) TREC question-type dataset Li & Roth (2002), and (ii) AG-News dataset Gulli (2005). We inject noise in the training and validation sets, while retaining the original clean test set for evaluation. Note that collecting real datasets with known patterns of label noise is a challenging task, and out of the scope of this work. We artificially inject noise in clean datasets, which enables easy and extensive experimentation.

Models [1] We use 2 architectures for model M: word-LSTM Hochreiter & Schmidhuber (1997) and word-CNN Kim (2014). For the noise model N_M, we use a 2-layer fully connected NN.

Metrics and Baseline We evaluate the robustness of the model to label noise on two fronts: (i) How well it performs on clean data, and (ii) How much it over-fits the noisy data. For the former, we report the test set accuracy(denoted by *Best*) corresponding to the model with best validation accuracy . For the latter, we examine the gap in test accuracies between the *Best*, and the *Last* model (after last training epoch). We evaluate our approach against only training M, for two types of noise: random and input-conditional, at different noise levels.

Random Noise For a specific Noise %, we randomly change the original labels of this percentage of samples. Since the noise function is independent of the input, we use logits from M as the input $R(x)$ to N_M. We report the *Best* and (*Best* - *Last*) test accuracies in Table 1. We observe that:

(i) L_{DN-S} and L_{DN-H} almost always outperforms the baseline across different noise levels. The performance of L_{DN-S} and L_{DN-H} are similar. We observe that training with L_{DN-S} tends to be better at low noise %, whereas L_{DN-H} tends to be better at higher noise %. Our method is more effective for TREC than AG-News, since even the baseline can learn robustly on AG-News.

(ii) Our approach using L_{DN-S} and L_{DN-H} drastically reduces over-fitting on noisy samples (visible from small gaps between *Best* and *Last* accuracies). For the baseline, this gap is significantly larger, especially at high noise levels, indicating over-fitting to the label noise. For example, consider the word-LSTM model on TREC at 30% noise: while the baseline suffers a sharp drop of 24.8 points from 79.6%, the accuracy of the L_{DN-S} model drops just 1.0% from 83.4%.

[1] Code can be found at `https://github.com/thumbe3/label-noise-nlp`.

Model	Noise %	TREC (word-LSTM: 93.8, word-CNN: 92.6)					AG-News (word-LSTM: 92.5, word-CNN: 91.5)				
		10	20	30	40	50	10	20	30	40	50
word LSTM	Baseline	88.0 (-0.6)	89.4 (-9.6)	83.4 (-19.0)	79.6 (-24.8)	77.6 (-27.2)	**91.9** (-1.7)	**91.3** (-1.5)	90.5 (-2.5)	89.3 (-3.7)	88.6 (-10.5)
	$\mathcal{L}_{DN-H}$	92.2 (-0.6)	**90.2** (-0.2)	**88.8** (-0.4)	83.0 (-3.6)	82.4 (0.0)	91.5 (-0.1)	90.6 (-0.1)	90.8 (-0.1)	**90.3** (0.0)	**89.0** (-0.1)
	$\mathcal{L}_{DN-S}$	**92.4** (-1.0)	90.0 (-0.2)	87.4 (-2)	**83.4** (-1.0)	**82.6** (-8.4)	91.8 (-0.3)	90.8 (-0.2)	**91.0** (-0.1)	90.3 (-0.1)	88.6 (-0.1)
word CNN	Baseline	88.8 (-1.4)	89.2 (-1.8)	84.8 (-8.0)	**82.2** (-15.0)	77.6 (-16.0)	90.9 (-2.7)	90.6 (-6.2)	89.3 (-10.2)	**89.2** (-17.9)	**87.4** (-25.2)
	$\mathcal{L}_{DN-H}$	91 (-0.2)	90.8 (-0.2)	**89.4** (-1.0)	81.4 (0.0)	**81.4** (-4.8)	**91.3** (-0.2)	**91.0** (-0.4)	**90.3** (-0.3)	88.3 (-3.2)	86.6 (-3.5)
	$\mathcal{L}_{DN-S}$	**92.2** (-1.4)	**91.8** (-2.0)	88.8 (-2.8)	77.0 (-2.4)	77.2 (-7.0)	90.9 (0.0)	90.4 (-0.1)	88.7 (-1.1)	86.6 (-3.5)	84.5 (-10.2)

Table 1: Results from experiments using random noise. Here for A(B): A refers to the *Best* model accuracy while B refers to (*Last-Best*) accuracy. The models with highest *Best* accuracies are in **bold**. For each noise %, the least and most reductions in *Last* accuracy are highlighted in green and red. Baseline (0% noise) reported beside dataset.

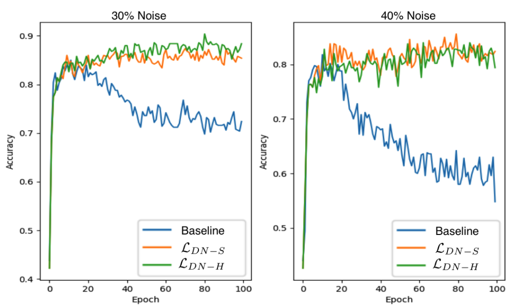


Figure 3: Test accuracy across training epochs of LSTM model on the TREC dataset with two levels of random noise: 30% and 40%.

Model	Noise %	'How'/'What' based			Question Length based		
		Base	$\mathcal{H}$	$\mathcal{S}$	Base	$\mathcal{H}$	$\mathcal{S}$
word LSTM	10	89.2	**91.8**	91.8	91.4	91.6	**92.0**
	20	84.4	87.4	**90.6**	87.0	90.2	**90.6**
	30	77.8	**84.2**	83.8	82.2	**87.4**	85.4
	40	76.0	79.0	**79.2**	82.4	**87.4**	84.0
	50	71.8	67.8	**75.6**	74.2	**79.0**	75.0
word CNN	10	90.4	90.0	**91.2**	91.0	90.6	**92.8**
	20	83.8	86.6	**86.8**	88.0	89.6	**91.0**
	30	82.4	**84.4**	84.2	85.2	**87.2**	86.8
	40	78.8	80.6	**81.8**	82.0	82.6	**86.0**
	50	52.0	**74.0**	65.2	73.6	**77.0**	75.4

Table 2: Input-conditional noise on TREC.

Noise Type (%)	word LSTM			word CNN		
	Base	$\mathcal{H}$	$\mathcal{S}$	Base	$\mathcal{H}$	$\mathcal{S}$
AP (7.8 %)	82.8	82.7	**82.8**	**83.1**	82.4	82.5
Reuters (10.8%)	85.6	**85.7**	85.5	85.7	**86.2**	86.1
AP+Reuters (18.6%)	75.7	**76.6**	76.0	**76.6**	76.1	76.4

Table 3: Results from input-conditional noise on AG-News. $\mathcal{H}/\mathcal{S}$ denotes $\mathcal{L}_{\mathcal{DN}-\mathcal{H}/\mathcal{S}}$.

We further demonstrate that our approach avoids over-fitting, thereby *stabilizing* the model training by plotting the test accuracies across training epochs in Fig. 3. We observe that the baseline model over-fits the label noise with more training epochs, thereby degrading test accuracy. The degree of over-fitting is greater at higher levels of noise (Fig. 3(b) vs Fig. 3(a)). In comparison, our de-noising approach using both L_{DN-S} and L_{DN-H} does not over-fit on the noisy labels as demonstrated by stable test accuracies across epochs.

Input-Conditional Noise We heuristically condition the noise function $\mathcal{F}$ on lexical and syntactic input features. We are the first to study input-conditional label noise, to the best of our knowledge. For both the TREC and AG-News datasets, we condition $\mathcal{F}$ on syntactic features of the input:

(i) The TREC dataset contains different types of questions. We selectively corrupt the labels of inputs that contain the question words 'How' or 'What' (chosen based on occurrence frequency). For texts starting with 'How' or 'What', we insert random label noise (at different levels).

(ii) The AG-News dataset contains news articles from different news agency sources. We selectively insert random label noise for inputs containing either one or both of the tokens 'Reuters' and 'AP'. For the TREC dataset, we also consider $\mathcal{F}$ conditional on the text length (a lexical feature). More specifically, we inject random label noise for the longest x% inputs in the dataset. We concatenate the contextualised input embedding from the penultimate layer of M and the logits corresponding to $\hat{y}^{(c)}$ as the input $R_M(x)$ to N_M. We observe that the baseline does not over-fit a lot on the noisy labels, possibly due to its complexity, and hence we only present the *Best* accuracies in Tables 2 and 3.

On TREC, our method outperforms the baseline for both the noise patterns we consider. Remarkably, our approach is robust to complex label noise conditional on subtle input features. For the question-length based noise, we observe the same trend of $L_{DN-Hard}$ outperforming $L_{DN-Soft}$ at high noise levels, and vice-versa. On AG-News, the noise % are relatively low, and our method performs at par or marginally improves over the baseline. These experiments reveal promising preliminary results on learning with input-conditional noise. For complete results refer to Appendix.

5 Conclusion

We have presented an approach to improve text classification when learning from noisy labels by jointly training a classifier and a noise model using a de-noising loss. We have evaluated our approach on two text classification tasks. Interesting future work includes studying more complex $\mathcal{F}$ for other NLP problems like natural language inference.

References

Sumeet Agarwal, Shantanu Godbole, Shourya Roy, and Diwakar Punjani. How much noise is too much: A study in automatic text classification. In *In Proc. of ICDM*, 2007.

Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin Mcguinness. Unsupervised label noise modeling and loss correction. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 312–321, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL `http://proceedings.mlr.press/v97/arazo19a.html`.

Ehsan Ardehaly and Aron Culotta. Learning from noisy label proportions for classifying online social data. *Social Network Analysis and Mining*, 8, 12 2018. doi: 10.1007/s13278-017-0478-6.

Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25:845–869, 2014.

James Goodman, Andreas Vlachos, and Jason Naradowsky. Noise reduction and targeted exploration in imitation learning for Abstract Meaning Representation parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1–11, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1001. URL `https://www.aclweb.org/anthology/P16-1001`.

A. Gulli. The anatomy of a news search engine. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, WWW '05, pp. 880–881, New York, NY, USA, 2005. Association for Computing Machinery.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `http://dx.doi.org/10.1162/neco.1997.9.8.1735`.

Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, HCOMP '10, pp. 64–67, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450302227. doi: 10.1145/1837885.1837906. URL `https://doi.org/10.1145/1837885.1837906`.

Junjun Jiang, Jiayi Ma, Zheng Wang, Chen Chen, and Xianming Liu. Hyperspectral image classification in the presence of noisy labels. *IEEE Transactions on Geoscience and Remote Sensing*, 57: 851–865, 2019.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 10–15 Jul 2018. URL `http://proceedings.mlr.press/v80/jiang18c.html`.

Ishan Jindal, Daniel Pressel, Brian Lester, and Matthew Nokleby. An effective label noise model for dnn text classification. In *Proceedings of North American Chapter of the Association of Computational Linguistics 2019*, 2019.

Aditya Joshi, Abhijit Mishra, Nivvedan Senthamilselvan, and Pushpak Bhattacharyya. Measuring sentiment annotation complexity of text. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, pp. 1746–1751, 2014.

Stefan Larson, Anish Mahendran, Andrew Lee, Jonathan K. Kummerfeld, Parker Hill, Michael A. Laurenzano, Johann Hauswald, Lingjia Tang, and Jason Mars. Outlier detection for improved data quality and diversity in dialog systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 517–527, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1051. URL `https://www.aclweb.org/anthology/N19-1051`.

Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pp. 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

H. H. Malik and V. S. Bhardwaj. Automatic training data cleaning for text classification. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pp. 442–449, 2011.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014. URL `http://www.aclweb.org/anthology/D14-1162`.

Scott E. Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In Yoshua Bengio and Yann LeCun (eds.), *ICLR (Workshop)*, 2015. URL `http://dblp.uni-trier.de/db/conf/iclr/iclr2015w.html#ReedLASER14`.

Sunil Thulasidasan, Tanmoy Bhattacharya, Jeff Bilmes, Gopinath Chennupati, and Jamal Mohd-Yusof. Combating label noise in deep learning using abstention. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 09–15 Jun 2019. URL `http://proceedings.mlr.press/v97/thulasidasan19a.html`.

Fei Wang, Liren Chen, Cheng Li, Shiyao Huang, Yanjie Chen, Chen Qian, and Chen Change Loy. The devil of face recognition is in the noise. *arXiv preprint arXiv:1807.11649*, 2018.

Xueying Zhan, Yaowei Wang, Yanghui Rao, and Qing Li. Learning from multi-annotator data: A noise-aware classification framework. *ACM Trans. Inf. Syst.*, 37(2), February 2019. ISSN 1046-8188. doi: 10.1145/3309543. URL `https://doi.org/10.1145/3309543`.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ICLR*, 2016. URL `http://arxiv.org/abs/1611.03530`. cite arxiv:1611.03530Comment: Published in ICLR 2017.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=r1Ddp1-Rb`.

Aleksandar Zlateski, Ronnachai Jaroensri, Prafull Sharma, and Fredo Durand. On the importance of label quality for semantic segmentation. In *CVPR*, pp. 1479–1487, 06 2018. doi: 10.1109/CVPR.2018.00160.

Appendix

A Details of BMM

We fit a 2-component Beta mixture model over the normalized training losses ($\mathcal{L}_{CE}(\hat{y}^{(c)}, \cdot) \in [0,1]$) obtained after training the model for some warmup epochs T_0. Similar to Arazo et al. (2019), we observe that using a Beta mixture model works better than using a Gaussian mixture model as it allows for asymmetric distributions and can capture the short left-tails of the clean sample losses. For a sample (x, y) with normalized loss $\mathcal{L}_{CE}(\hat{y}^{(c)}, y) = \ell$, the 2-component BMM can be represented as:

$$p(\ell) = \lambda_c \cdot p(\ell|\text{clean}) + \lambda_n \cdot p(\ell|\text{noisy})$$
$$p(\ell|\text{clean}) = \frac{\Gamma(\alpha_c + \beta_c)}{\Gamma(\alpha_c)\Gamma(\beta_c)} \ell^{\alpha_c - 1}(1-\ell)^{\beta_c - 1}$$
$$p(\ell|\text{noisy}) = \frac{\Gamma(\alpha_n + \beta_n)}{\Gamma(\alpha_n)\Gamma(\beta_n)} \ell^{\alpha_n - 1}(1-\ell)^{\beta_n - 1}$$

where Γ denotes the gamma distribution and $\alpha_{c/n}, \beta_{c/n}$ are the parameters corresponding to the individual clean/noisy BMMs. The mixture coefficients λ_c and λ_n, and parameters ($\alpha_{c/n}, \beta_{c/n}$) are learnt using the EM algorithm. On fitting the BMM $\mathcal{B}$, for a given input x with a normalized loss $\mathcal{L}_{CE}(\hat{y}^{(c)}, y) = \ell$, we the posterior probability of x having a clean label is given by:

$$\mathcal{B}(x) = \frac{\lambda_c \cdot p(\ell|\text{clean})}{\lambda_c \cdot p(\ell|\text{clean}) + \lambda_n \cdot p(\ell|\text{noisy})}$$

B Experimental Details

Datasets We experiment with two popular datasets: (i) the TREC question-type classification dataset Li & Roth (2002), and (ii) AG-News classification dataset Gulli (2005). The statistics of the datasets are shown in Table 4.

Dataset	# Classes	Train	Validation	Test
TREC	6	4949	503	500
AG-News	4	112000	8000	7600

Table 4: Summary statistics of the datasets

Models We conduct experiments on two popular model architectures: word-LSTM Hochreiter & Schmidhuber (1997) and word-CNN Kim (2014). For word-LSTM, we use a 2-layer BiLSTM with hidden dimension of 150. In the word-CNN, we use 300 kernel filters each of size 3, 4 and 5. We use the pre-trained GloVe embeddings Pennington et al. (2014) for training both models. We train models on TREC and AG-News for 100 and 30 epochs respectively. We use an Adam optimizer with a learning rate of 10^{-5} and a dropout of 0.3 during training. For the noise model N_M, we use a simple 2-layer feedforward neural network, with the number of hidden units $n_{hidden} = 4 \cdot n_{input}$. We choose the inputs to the noise model $R_M(x)$ as per the class of label noise, as described in the main text. We conduct hyper-parameter tuning for the number of warmup epochs T_0 and β using grid search over the ranges of {6,10,20} and {2,4,6,8,10} respectively.

C Full Results

Tables 5 and 6 show the full results of our experiments on input-conditional noise. We report two metrics for each model trained: (i) *Best*: the test accuracy of the model which achieves the best validation accuracy, and (ii) *Last*: the test accuracy of the model obtained after the last training epoch.

Input Dependent Noise (TREC)		Question Token (How/What) based						Question Length based					
		Baseline		$\mathcal{L}_{DN-H}$		$\mathcal{L}_{DN-S}$		Baseline		$\mathcal{L}_{DN-H}$		$\mathcal{L}_{DN-S}$	
Model	Noise %	Best	Last	Best	Last	Best	Last	Best	Last	Best	Last	Best	Last
word LSTM	0	93.8	93.0	94.0	92.6	**95.0**	94.0	93.8	93.0	94.0	92.6	**95.0**	94.0
	10	89.2	88.8	**91.8**	91.8	91.8	92.0	91.4	90.4	91.6	91.0	**92.0**	92.4
	20	84.4	76.2	87.4	85.2	**90.6**	89.4	87.0	87.6	90.2	89.4	**90.6**	91.6
	30	77.8	67.2	**84.2**	84.6	83.8	77.0	82.2	84.0	**87.4**	87.2	85.4	85.6
	40	76.0	59.0	79.0	80.0	**79.2**	60.0	82.4	79.8	**87.4**	86.6	84.0	81.2
	50	71.8	56.0	67.8	69.2	**75.6**	59.8	74.2	71.2	**79.0**	79.0	75.0	72.0
word CNN	0	92.6	93.2	**92.8**	91.6	**92.8**	91.8	92.6	93.2	**92.8**	91.6	**92.8**	91.8
	10	90.4	86.8	90.0	90.8	**91.2**	89.8	91.0	91.0	90.6	89.8	**92.8**	89.2
	20	83.8	82.0	86.6	83.6	**86.8**	85.0	88.0	89.2	89.6	88.6	**91.0**	88.8
	30	82.4	75.0	**84.4**	83.8	84.2	80.0	85.2	84.4	**87.2**	87.0	86.8	85.4
	40	78.8	61.6	80.6	76.4	**81.8**	69.8	82.0	79.4	82.6	82.2	**86.0**	82.0
	50	52.0	53.4	**74.0**	66.4	65.2	52.8	73.6	72.2	**77.0**	70.8	75.4	77.2

Table 5: Results for TREC dataset with Input-Conditional Noise (Best model is shown in bold)

Input Dependent Noise (AG-News)	wordLSTM						wordCNN					
	Baseline		$\mathcal{L}_{DN-H}$		$\mathcal{L}_{DN-S}$		Baseline		$\mathcal{L}_{DN-H}$		$\mathcal{L}_{DN-S}$	
Noise Type (%)	Best	Last	Best	Last	Best	Last	Best	Last	Best	Last	Best	Last
None (0 %)	92.5	92.1	92.4	92.0	**92.8**	92.6	92.5	92.1	92.4	92.0	**92.8**	92.6
AP (7.8 %)	82.8	82.3	82.7	82.7	**82.8**	83.1	**83.1**	82.9	82.4	83.2	82.5	83.0
Reuters (10.8%)	85.6	84.8	**85.7**	85.6	85.5	85.6	85.7	85.7	**86.2**	86.2	86.1	86.2
AP+Reuters (18.6%)	75.7	75.3	**76.6**	76.2	76.0	75.9	**76.6**	75.7	76.1	76.2	76.4	76.4

Table 6: Results for AG-News dataset with Input-Conditional Noise (Best model is shown in bold)