TEDL: A Two-stage Evidential Deep Learning Method for Classification Uncertainty Quantification

Xue Li  
Microsoft  
USA  
xeli@microsoft.com

Wei Shen  
Microsoft  
USA  
sashen@microsoft.com

Denis Charles  
Microsoft  
USA  
cdx@microsoft.com

ABSTRACT

In this paper, we propose TEDL, a two-stage learning approach to quantify uncertainty for deep learning models in classification tasks, inspired by our findings in experimenting with Evidential Deep Learning (EDL) method, a recently proposed uncertainty quantification approach based on the Dempster-Shafer theory. More specifically, we observe that EDL tends to yield inferior AUC compared with models learnt by cross-entropy loss and is highly sensitive in training. Such sensitivity is likely to cause unreliable uncertainty estimation, making it risky for practical applications. To mitigate both limitations, we propose a simple yet effective two-stage learning approach based on our analysis on the likely reasons causing such sensitivity, with the first stage learning from cross-entropy loss, followed by a second stage learning from EDL loss. We also re-formulate the EDL loss by replacing ReLU with ELU to avoid the Dying ReLU issue. Extensive experiments are carried out on varied sized training corpus collected from a large-scale commercial search engine, demonstrating that the proposed two-stage learning framework can increase AUC significantly and greatly improve training robustness.

CCS CONCEPTS

• Information systems → Retrieval models and ranking; Computational advertising; Language models.

KEYWORDS

search ads recommendation, deep learning, classification, uncertainty quantification, BERT, TwinBERT

1 INTRODUCTION

Uncertainty quantification of deep learning models has been a hot topic in the community ever since the rise of deep learning, and the demand for effective uncertainty quantification methods is becoming increasingly urgent in the recent decade as deep learning continue to reshape many industries. Search recommendation, as perhaps the most radically reshaped industry, often relies on many different deep learning models to give accurate recommendations, which makes uncertainty quantification especially important since unreliable predictions could accumulate in the system and finally lead to inaccurate or even embarrassing recommendation results.

To make machine learning models aware of their own prediction confidence, many uncertainty quantification approaches have been proposed [8], including single deterministic methods, Bayesian methods and ensemble methods, etc., among which the single deterministic methods could be further grouped into internal or external methods depending on whether additional components are required for uncertainty estimation. We present a brief review on this topic in Section 2. In this paper, we are particularly interested in single deterministic methods, especially internal approaches, since such methods typically need only a single forward pass on a deterministic network to estimate uncertainty, and hence does not require stochastic DNN or ensemble models, making both training and inference more efficient.

More specifically, instead of considering the model outputs as a pointwise maximum-a-posteriori (MAP) estimation, internal single deterministic methods usually interpret model outputs as parameters of a prior distribution over all the possible predictions, and then give prediction by taking the expected value over the prior distribution. For classification tasks, Dirichlet distribution is often chosen as prior since it is the conjugate prior of the categorical distribution. Meanwhile, statistical distance metrics such as Kullback-Leibler (KL) divergence are often included in their loss functions due to the need to optimize on parameters of distributions [21, 36].

However, the efficiency of such methods comes with a cost. As mentioned in [8], they are typically more sensitive towards training settings such as initialization, hyper-parameters, training data, etc., which is what we observed when apply EDL [36], a recently proposed single deterministic method, to practical scenarios.

To be more specific, in our experiments we identify several issues in the EDL method. Firstly, as shown in Figure 2, when applied to binary classification tasks, the ROC AUC achieved by the EDL method is significantly lower than that obtained by cross-entropy loss, and such gap cannot be bridged by simply adding more training samples. Secondly, EDL tends to be sensitive to initialization and some hyper-parameters, where improper settings may lead to significantly degraded AUC and unreliable uncertainty estimation.

To see this more clearly, in Figure 5 (the orange curve) we summarize the per epoch ROC AUC obtained in EDL training with different $\lambda$, a hyper-parameter controlling how close the Dirichlet
prior is to a uniform distribution. As we can see, the AUC of EDL suffers in the beginning under all the four settings, and in some cases (for example when $\lambda = 0.5$) there is no signs of improvement at all. In cases where AUC does improve, its final AUC is still significantly lower than that from the proposed method (the green curve). On the other hand, consider evaluating AUC on validation samples with uncertainty lower than a certain threshold: If the learnt uncertainty is of high quality, smaller thresholds should indicate higher confidence, and hence should be associated with higher AUC. However, this is not always the case for EDL, as shown in the first row of Figure 6. Besides, we also observe that when a large $\lambda$ is used (for example $\lambda=0.75$), there would be a higher risk of running into the Dying ReLU problem where all outputs are zero, leading to an AUC that similar to a random guess. All these issues make it risky to apply methods like EDL into real-world applications.

To fix these issues, we firstly present an analysis in this paper on the likely reasons causing the above issues in Section 3, and based on our analysis, we further propose TEDL, short for Two-stage Evidential Deep Learning, as a simple but effective training framework to mitigate all the aforementioned issues in a single shot. As we will see in Section 3, the basic idea of TEDL is to transform the difficult uncertainty quantification problem into two sub-problems that are much easier to tackle, i.e., 1) finding a reasonably good pointwise estimation of the categorical distribution, and 2) leveraging this pointwise estimation as an anchor point for estimating the Dirichlet prior of categorical distribution, based on which we can quantify uncertainty.

The overall training framework of TEDL is illustrated in Figure 1, where two stages are needed: in the first stage, we train our classification model with cross-entropy loss, in order to obtain a model that is able to output reasonable pointwise estimations of the categorical distribution. And then in the second stage, we initialize the model from the weights obtained in the previous stage, and go through the same training corpus by learning with the reformulated EDL loss where ReLU is replaced by ELU. As shown in Section 4, compared with the EDL baseline, TEDL can achieve higher AUC across all evaluation settings and effectively avoid the risk of running into Dying ReLU problem. More importantly, TEDL also shows significantly improved robustness towards training settings, making it more reliable for practical applications.

It is also worth to mention that we name our proposed method following EDL mainly due to the convenience of experimentation, as it is proposed recently and is easy to implement with code open-sourced by the authors. However, our analysis in Section 3 also applies to other single deterministic uncertainty quantification methods suffering from similar issues, and hence the two-stage learning framework we propose in this paper could be readily extended to those methods as well.

2 RELATED WORKS

The interest for uncertainty estimation dates back to the days even before the rise of deep learning, entailing a large body of literature on this topic. Based on whether model ensemble is used and whether the model is stochastic, uncertainty quantification methods could be roughly grouped into three categories, including single deterministic methods, Bayesian neural networks and ensemble methods. Please refer to [8] for a comprehensive survey.

**Single deterministic methods** [24, 25, 30] estimate uncertainty based on one single forward pass within a deterministic network, and could be further split into external approach [15, 31] and internal approach [21, 32, 36] depending on whether additional method is used for deriving uncertainty estimation. Methods in this category typically have lower requirements on computational resources.
since no stochastic networks nor model ensembles are needed, but suffer from sensitivity to initialization and parameters compared with other categories. The proposed TEDL method in this paper, as well as the original EDL method, both fall into this category.

Bayesian neural networks cover all kinds of stochastic DNNs, including methods based on variational inference [2, 3, 7, 9, 11, 19, 33], sampling methods [26–29, 38], and Laplace approximation [16, 34, 35]. Methods in this category usually have higher computational complexity in both the training and inference phases due to stochastic sampling.

Ensemble methods [1, 4, 10, 12, 14, 18, 22, 23, 37, 39] combine the predictions from several different deterministic networks at inference. Methods in this category typically have higher requirements on both the memory and computational resources at inference phase.

The proposed method also relates to the concept of two-stage learning, which bears similarity to transfer learning but has some subtle differences. Transfer learning generally refers to the procedure that transfers knowledge obtained from different but related source domains to target domains, usually to reduce training data required on the target domains. [41] gives a comprehensive survey on transfer learning. In contrast, in two-stage learning [5, 13], although it also consists of two consecutive stages, these two stages are often conducted on the same data. In a typical two-stage learning setting, the second stage should be the final stage that yields the desired output, while the first stage serves as a preparation step. Given such differences, the proposed method should be categorized as two-stage learning.

3 APPROACH

3.1 A Recap on EDL Uncertainty Quantification
The basic idea of EDL method is treating softmax output as the pointwise estimation of the categorical distribution, and placing a Dirichlet prior over the distribution of all possible softmax outputs. Then, following the Dempster-Shafer theory, assume we have K categories and \( \alpha_j = (\alpha_{i1}, \ldots, \alpha_{iK}) \) is the parameter of a Dirichlet distribution for the classification of sample \( i \), the authors propose to replace softmax with ReLU and represent the Dirichlet parameter as \( \alpha_j = f(x_i|\Theta) + 1 \) where \( \Theta \) represents network parameters and \( f(x_i|\Theta) \) is the ReLU outputs. The \( \alpha_{ij} \) here also represents the subjective opinion collected from sample \( i \) and category \( j \), and \( S_i = \sum_{j=1}^{K} \alpha_{ij} \) is referred to as the Dirichlet strength. Note that \( S_i \) is inversely proportional to uncertainty: a larger \( S_i \) indicates more evidence are collected for sample \( i \), and hence less uncertainty.

Based on the above assumptions, the EDL loss is defined as below:

\[
L_i(\Theta) = \sum_{i=1}^{N} L_i(\Theta) + \lambda_t \sum_{i=1}^{N} KL[D(p_i|\tilde{\alpha}_i) \| D(p_i|(1, \ldots, 1))] 
\]

where \( L_i(\Theta) \) is formulated as the expected value of a basic loss. According to the authors of [36], EDL method appears relatively more stable when sum of squares loss is used as the basic loss, as below:

\[
L_i(\Theta) = \sum_{j=1}^{K} (y_{ij} - \hat{p}_{ij})^2 + \hat{\beta}_{ij}(1 - \hat{p}_{ij}) \frac{s_i + 1}{s_i} = \sum_{j=1}^{K} (y_{ij} - \frac{a_{ij}}{s_i})^2 + \frac{a_{ij}(S_i - a_{ij})}{s_i^2(S_i + 1)} \tag{2}
\]

where \( y_{ij} \) and \( \hat{p}_{ij} \) denote the class label and expectation for sample \( i \) and class \( j \), respectively.

Equation (1) also contains a regularization term minimizing the KL divergence between the estimated Dirichlet distribution \( D(p_i|\tilde{\alpha}_i) \) and the uniform distribution. Its associated coefficient \( \lambda_t \) is heuristically set to increase with epoch \( t \) (zero-based), i.e., \( \lambda_t = \min(1.0, t + \lambda) \) where \( \lambda = 0.1 \). Note that we denote the per-epoch increment as \( \lambda \). For brevity, we will treat \( \lambda \) rather than \( \lambda_t \) as the hyper-parameter henceforth, since \( \lambda_t \) is determined only by \( \lambda \).

3.2 A Closer Look into the EDL Method
Equation (1) could be split into two parts: the first part is Equation (2) which is designed to estimate the Dirichlet prior, and the second part is the regularization term derived from KL divergence. Next, we will take a closer look at these two parts respectively to understand the cause of sensitivity.

As we mentioned previously, unlike cross-entropy loss which is designed to learn the pointwise estimations of the categorical distribution as a MAP estimate, the loss function in Equation (2) is derived to learn the parameter of a Dirichlet prior distribution over all the possible predictions. Therefore, the pointwise estimation should also be covered by the Dirichlet prior distribution. This perspective highlights the huge gap in terms of how difficult the optimization problems behind these two loss functions are, especially given that obtaining a good MAP estimation is already a hard problem in many applications. This perspective also highlights the importance of a sufficiently large training data, as it would be meaningless to model a distribution without sufficient samples.

In the meanwhile, the KL divergence also makes optimization more complicated since it is not Lipschitz smooth. More precisely, given a function \( f \), it is said to be Lipschitz smooth if and only if there exists a finite value \( L \) such that

\[
\| \nabla f(a) - \nabla f(b) \| < L \cdot \| a - b \| \tag{3}
\]

In other words, the gradient of \( f \) should exist and be bounded by a finite value \( L \). However, the regularization term in Equation (1) does not satisfy this condition since its gradient will go to infinity when \( D(p_i|\tilde{\alpha}_i) \rightarrow 0 \), as even though \( \tilde{\alpha}_i \) is guaranteed to be positive, \( p_i \) may still become very close to zero when a certain \( \tilde{\alpha}_i \) is extremely large, leading to very large gradients and hence unstable training.

In summary, internal single deterministic methods are trying to optimize an inherently difficult problem, with potentially ill-conditioned loss functions due to existence of KL divergence.

3.3 The Proposed Two-stage Learning Framework
Having analyzed the possible reasons causing training sensitivity, a more important question is how could we fix such issues and make
training more stable. At first glance, this appears to be infeasible since we can neither bypass distribution modeling nor drop the terms related to KL divergence in loss functions. In this paper, we propose an alternative approach, which can fix both issues with a simple yet effective strategy: decomposing the original problem into two sub-problems and tackling one at a time, leading to a two-stage learning method as illustrated in Figure 1. Compared with the original EDL method, the only cost introduced by TEDL is a preparation stage learning from the cross-entropy loss, however as we will see in Section 4, such cost is well paid off given the significant AUC increase and greatly improved robustness in training.

So why does such a simple strategy work? On one hand, the first stage in TEDL learns a pointwise estimation of the categorical distribution, which is a much easier problem compared with modeling the entire distribution and entails much fewer training samples. Then in the second stage, since the model is initialized from the weights obtained in stage 1, it amounts to modeling the prior distribution using the pointwise estimation as certain anchor points, which is much easier than modeling the prior from scratch, if we can assume that the pointwise estimation is close to the expected value of the prior. This assumption should be easily hold for most practical applications, otherwise we will not be able to apply internal single deterministic methods at all, since the expected value from the prior distribution is unlikely to derive meaningful predictions in that case.

On the other hand, by learning from cross-entropy loss, we could effectively avoid assigning extremely small values to \( \mathcal{P}_i \), given that softmax involves exponential operations and there is no point in pushing model outputs before softmax to extremely large values. That means, when softmax is replaced by ELU later in stage 2, it is unlikely for us to see extremely large \( \bar{\alpha}_i \) values.

4 EXPERIMENTS

4.1 Implementation Details

All experiments throughout this paper are conducted on a binary classification task, with the goal to predict whether a \( \langle \text{query}, \text{ad} \rangle \)-pair is relevant or not. Both the training (1.4M) and validation samples (100K) are sampled from a large-scale commercial search engine, with human-provided relevance labels. In order to examine the impact of the size of training data, we further create a synthetic training set with soft labels, by sampling a large corpus and inference using an ensemble of BERT [6] models fine-tuned on the human-labeled training set, similar to what we do in knowledge distillation [17]. This allows us to experiment on a much larger scale, without breaking any assumptions in the EDL method. Without further clarification, we will henceforth refer to this synthetic training set as our training corpus, and experiments will be conducted on subsets sampled from this synthetic training set, with 1M, 5M, 50M and 500M samples, respectively.

In addition, in this paper we will use TwinBERT [20, 40] as our deep classification model, which uses two BERT encoders to encode query and ad respectively, and then calculates their relevance score by cosine similarity. We choose this model mainly for its simplicity and efficiency, and the conclusions of this paper should hold for other model architectures as well, since no particular assumptions for model architectures are made in the proposed TEDL method.

In terms of metrics, since we are working on binary classification task, we will use ROC AUC to evaluate the prediction performance (in our experiments PR AUC shows a very similar trend to ROC AUC). Meanwhile, to measure the quality of uncertainty, we follow the approach in [36] to split our validation data using different uncertainty thresholds first, and then evaluate ROC AUC on each individual subset. For example, when threshold is 0.1, ROC AUC will be calculated only on validation samples with uncertainty lower than 0.1. Therefore, if uncertainty is properly quantified, we should expect higher ROC AUC on lower thresholds, since this is the subset that our model feels more confident with. This way, we can plot a curve over ROC AUC v.s. uncertainty thresholds.

4.2 Results and Analysis

4.2.1 Classification Performance evaluated by ROC AUC. Figure 2 summarizes the per-epoch ROC AUC of models learnt by cross-entropy loss, EDL method and the proposed TEDL method, with 1M, 5M and 50M training samples respectively. In all the three
settings, we consistently observe that the ROC AUC from EDL method is much lower than that from cross-entropy loss, while the proposed TEDL method is able to achieve comparable performance than cross-entropy loss, outperforming EDL significantly.

In addition, if we look into ROC AUC measured on different epochs in Figure 2, we can also see that TEDL is much more stable than EDL, especially when training corpus is relatively small.

4.2.2 Quality of Uncertainty. As mentioned previously, we will measure the quality of the learnt uncertainty by plotting a curve over ROC AUC v.s. uncertainty thresholds, as shown in Figure 3, where the first row corresponds to EDL, while the second row is for TEDL. By comparing plots from different epochs, we can see that the quality of uncertainty learnt from both EDL and TEDL gets steadily improved over the training process, and the improving pattern for EDL and TEDL are very similar. However, this only happens when a relatively small $\lambda$ is used. Later in Section 4.3 we will see that compared with EDL, TEDL is much more robust towards $\lambda$. We also plot the distribution of uncertainty in each training epoch, as shown in Figure 4, where TEDL also looks similar to EDL when $\lambda$ is relatively small, but later in Section 4.3 we will see their difference when $\lambda$ gets larger.

4.3 Sensitivity towards Hyper-parameters

So far all the results we report are obtained under mild conditions with $\lambda = 0.1$, however as we mentioned in Section 1, $\lambda$ and the number of training epochs may have dramatic impact on EDL, and hence it is necessary to examine how robust TEDL is towards these two hyper-parameters.

4.3.1 ROC AUC. Figure 5 compares the ROC AUC obtained by EDL and TEDL method, respectively, under different $\lambda$ values. Similar to Figure 2, TEDL constantly outperforms EDL, and is more stable when more training epochs are used. In particular, when $\lambda = 0.75$ we observe the Dying ReLU problem in EDL, which inspires us to replace ReLU by ELU in TEDL.

4.3.2 Quality of Uncertainty. Figure 6 and Figure 7 compare the quality of uncertainty learnt by EDL and TEDL method, respectively, under different $\lambda$ values. Compared with Figure 3 and Figure 4, the uncertainty quality learnt from EDL degrades dramatically when larger $\lambda$ is used, as shown in the case where $\lambda = 0.25$ and $\lambda = 0.5$. By contrast, for TEDL, both its plots over ROC AUC vs. uncertainty as well as its uncertainty distribution look very similar to what we observed for $\lambda = 0.1$, demonstrating significantly improved robustness towards $\lambda$.

5 CONCLUSION

In this paper, we propose TEDL, a two-stage learning approach to quantify uncertainty for deep classification models. TEDL contains two stages: the first stage learns from cross-entropy loss to obtain a good point estimate of the Dirichlet prior distribution, and then the second stage learns to quantify uncertainty via the reformulated EDL loss. We conduct extensive experiments using training corpus...
Figure 4: Uncertainty distribution of EDL (first row) and TEDL (second row), learnt on 1M, 5M and 50M training samples with $\lambda = 0.1$. The first row is for EDL, while the second row is for TEDL.

Figure 5: Comparison of ROC AUC for EDL and TEDL, learnt on 1M training corpus with different $\lambda$. Compared to EDL, TEDL not only achieves higher ROC AUC, but also shows improved robustness towards $\lambda$, especially when $\lambda = 0.75$ where EDL method runs into the Dying ReLU problem.

Sampled from a real commercial search engine, which demonstrates that compared with EDL, the proposed TEDL not only achieves higher AUC, but also shows improved robustness towards hyper-parameters. As future work, the uncertainty learnt by TEDL may be leveraged to developing active learning algorithms.
Figure 6: Comparison of ROC AUC vs. uncertainty for EDL (first row) and TEDL (second row), learnt on 1M training corpus with different $\lambda$, where TEDL shows significantly better robustness.

Figure 7: Comparison of uncertainty distribution for EDL (first row) and TEDL (second row), learnt on 1M training corpus with different $\lambda$, where TEDL shows significantly better robustness.
REFERENCES

[1] Omer Achrack, Raizi Kellerman, and Ouriel Barzilay. 2020. Multi-loss sub-ensembles for accurate classification with uncertainty estimation. arXiv preprint arXiv:2001.01917 (2020).

[2] David Barber and Christopher M Bishop. 1998. Ensemble learning in Bayesian neural networks. Nato ASI Series F Computer and Systems Sciences 168 (1998), 215–238.

[3] Charles Blundell, Julian Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural network. In International conference on machine learning. PMLR, 1613–1622.

[4] George DC Cavalcanti, Luiz S Oliveira, Thiago JM Moura, and Guillerme V Carvalho. 2016. Combining diversity measures for ensemble pruning. Pattern Recognition Letters 74 (2016), 38–45.

[5] Van Dang, Michael Bendersey, and W Bruce Croft. 2013. Two-stage learning to rank for information retrieval. In European Conference on Information Retrieval. Springer, 423–434.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018).

[7] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In International conference on machine learning. PMLR, 1050–1059.

[8] Jakob Gawlikowski, Cedrique Rovde Njeutcheu Tassi, Mohsin Ali, Jingfeok Lee, Matthias Hutt, Junxiong Feng, Anna Kruspe, Rudolf Triebel, Peter Jang, Ribana Roscher, et al. 2021. A survey of uncertainty in deep neural networks. arXiv preprint arXiv:2107.03342 (2021).

[9] Alex Graves. 2011. Practical variational inference for neural networks. Advances in neural information processing systems 24 (2011).

[10] Huoping Guo, Hongbing Liu, Ran Li, Changan Wu, Yibo Guo, and Mingliang Xu. 2018. Margin & diversity based ordering ensemble pruning. Neurocomputing 275 (2018), 237–246.

[11] Geoffrey E Hinton and Drew Van Camp. 1993. Keeping the neural networks simple by minimizing the description length of the weights. In Proceedings of the sixth annual conference on Computational learning theory. 5–13.

[12] Guo Huang, Yuxian Li, Geoff Pleiss, Zhengan Liu, John E Hopcroft, and Kilian Q Weinberger. 2017. Snapshot ensembles: Train 1, get m for free. arXiv preprint arXiv:1704.00109 (2017).

[13] Farrukh Aslam Khan, Abdu Gumaei, Abdelouahid Derhab, and Amir Hussain. 2021. Ensemble pruning via quadratic margin maximization. arXiv preprint arXiv:2005.15781 (2020).

[14] Jakob Lindqvist, Amanda Olmin, Fredrik Lindsten, and Lennart Svensson. 2020. Margin & diversity based ordering ensemble pruning. arXiv preprint arXiv:1910.08168 (2019).

[15] Xue Li, Zhipeng Luo, Hao Sun, Jianjin Zhang, Weihao Han, Xianqi Chu, Liangjie Zhang, and Qi Zhang. 2019. Learning Fast Matching Models from Weak Annotation. In International Conference on Machine Learning. PMLR, 5281–5290.

[16] Jay Nandy, Wynne Hsu, and Mong Li Lee. 2020. Towards maximizing the representation gap between in-domain & out-of-distribution examples. Advances in Neural Information Processing Systems 33 (2020), 9239–9250.

[17] Radford M Neal. 1994. An improved acceptance procedure for the hybrid Monte Carlo algorithm. J. Comput. Phys. 111, 1 (1994), 194–203.

[18] Radford M Neal. 2012. Bayesian learning for neural networks. Vol. 118. Springer Science & Business Media.

[19] Christopher Nemeth and Paul Fearnhead. 2021. Stochastic gradient markov chain monte carlo. J. Amer. Statist. Assoc. 116, 533 (2021), 433–450.

[20] Matheo Raghu, Katy Blumer, Rory Sayres, Ziad Obermeyer, Bobby Kleinberg, Senthil Mullainathan, and Jon Kleinberg. 2019. Direct uncertainty prediction for medical second opinions. In International Conference on Machine Learning. PMLR, 5281–5290.

[21] Triago Ramalho and Miguel Miranda. 2020. Density estimation in representation space to predict model uncertainty. In International Workshop on Engineering Dependable and Secure Machine Learning Systems. Springer, 84–96.

[22] Danilo Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. In International conference on machine learning. PMLR, 1530–1538.

[23] Hippolyt Ritter, Alexandras Botev, and David Barber. 2018. A scalable laplace approximation for neural networks. In 6th International Conference on Learning Representations, ICLR 2018–Conference Track Proceedings. Vol. 6. International Conference on Representation Learning.

[24] Tim Salimans and Dario Kingma. 2016. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. Advances in neural information processing systems 29 (2016).

[25] Murat Sensoy, Lance Kaplan, and Melih Kendemir. 2018. Evidential Deep Learning to Quantify Classification Uncertainty. Advances in Neural Information Processing Systems 31 (2018).

[26] Matias Valdenegro-Toro. 2019. Deep sub-ensembles for fast uncertainty estimation in image classification. arXiv preprint arXiv:1910.08168 (2019).

[27] Max Well and Yee W Teh. 2011. Bayesian learning via stochastic gradient Langevin dynamics. In Proceedings of the 28th international conference on machine learning (ICML-11). Citeseer, 681–688.

[28] Yeming Wen, Dustin Tran, and Jimmy Ba. 2020. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. arXiv preprint arXiv:2002.00715 (2020).

[29] Jason Zhu, Yanling Cui, Yuming Liu, Hao Sun, Xue Li, Markus Pelger, Tianqi Yang, Liangjie Zhang, Ruofei Zhang, and Huahua Zhao. 2021. TextGNN: Improving Text Encoder via Graph Neural Network in Sponsored Search. In Proceedings of the Web Conference 2021. 2848–2857.

[30] Fuwen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. Proc. IEEE 109, 1 (2020), 43–76.