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

Xue Li[1,*], Wei Shen[2] and Denis Charles[2]

[1] 1045 La Avenida St, Mountain View, CA, 94043, United States
[2] 555 110th Ave NE, Bellevue, WA, 98004, United States

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.

Keywords

uncertainty quantification, search ads recommendation, deep learning, classification, 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 [1], 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 [2, 3].

However, the efficiency of such methods comes with a cost. As mentioned in [1], they are typically more sensitive towards training settings such as initialization, hyper-parameters, training data, etc., which is what we observed when apply EDL [2], 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

DL4SR'22: Workshop on Deep Learning for Search and Recommendation, co-located with the 31st ACM International Conference on Information and Knowledge Management (CIKM), October 17-21, 2022, Atlanta, USA

*Corresponding author.
xeli@microsoft.com (X. Li); sashen@microsoft.com (W. Shen); cdx@microsoft.com (D. Charles)

© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-3317/Paper11.pdf

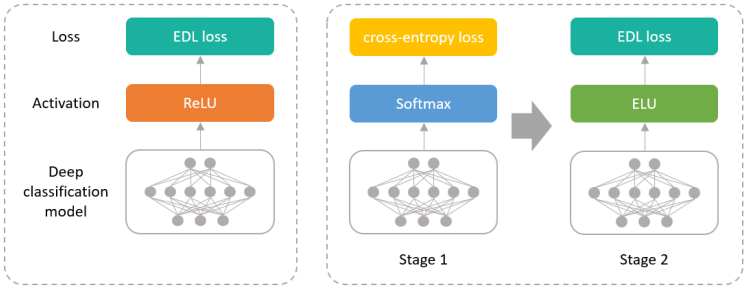

Figure 1: A schematic illustration of the proposed TEDL method. (a) The original EDL method transforms the model outputs to strictly positive values using *ReLU* activation and learns to quantify uncertainty via the EDL loss in Equation (1), which yields inferior AUC and is sensitive to training. (b) The proposed TEDL method employs a two-stage learning strategy to decompose the original problem into two easier sub-problems and tackle one at a time: the first stage learns to make good pointwise estimations via cross-entropy loss; and then the second stage will learn to quantify uncertainty using the pointwise estimation as anchor points, with *ReLU* replaced by *ELU* to avoid the *Dying ReLU* issue.

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 λ, 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 λ is used (for example λ=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 **T**wo-stage **E**vidential **D**eep **L**earning, 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 stochas-

tic, uncertainty quantification methods could be roughly grouped into three categories, including single deterministic methods, Bayesian neural networks and ensemble methods. Please refer to [1] for a comprehensive survey.

Single deterministic methods [4, 5] estimate uncertainty based on one single forward pass within a deterministic network, and could be further split into external approaches [6, 7] and internal approaches [2, 3, 8] 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 [9, 10, 11, 12, 13, 14, 15], sampling methods [16, 17, 18, 19], and Laplace approximation [20, 21, 22]. Methods in this category usually have higher computational complexity in both the training and inference phases due to stochastic sampling.

Ensemble methods [23, 24, 25, 26] 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. [27] gives a comprehensive survey on transfer learning. In contrast, in two-stage learning [28, 29], 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 $\boldsymbol{\alpha}_i = \langle \alpha_{i1}, \ldots, \alpha_{iK} \rangle$ 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 $\boldsymbol{\alpha}_i = f(\boldsymbol{x}_i|\Theta) + 1$ where Θ represents network parameters and $f(\boldsymbol{x}_i|\Theta)$ is the *ReLU* outputs. The α_{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 is collected for sample i, and hence lower uncertainty.

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

$$\mathcal{L}(\Theta) = \sum_{i=1}^{N} \mathcal{L}_i(\Theta) + \lambda_t \mathcal{L}_{KL} \tag{1}$$

where $\mathcal{L}_i(\Theta)$ is formulated as the expected value of a basic loss and $\mathcal{L}_{KL}$ represents regularization. According to the authors of [2], EDL method appears relatively more stable when sum of squares loss is used as the basic loss, as below:

$$\begin{aligned} \mathcal{L}_i(\Theta) &= \sum_{j=i}^{K} (y_{ij} - \hat{p}_{ij})^2 + \frac{\hat{p}_{ij}(1-\hat{p}_{ij})}{S_i + 1} \\ &= \sum_{j=1}^{K} (y_{ij} - \frac{\alpha_{ij}}{S_i})^2 + \frac{\alpha_{ij}(S_i - \alpha_{ij})}{S_i^2 (S_i + 1)} \end{aligned} \tag{2}$$

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

Meanwhile, the above loss function is further regularized by minimizing the KL divergence between the estimated Dirichlet distribution $D(\boldsymbol{p}_i|\tilde{\boldsymbol{\alpha}}_i)$ and the uniform distribution, as below:

$$\mathcal{L}_{KL} = \sum_{i=1}^{N} KL\left[D(\boldsymbol{p}_i|\tilde{\boldsymbol{\alpha}}_i) \;||\; D(\boldsymbol{p}_i|\langle 1, \ldots, 1 \rangle)\right] \tag{3}$$

The coefficient λ_t in Equation (1) 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 λ. For brevity, we will treat λ rather than λ_t as the hyper-parameter henceforth, since λ_t is determined only by λ.

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 in Equation (3) 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

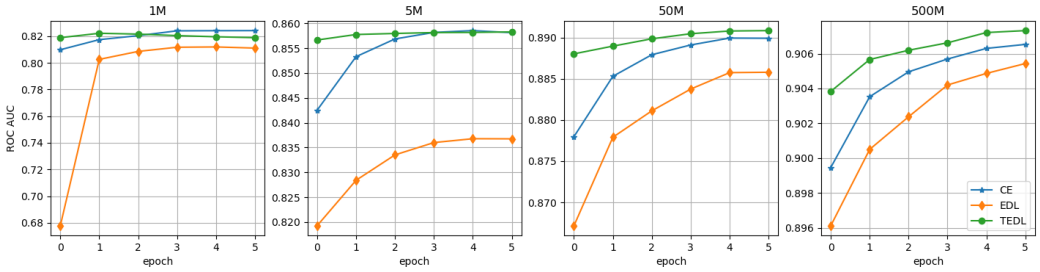

Figure 2: AUC comparison between cross-entropy loss, EDL loss and TEDL loss, evaluated on the same validation data. All the three methods are learnt on training corpus with 1M, 5M, 50M and 500M samples, respectively. In all the training settings, EDL method achieves inferior AUC compared to cross-entropy loss, while the proposed TEDL method yields comparable AUC than cross-entropy, outperforming EDL significantly.

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

$$\|\nabla f(a) - \nabla f(b)\| < L \cdot \|a - b\| \tag{4}$$

In other words, the gradient of f should exist and be bounded by a finite value L. However, the regularization term in Equation (3) does not satisfy this condition since its gradient will go to infinity when $D(\boldsymbol{p}_i|\tilde{\boldsymbol{\alpha}}_i) \to 0$, as even though $\tilde{\boldsymbol{\alpha}}_i$ is guaranteed to be positive, $\boldsymbol{p}_i$ may still become very close to zero when a certain $\tilde{\boldsymbol{\alpha}}_i$ 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 $\boldsymbol{p}_i$, 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 $\tilde{\boldsymbol{\alpha}}_i$ 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 <*query*, *ad*> 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 [30] models fine-tuned on the human-labeled training set, similar to what we do in knowledge distillation [31]. This allows us to experiment on a much larger scale, without breaking any assumptions in the EDL method. Without further

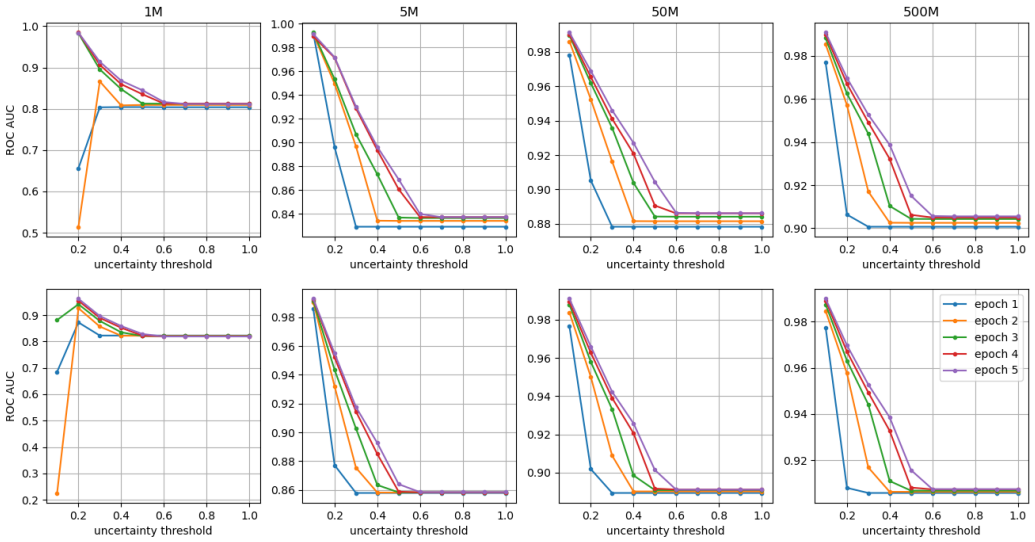

Figure 3: *ROC AUC* vs. uncertainty thresholds with 1M, 5M, 50M and 500M training corpus, respectively, and $\lambda = 0.1$. The first row is for EDL, while the second row is for TEDL. This figure shows that under a relatively small λ, the quality of uncertainty learnt by both EDL and TEDL improves as training proceeds.

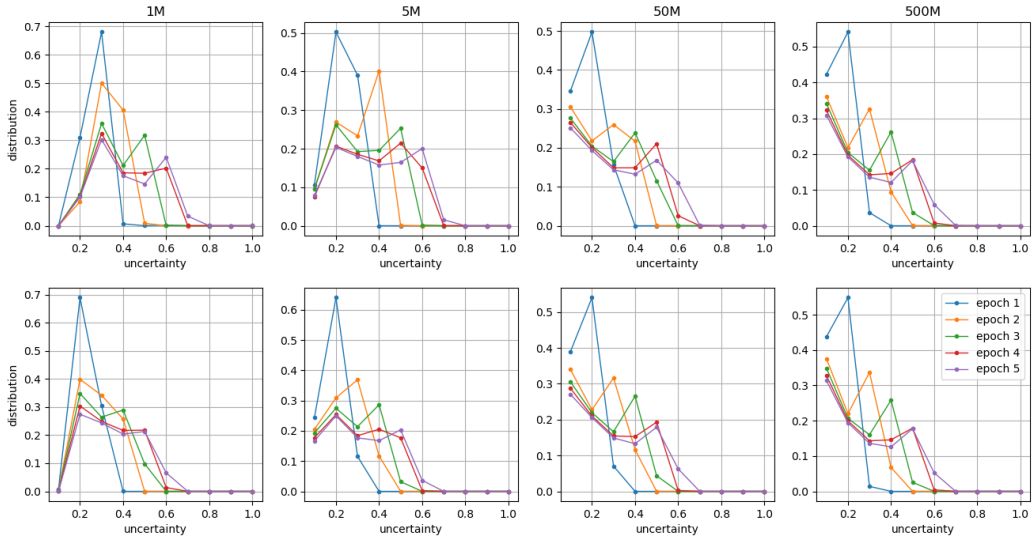

Figure 4: Uncertainty distribution of EDL (first row) and TEDL (second row), learnt on 1M, 5M, 50M and 500M training samples with $\lambda = 0.1$. The first row is for EDL, while the second row is for TEDL.

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 [32, 33] 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,

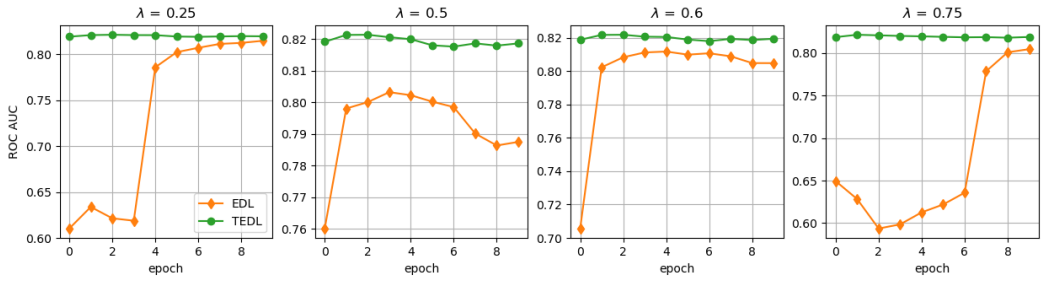

Figure 5: Comparison of *ROC AUC* for EDL and TEDL, learnt on 1M training corpus with different λ. Compared to EDL, TEDL not only achieves higher *ROC AUC*, but also shows improved robustness towards λ, especially when $\lambda = 0.75$ where EDL method runs into the *Dying ReLU* problem.

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 [2] 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, 50M and 500M training samples respectively. In all these 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 λ is used. Later in Section 4.3 we will see that compared with EDL, TEDL is much more robust towards λ. We also plot the distribution of uncertainty in each training epoch, as shown in Figure 4, where TEDL also looks similar to EDL when λ is relatively small, but later in Section 4.3 we will see their difference when λ 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, λ 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 λ 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 λ values. Compared with Figure 3 and Figure 4, the uncertainty quality learnt from EDL degrades dramatically when larger λ 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 λ.

5. Conclusion

In this paper, we propose TEDL, a two-stage learning approach to quantify uncertainty for deep classification models. TEDL

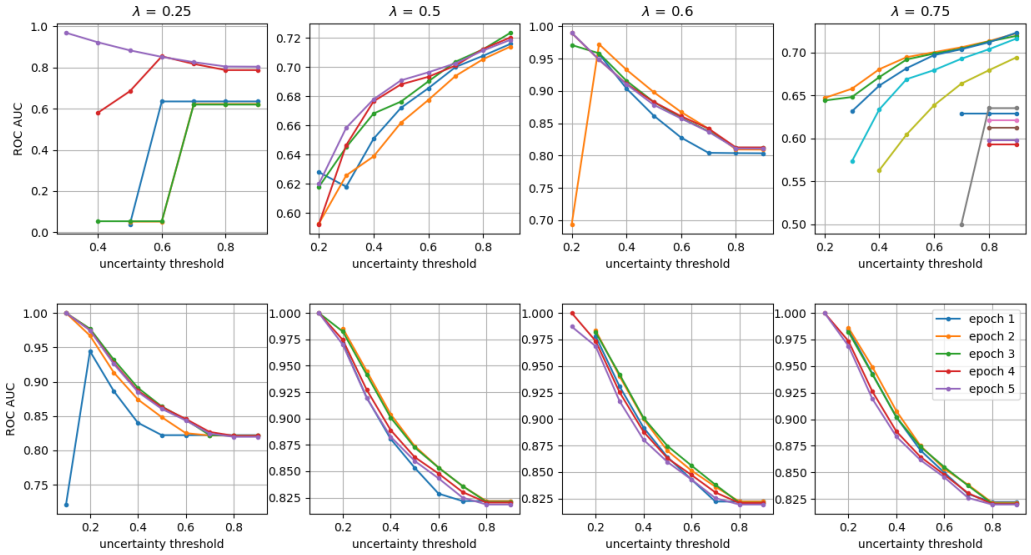

Figure 6: Comparison of *ROC AUC* vs. uncertainty for EDL (first row) and TEDL (second row), learnt on 1M training corpus with different λ, where TEDL shows significantly better robustness.

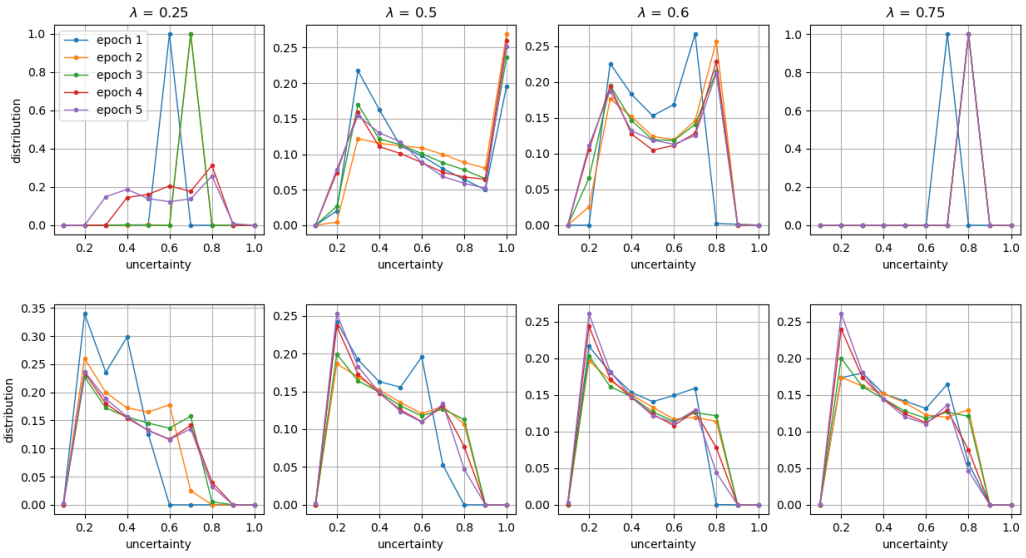

Figure 7: Comparison of uncertainty distribution for EDL (first row) and TEDL (second row), learnt on 1M training corpus with different λ, where TEDL shows significantly better robustness.

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 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 develop active learning algorithms.

References

[1] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, et al., A survey of uncertainty in deep neural networks, arXiv preprint arXiv:2107.03342 (2021).

[2] M. Sensoy, L. Kaplan, M. Kandemir, Evidential deep learning to quantify classification uncertainty, in: Advances in Neural Information Processing Systems, volume 31, 2018, pp. 3183–3193.

[3] A. Malinin, M. Gales, Predictive uncertainty estimation via prior networks, in: Advances in Neural Information Processing Systems, volume 31, 2018, pp. 7047–7058.

[4] J. Nandy, W. Hsu, M. L. Lee, Towards maximizing the representation gap between in-domain & out-of-distribution examples, in: Advances in Neural Information Processing Systems, volume 33, 2020, pp. 9239–9250.

[5] M. Możejko, M. Susik, R. Karczewski, Inhibited softmax for uncertainty estimation in neural networks, arXiv preprint arXiv:1810.01861 (2018).

[6] J. Lee, G. AlRegib, Gradients as a measure of uncertainty in neural networks, in: International Conference on Image Processing, IEEE, 2020, pp. 2416–2420.

[7] M. Raghu, K. Blumer, R. Sayres, Z. Obermeyer, B. Kleinberg, S. Mullainathan, J. Kleinberg, Direct uncertainty prediction for medical second opinions, in: International Conference on Machine Learning, PMLR, 2019, pp. 5281–5290.

[8] T. Ramalho, M. Miranda, Density estimation in representation space to predict model uncertainty, in: International Workshop on Engineering Dependable and Secure Machine Learning Systems, Springer, 2020, pp. 84–96.

[9] G. E. Hinton, D. Van Camp, Keeping the neural networks simple by minimizing the description length of the weights, in: Computational Learning Theory, 1993, pp. 5–13.

[10] Y. Gal, Z. Ghahramani, Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in: International Conference on Machine Learning, PMLR, 2016, pp. 1050–1059.

[11] C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra, Weight uncertainty in neural network, in: International Conference on Machine Learning, PMLR, 2015, pp. 1613–1622.

[12] A. Graves, Practical variational inference for neural networks, in: Advances in Neural Information Processing Systems, volume 24, 2011, pp. 2348–2356.

[13] C. Louizos, K. Ullrich, M. Welling, Bayesian compression for deep learning, in: Advances in Neural Information Processing Systems, volume 30, 2017, pp. 3288–3298.

[14] D. Rezende, S. Mohamed, Variational inference with normalizing flows, in: International Conference on Machine Learning, PMLR, 2015, pp. 1530–1538.

[15] D. Barber, C. M. Bishop, Ensemble learning in bayesian neural networks, Nato ASI Series F Computer and Systems Sciences 168 (1998) 215–238.

[16] R. M. Neal, An improved acceptance procedure for the hybrid monte carlo algorithm, Journal of Computational Physics 111 (1994) 194–203.

[17] R. M. Neal, Bayesian learning for neural networks, volume 118, Springer Science & Business Media, 2012.

[18] M. Welling, Y. W. Teh, Bayesian learning via stochastic gradient langevin dynamics, in: International Conference on Machine Learning, 2011, pp. 681–688.

[19] C. Nemeth, P. Fearnhead, Stochastic gradient markov chain monte carlo, Journal of the American Statistical Association 116 (2021) 433–450.

[20] T. Salimans, D. P. Kingma, Weight normalization: A simple reparameterization to accelerate training of deep neural networks, in: Advances in Neural Information Processing Systems, volume 29, 2016, pp. 901–909.

[21] J. Lee, M. Humt, J. Feng, R. Triebel, Estimating model uncertainty of neural networks in sparse information form, in: International Conference on Machine Learning, PMLR, 2020, pp. 5702–5713.

[22] H. Ritter, A. Botev, D. Barber, A scalable laplace approximation for neural networks, in: International Conference on Learning Representations, volume 6, 2018.

[23] B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, in: Advances in Neural Information Processing Systems, volume 30, 2017, pp. 6404–6416.

[24] H. Guo, H. Liu, R. Li, C. Wu, Y. Guo, M. Xu, Margin & diversity based ordering ensemble pruning, Neurocomputing 275 (2018) 237–246.

[25] W. G. Martinez, Ensemble pruning via quadratic margin maximization, IEEE Access 9 (2021) 48931–48951.

[26] J. Lindqvist, A. Olmin, F. Lindsten, L. Svensson, A general framework for ensemble distribution distillation, in: International Workshop on Machine Learning for Signal Processing, IEEE, 2020, pp. 1–6.

[27] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, Q. He, A comprehensive survey on transfer learning, Proceedings of the IEEE 109 (2021) 43–76.

[28] V. Dang, M. Bendersky, W. B. Croft, Two-stage learning to rank for information retrieval, in: European Conference on Information Retrieval, Springer, 2013, pp. 423–434.

[29] F. A. Khan, A. Gumaei, A. Derhab, A. Hussain, A novel two-stage deep learning model for efficient network intrusion detection, IEEE Access 7 (2019) 30373–30385.

[30] J. D. M.-W. C. Kenton, L. K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of NAACL-HLT, 2019, pp. 4171–4186.

[31] X. Li, Z. Luo, H. Sun, J. Zhang, W. Han, X. Chu, L. Zhang, Q. Zhang, Learning fast matching models from weak annotations, in: Proceedings of the Web Conference, 2019, pp. 2985–2991.

[32] W. Lu, J. Jiao, R. Zhang, Twinbert: Distilling knowledge to twin-structured compressed bert models for large-scale retrieval, in: Proceedings of the ACM International Conference on Information & Knowledge Management, 2020, pp. 2645–2652.

[33] J. Zhu, Y. Cui, Y. Liu, H. Sun, X. Li, M. Pelger, T. Yang, L. Zhang, R. Zhang, H. Zhao, Textgnn: Improving text encoder via graph neural network in sponsored search, in: Proceedings of the Web Conference, 2021, pp. 2848–2857.