

Article

Revisiting Softmax for Uncertainty Approximation in Text Classification

Andreas Nugaard Holm , Dustin Wright *  and Isabelle Augenstein 

Department of Computer Science, University of Copenhagen, 1172 Copenhagen, Denmark; aholm@di.ku.dk (A.N.H.); augenstein@di.ku.dk (I.A.)

* Correspondence: dw@di.ku.dk

Abstract: Uncertainty approximation in text classification is an important area with applications in domain adaptation and interpretability. One of the most widely used uncertainty approximation methods is Monte Carlo (MC) dropout, which is computationally expensive as it requires multiple forward passes through the model. A cheaper alternative is to simply use a softmax based on a single forward pass without dropout to estimate model uncertainty. However, prior work has indicated that these predictions tend to be overconfident. In this paper, we perform a thorough empirical analysis of these methods on five datasets with two base neural architectures in order to identify the trade-offs between the two. We compare both softmax and an efficient version of MC dropout on their uncertainty approximations and downstream text classification performance, while weighing their runtime (cost) against performance (benefit). We find that, while MC dropout produces the best uncertainty approximations, using a simple softmax leads to competitive, and in some cases better, uncertainty estimation for text classification at a much lower computational cost, suggesting that softmax can in fact be a sufficient uncertainty estimate when computational resources are a concern.

Keywords: text classification; uncertainty quantification; efficiency



Citation: Holm, A.N.; Wright, D.; Augenstein, I. Revisiting Softmax for Uncertainty Approximation in Text Classification. *Information* **2023**, *14*, 420. <https://doi.org/10.3390/info14070420>

Academic Editors: Katsuhide Fujita and Ralf Krestel

Received: 19 May 2023

Revised: 3 July 2023

Accepted: 18 July 2023

Published: 20 July 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The pursuit of pushing state-of-the-art performance on machine learning benchmarks often comes with an added cost of computational complexity. On top of already complex base models, such as transformer models [1,2], successful methods often employ additional techniques to improve the uncertainty estimation of these models, as they tend to be overconfident in their predictions. Though these techniques can be effective, the overall benefit in relation to the added computational cost is under-studied.

More complexity does not always imply better performance. For example, transformers can be outperformed by much simpler convolutional neural nets (CNNs) when the latter are pre-trained as well [3]. Here, we turn our attention to neural network uncertainty estimation methods in text classification, which have applications in domain adaptation and decision making, and can help make models more transparent and explainable. In particular, we focus on a setting where efficiency is of concern, which can help improve the sustainability and democratisation of machine learning, as well as enable use in resource-constrained environments.

Quantifying predictive uncertainty in neural nets has been explored using various techniques [4], with the methods being divided into three main categories: Bayesian methods, single deterministic networks, and ensemble methods. Bayesian methods include Monte Carlo (MC) dropout [5] and Bayes by back-prop [6]. Single deterministic networks can approximate the predictive uncertainty by a single forward pass in the model, with softmax being the prototypical method. Lastly, ensemble methods utilise a collection of models to calculate the predictive uncertainty. However, while uncertainty estimation can improve when using more complex Bayesian and ensembling techniques, efficiency takes a hit.

In this paper, we perform an empirical investigation of the trade-off between choosing cheap vs. expensive uncertainty approximation methods for text classification, with the goal of highlighting the efficacy of these methods in an efficient setting. We focus on one single deterministic and one Bayesian method. For the single deterministic method, we study the softmax, which is calculated from a single forward pass and is computationally very efficient. While softmax is a widely used method, prior work has posited that the softmax output, when taken as a single deterministic operation, is not the most dependable uncertainty approximation method [5,7]. As such, it has been superseded by newer methods such as MC dropout, which leverages the dropout function in neural nets to approximate a random sample of multiple networks and aggregates the softmax outputs of this sample. MC dropout is favoured due to its close approximation of uncertainty, and because it can be used without any modification to the applied model. It has also been widely applied in text classification tasks [8,9].

To understand the cost vs. benefit of softmax vs. MC dropout, we perform experiments on five datasets using two different neural network architectures, applying them to three different downstream text classification tasks. We measure both the added computational complexity in the form of runtime (cost) and the downstream performance on multiple uncertainty metrics (benefit). We show that by using a single deterministic method like softmax, instead of MC dropout, we can improve the runtime by 10 times while still providing reasonable uncertainty estimates on the studied tasks. As such, given the already high computational cost of deep-neural-network-based methods and recent pushes for more sustainable ML [10,11], we recommend not discarding efficient uncertainty approximation methods such as softmax in resource-constrained settings, as they can still potentially provide reasonable estimations of uncertainty.

Contribution In summary, our contributions are: (1) an empirical study of an efficient version of MC dropout and softmax for text classification tasks, using two different neural architectures, and five datasets; (2) a comparison of uncertainty estimation between MC dropout and softmax using expected calibration error; and (3) a comparison of the cost vs. benefit of MC dropout and softmax in a setting where efficiency is of concern.

2. Related Work

2.1. Uncertainty Quantification

Quantifying the uncertainty of a prediction can be performed using various techniques [4,12,13], such as single deterministic methods [14,15], which calculate the uncertainty on a single forward pass of the model. They can further be classified as internal or external methods, which describe if the uncertainty is calculated internally in the model or post-processing the output. Another family of techniques are Bayesian methods, which combine NNs and Bayesian learning. Bayesian neural networks (BNNs) can also be split into subcategories, namely variational inference [16], sampling [17], and Laplace approximation [18]. Some of the more notable methods are Bayes by backprop [6] and Monte Carlo dropout [5]. One can also approximate uncertainty using ensemble methods, which use multiple models to better measure predictive uncertainty, compared to using the predictive uncertainty given by a single model [9,19,20]. Recently, we have seen uncertainty methods being used to develop methods for new tasks [8,9], where mainly Bayesian methods have been used. We present a thorough empirical study of how uncertainty quantification behaves for text classification tasks. Unlike prior work, we do not only evaluate based on the performance of the methods, but perform an in-depth comparison to much simpler deterministic methods based on multiple metrics.

2.2. Uncertainty Metrics

Measuring the performance of uncertainty approximation methods can be performed in multiple ways, each offering benefits and downsides. Niculescu-Mizil and Caruana [21] explore the use of obtaining confidence values from model predictions to use for supervised learning. One of the more widespread and accepted methods is using expected calibra-

tion error (ECE, Guo et al. [22]), while ECE measures the underlying confidence of the uncertainty approximation, we have also seen the use of human intervention for text classification [8,9]. There, the uncertainty estimates are used to identify uncertain predictions from the model and ask humans to classify these predictions. The human-classified data are assumed to have 100% accuracy and to be suitable for measuring how well the model scores after removing a proportion of the most uncertain data points. Using metrics such as ECE, the calibration of models is shown, and this calibration can be improved using scaling techniques [22,23]. We use uncertainty approximation metrics like expected calibration error and human intervention (which we refer to as holdout experiments) to measure the difference in the performance of MC dropout and softmax compared against each other on text classification tasks.

3. Uncertainty Approximation for Text Classification

We focus on one deterministic method and one Bayesian method of uncertainty approximation. Both methods assume the existence of an already-trained base model, and are applied at test time to obtain uncertainty estimates from the model’s predictions. In the following sections, we formally introduce the two methods we study, namely MC dropout and softmax. MC dropout is a Bayesian method which utilises the dropout layers of the model to measure the predictive uncertainty, while softmax is a deterministic method that uses the classification output. In Figure 1, we visualise the differences between the two methods and how they are connected to base text classification models.

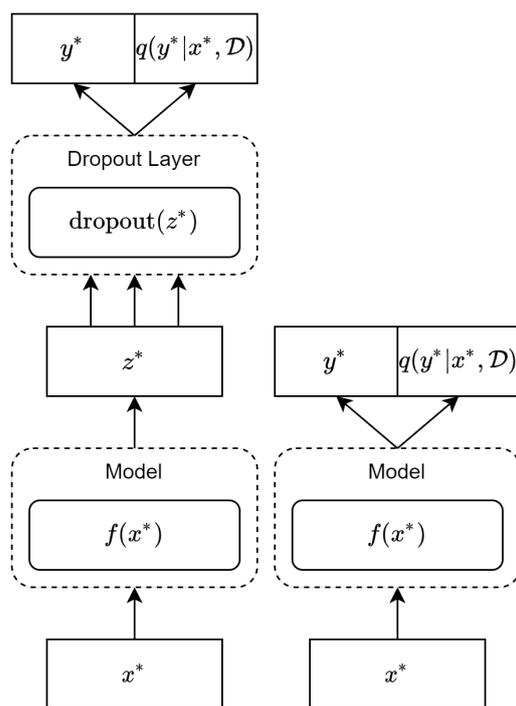


Figure 1. MC dropout (left) and softmax (right). In the version of MC dropout tested in this paper, a test input x^* is passed through model f to obtain a representation z^* , which is then subsequently passed through a dropout layer multiple times, and passed through the final part of the network to obtain prediction y^* . For softmax, dropout is disabled and a single prediction is obtained.

3.1. Bayesian Learning

Before introducing the MC dropout method, we quickly introduce the concept of *Bayesian learning*. We start by comparing Bayesian learning to a traditional NN. A traditional NN assumes that the network weights $\omega \in \mathbb{R}^n$ are real but of an unknown value and can be found through maximum-likelihood estimation, and the input data $(x, y) \in \mathcal{D}$ are treated as random variables. Bayesian learning instead views the weights as random variables, and

infers a posterior distribution $p(\omega|\mathcal{D})$ over ω after observing \mathcal{D} . The posterior distribution is defined as follows:

$$p(\omega|\mathcal{D}) = \frac{p(\omega)p(\mathcal{D}|\omega)}{p(\mathcal{D})} = \frac{p(\omega)p(\mathcal{D}|\omega)}{\int p(\omega)p(\mathcal{D}|\omega)d\omega} \tag{1}$$

Using the posterior distribution, we can find the prediction of an input of unseen data x^* and y^* as follows:

$$p(y^*|x^*, \mathcal{D}) = \int p(y^*|x^*, \omega)p(\omega|\mathcal{D})d\omega. \tag{2}$$

However, the posterior distribution is infeasible to compute due to the marginal likelihood in the denominator, so we cannot find a solution analytically. We therefore resort to approximating the posterior distribution. For this approximation, we rely on methods such as Bayes by backpropagation [6] and Monte Carlo dropout [5].

3.2. Monte Carlo Dropout

At a high level, MC dropout approximates the posterior distribution $p(\omega|\mathcal{D})$ by leveraging the dropout layers in a model [5,24]. Mathematically, it is derived by introducing a distribution $q(\omega)$, representing a distribution of weight matrices whose columns are randomly set to 0, to approximate the posterior distribution $p(\omega|\mathcal{D})$, which results in the following predictive distribution:

$$q(y^* | x^*, \mathcal{D}) = \int p(y^*|x^*, \omega)q(\omega)d\omega. \tag{3}$$

As this integral is still intractable, it is approximated by taking K samples from $q(\omega)$ using the dropout layers of a learned network f , which approximates $p(y^*|x^*, \omega)$. As such, calculating $p(y^*|x^*, \omega)q(\omega)$ amounts to leaving the dropout layers active during testing, and approximating the integral amounts to aggregating predictions across multiple dropout samples. For the proofs, see Gal and Ghahramani [5].

MC dropout requires multiple forward passes, so its computational cost is a multiple of the cost of performing a forward pass through the entire network. As this is obviously more computationally expensive than the single forward pass required for deterministic methods, we provide a fairer comparison between softmax and MC dropout by using an efficient version of MC dropout which caches an intermediate representation and only activates the dropout layers of the latter part of the network. As such, we obtain a representation z^* by passing an input through the first several layers of the model, and pass only this representation through the latter part of the model multiple times, reducing the computational cost while approximating the sampling of multiple networks.

Combining Sample Predictions

With multiple samples of the same data point, we have to determine how to combine them to quantify the predictive uncertainty. We test two methods that can be calculated using the logits of the model, requiring no model changes. The first approach, which we refer to as mean MC, is averaging the output of the softmax layer from all forward passes:

$$u_i = \frac{1}{K} \sum_{k=1}^K \text{Softmax}(f(z_i^k)), \tag{4}$$

where z_i^k is a representation of the i 'th data point of the k 'th forward pass, and f is a fully-connected layer. The second method we use to quantify the predictive uncertainty is dropout entropy (DE) [8], which uses a combination of binning and entropy:

$$b_i = \frac{1}{K} \text{BinCount}(\text{argmax}(f(z_i))) \quad (5)$$

$$u_i = - \sum_{j=1}^C b_i(j) \log b_i(j) \quad (6)$$

where BinCount is the number of predictions of each class and b is a vector the probabilities of a class's occurrence based on the bin count. We show the performance of the two methods in Section 4.3.2.

3.3. Softmax

Softmax, a common normalising function for producing a probability distribution from neural network logits, is defined as follows:

$$u_i = \frac{e^{z_i}}{\sum_{j=1}^C e^{z_i(j)}}, \quad (7)$$

where z_i are the logits of the i 'th data point. The softmax yields a probability distribution over the predicted classes. However, the predicted probability distribution is often overconfident toward the predicted class [5,7]. The issue of softmax's overconfidence can also be exploited [5,25]—in the worst case, this leads to the softmax producing imprecise uncertainties. However, model calibration methods like temperature scaling have been found to lessen the overconfidence to some extent [22]. As temperature scaling also incurs a cost in terms of runtime in order to find an optimal temperature, we choose to compare raw softmax probabilities to the efficient MC dropout method described previously, though uncertainty estimation could potentially be improved by scaling the logits appropriately.

4. Experiments and Results

We consider five different datasets and two different base models in our experiments. Additionally, we conduct experiments to determine the optimal hyperparameters for the MC dropout method, particularly the optimal amount of samples which affects the efficiency and performance of MC dropout. In the paper, we focus on the results of the 20 Newsgroups dataset; the results of the other four datasets are shown in the Appendices B and C. We further find the optimal dropout percentage in Appendix A.3.

4.1. Data

To test the predictive uncertainty of the two methods, we use five datasets for diverse text classification tasks. We use the following five datasets: The 20 Newsgroups dataset [26] is a text classification consisting of a collection of 20,000 news articles. The news articles are classified into 20 different classes. The Amazon dataset [27] is a sentiment classification task. We use the 'sports and outdoors' category, which consists of 272,630 reviews ranging from 1 to 5. The IMDb dataset [28] is also a sentiment classification task. However, compared to the Amazon dataset, this is a binary problem. The dataset consists of 50,000 reviews. The SST-2 dataset [29] is also a binary sentiment classification dataset, consisting of 70,042 sentences. Lastly, we also use the Wiki dataset [30], which is a citation needed task, i.e., we predict if a citation is needed. The dataset consists of 19,998 texts. For the 20 Newsgroups, Amazon, IMDb, and Wiki datasets, we use a split of 60, 20 and 20 for the training, validation and test data, the data in splits have been selected randomly. We used the provided splits for the SST-2 dataset, but due to the test labels being hidden, we used the validation set for testing. We select these datasets as they are large, the tasks are diverse, and they cover multiple domains of text. Additionally, they represent well-studied and standard benchmarks in the field of text classification, which helps with the reproducibility of the results and comparison with baselines.

4.2. Experimental Setup

We use two different base neural architectures with two different embeddings in our experiments. To recreate baseline results, the first model is the same model as proposed in [8], which is a CNN using pre-trained GloVe embeddings (Glove-CNN) with a dimension of 200 [31]. The second model uses a pre-trained BERT model [32] fine-tuned as a masked language model on the dataset under evaluation to obtain contextualised embeddings, which are then input to a CNN with 4 layers (BERT-CNN). The selection of these models allows us to compare the established baseline architecture from [8] with a more modern version of it which takes advantage of large language models. For both models, we use the final dropout layer for MC dropout. Both models are optimised using Adam [33] and are trained for 1000 epochs, with early stopping after 10 iterations if there have been no improvements, and we set the learning rate to 0.001.

MC Dropout Sampling

To make full use of MC dropout, we first determine the optimal number of forward passes through the model needed to obtain the best performance while maintaining high efficiency. This hyper-parameter search is imperative because the MC dropout performance and efficiency are correlated with the number of samples generated. To make a fair comparison against the already cheap softmax method, we want to find the minimum number of samples needed to approximate a good uncertainty. In Table 1, we show the performance, using the F1 score, of the MC dropout method with the BERT-CNN model on the 20 Newsgroups dataset for the following number of samples: [1, 5, 10, 25, 50, 100, 1000]. The table shows how the performance of the uncertainty approximation increases, given the number of samples. However, the performance gained by the number of samples falls off at 50. Given this, we use 50 MC samples in our experiments in order to balance good performance and efficiency.

Table 1. This table shows how the number of samples affect the performance of the MC dropout method, on the 20 Newsgroups dataset, using the BERT-CNN model. The results are reported using macro F1.

1	10	25	50	100	1000
0.8212	0.8623	0.8540	0.8591	0.8559	0.8573

4.3. Evaluation Metrics

We use complementary evaluation metrics to benchmark the performance of MC dropout and softmax. Namely, we measure how well each of the methods identify uncertain predictions as well as the runtime of the methods.

4.3.1. Efficiency

To quantify efficiency, we measure the runtime of each of the methods during inference and the calculation of the uncertainties. Since we do not calculate uncertainties during training, this is only performed on the test sets. Training the model is independent of the uncertainty estimation methods, since we only use them to quantify the uncertainty of the predictions of the model. We therefore only calculate the runtime of each of the methods based on the test data.

4.3.2. Performance Metrics

We use two main uncertainty metrics: test data holdout and expected calibration error (ECE). These metrics give us an estimation of the *epistemic* uncertainty of the model, i.e., the lack of certainty inherent in the model and its predictions. We do not cover metrics of *aleatoric* uncertainty in this paper, which focus on the inherent randomness of the data itself and which could be tested through the introduction of, e.g., label noise. For base model

performance, we record the macro F1 score on the 20 Newsgroups, IMDb, Wiki, and SST-2 datasets, and the accuracy on the Amazon dataset.

Test data holdout: This metric ranks all samples based on the predictive uncertainty, and calculates the F1 and accuracy scores on a percentage of the samples by removing those which the model is least certain about. In other words, a method is better if it achieves a greater improvement in performance metrics (e.g., F1) when removing the most uncertain samples. As such, this metric expresses the relationship between model calibration and accuracy. We choose to remove 10%, 20%, 30% and 40% of the least certain samples for our experiments. This metric shows how well the two methods can identify uncertain predictions of the model, as reflected by improvements in performance when more uncertain predictions are removed [8]. In our experiments, we use the aforementioned mean MC, DE, and softmax method to calculate the uncertainties; we further add the penultimate layer variance (PL-Variance), where the PL-Variance utilises the variance of the last fully-connected layer as the uncertainty [34].

Expected calibration error: As a second uncertainty estimation metric, we use the expected calibration error (ECE, Guo et al. [22]), which measures, in expectation, how confident the predictions for both correct and incorrect predictions are. This tells us how well each of the MC dropout and softmax methods estimate the uncertainties at the level of probability distributions, as opposed to the holdout method which only looks at downstream task performance. ECE works by dividing the data into m bins, where each bin in B contains data that is within a certain range of probabilities, using the probability of the predicted class. Formally, ECE is defined as:

$$ECE = \sum_{m=1}^M \frac{|B_m|}{n} |acc(B_m) - conf(B_m)| \quad (8)$$

where M is the size of the dataset and acc and $conf$ is the accuracy and mean confidence (i.e., predicted class probabilities) of the bin B_m .

Finally, to visualise the difference between the MC dropout and softmax, we create both confidence histograms and reliability diagrams [22]. The reliability diagrams show how close the models are to perfect calibration, where perfect calibration means that the models accuracy and confidence is equal to the bins confidence range. In all cases, we show reliability diagrams by comparing histograms of accuracy and confidence across confidence bins; as such, when confidence exceeds accuracy in a given bin, that indicates how overconfident the model is for that bin. The reliability diagrams help us visualise the ECE, by showing the accuracy and mean confidence of each bin, where each bin consists of the data which have a confidence within the range of the bin. To complement the reliability diagrams, we also use confidence histograms, which show the distribution of confidence.

4.4. Efficiency Results

In Table 2, we display the runtime of the different model and method combinations. The runtime for the forward passes is calculated as a sum of all the forward passes on the entire dataset, and the runtime for the uncertainty methods are calculated for the entire dataset. Observing the results, we see that softmax is overall faster, and is approximately 10 times faster when only looking at the forward passes, and using more complex aggregation methods in MC dropout, like DE, can be computationally heavy.

Table 2. Runtime measured in seconds for both MC dropout (top) and softmax (bottom). The times are on the full datasets split into the runtime of the forward passes and the runtime of calculating the uncertainty.

	Forward Passes	Mean MC	DE
20 Newsgroups	1.0876	0.0003	12.3537
IMDb	1.386	0.0018	216.11
Amazon	4.9126	0.0017	194.08
WIKI	1.1149	0.0010	15.8467
SST-2	1.0076	0.0003	3.4785
	Forward Passes	Softmax	PL-Variance
20 Newsgroups	0.0130	0.0002	0.0001
IMDb	0.0387	0.0003	0.0003
Amazon	0.4067	0.0004	0.0002
WIKI	0.0149	0.0002	0.0001
SST-2	0.0037	0.0002	0.0001

4.5. Test Data Holdout Results

Table 3 and the table in Appendix B show the performance of the two uncertainty approximation methods using the different datasets and models. The tables show the macro F1 score and accuracy (depending on the datasets), and the ratio of improvement from holding out data in parentheses. We observe that, in most cases, either dropout entropy (DE) or softmax has the highest score and improvement ratio. However, in most cases the two are close in performance and improvement ratio. We further observe that mean MC also performs well and is almost on par with DE; however, mean MC is a much more efficient method compared to DE, so the slight trade-off in performance could be beneficial in resource-constrained settings or non-critical applications.

Table 3. Macro F1 score and improvement rate for the 20 Newsgroups dataset.

BERT	0%	10%	20%	30%	40%
Mean MC	0.8591	0.8985 (1.0459)	0.9225 (1.0739)	0.9406 (1.0949)	0.9487 (1.1043)
DE	0.8591	0.9050 (1.0534)	0.9390 (1.0930)	0.9584 (1.1156)	0.9703 (1.1294)
Softmax	0.8576	0.9072 (1.0578)	0.9452 (1.1021)	0.9620 (1.1216)	0.9742 (1.1360)
PL-Variance	0.8576	0.9006 (1.0501)	0.9246 (1.0781)	0.9403 (1.0964)	0.9484 (1.1058)
GloVe					
Mean MC	0.7966	0.8450 (1.0608)	0.8674 (1.0888)	0.8846 (1.1104)	0.8960 (1.1248)
DE	0.7966	0.8469 (1.0631)	0.8855 (1.1116)	0.9155 (1.1492)	0.9416 (1.1820)
Softmax	0.7959	0.8465 (1.0636)	0.8846 (1.1115)	0.9149 (1.1496)	0.9402 (1.1813)
PL-Variance	0.7959	0.8436 (1.0599)	0.8667 (1.0891)	0.8848 (1.1118)	0.8966 (1.1266)

4.6. Model Calibration Results

To further investigate the differences between MC dropout and softmax, we utilise the expected calibration error (ECE) to observe the differences in the predictive uncertainties. In Table 4, we show the accuracy and ECE on the three datasets using the BERT embeddings.

The results from our holdout experiments in Table 3 and in Appendix B, combined with the results from our ECE calculations in Table 4, all point in the direction of the efficient MC dropout used in this study and softmax performing on par to each other, but with a large gap in runtime as shown in Table 2. To obtain a better understanding of if and where the two methods diverge, we plot the reliability diagrams and confidence histograms as described in Section 4.3.2.

Table 4. Accuracy and ECE of the two uncertainty approximation approaches on the three selected datasets.

	Accuracy	ECE
20 Newsgroups—Mean MC	0.8655	0.0275
20 Newsgroups—Softmax	0.8642	0.0253
IMDb—Mean MC	0.9354	0.0061
IMDb—Softmax	0.9364	0.0043
Amazon—Mean MC	0.7466	0.0083
Amazon—Softmax	0.7474	0.0097
WIKI—Mean MC	0.9227	0.0370
WIKI—Softmax	0.9230	0.0279
SST-2—Mean MC	0.7408	0.0535
SST-2—Softmax	0.7442	0.0472

Plot description: In Figures 2 and 3, we show the reliability diagrams and the confidence histograms on the 20 Newsgroups dataset using both our BERT-CNN and GloVe-CNN with both the MC dropout method and softmax. We create the reliability diagrams using 10 bins and the confidence histograms with 20. The reliability diagram’s and confidence histogram’s bins are an interval of confidence. We use 20 bins for the confidence histograms to obtain a more fine-grained view of the distribution. In the reliability diagram, the *x*-axis is the confidence and the *y*-axis is the accuracy. For the confidence histogram the *x*-axis is again the confidence and the *y*-axis is the percentage of the samples in the given bin.

Expectations: While ECE can quantify the performance of the models on a somewhat lower level than our other metrics, the metric can be deceived, especially in cases where models score high in accuracy. It will favour overconfident models; therefore, we expect the results to favour softmax. Looking at the ECE, we can observe that it will favour an overconfident method when the model achieves high accuracy. With this in mind, we expect the results to be skewed towards the softmax.

Observations reliability diagram: From the reliability diagram, we observe that the difference in confidence and outputs are small. The difference between the two uncertainty methods is also minimal, including both BERT and GloVe embeddings, suggesting minimal potential gains from using MC dropout in an efficient setting while still incurring a high cost in terms of runtime. We determine that there is minimal difference by visually inspecting the plots, and by observing the ECE displayed in Table 4. We further observe that in both MC dropout and softmax that the model worsens when we use the GloVe embeddings.

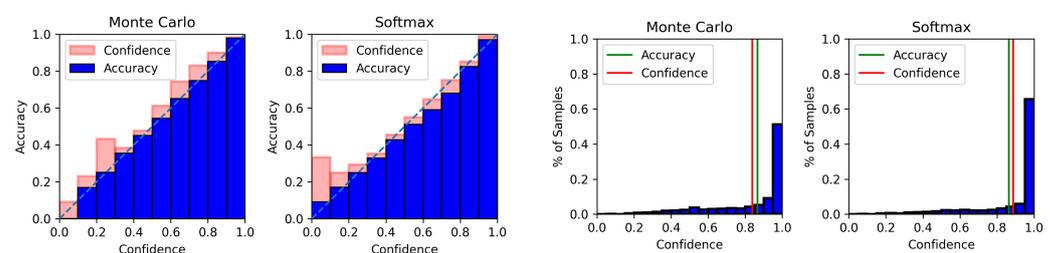


Figure 2. Reliability diagram (left, displayed as a stacked bar chart comparing accuracy and confidence) and confidence histogram (right) of 20 Newsgroups using BERT-CNN. Softmax and the efficient version of MC dropout tested in this paper are relatively similar in their calibration (a higher value for confidence than accuracy in any bin indicates overconfidence in that bin). At the same time, as indicated by the confidence histogram, softmax still produces more confident estimates on average.

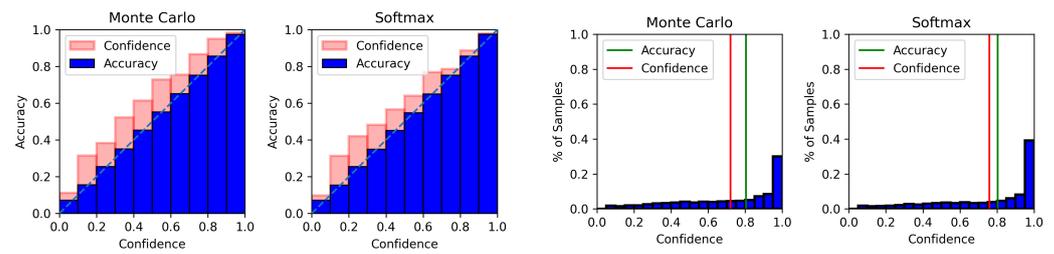


Figure 3. Reliability diagram (left, displayed as a stacked bar chart comparing accuracy and confidence) and confidence histogram (right) of 20 Newsgroups using GloVe-CNN. Comparing the plots of the figure to Figure 2, we see slight differences in both the reliability diagram and the confidence histogram. Most noticeably, we see slight differences in the reliability diagram, where we see more significant gaps between the confidence and the outputs, which indicates a less calibrated model due to the GloVe embeddings.

Observations confidence histogram: As mentioned earlier, we know that the softmax tends to be overconfident, which can be seen in the percentage of samples in the last bin. The MC dropout method, on the other hand, utilises the probability space to a greater extent. We include reliability diagrams and confidence histograms for the two other datasets in Appendix C.

Noise experiment: Inspecting both Table 4 showing the ECE values, and the performances in Tables 3, A2 and A3, we observe that using our two uncertainty estimation methods, we achieved very high F1 scores and accuracies and low ECEs. We hypothesised that high performance could lead to softmax achieving high ECE, due to naturally having high confidence, compared to MC dropout. We added zero-mean Gaussian noise to the 20 Newsgroups test embeddings and reperformed our ECE experiments to test our hypothesis. In Figure 4, we show the reliability diagram of the experiment with added noise, which shows the MC dropout outperforming softmax. To further build on the theory, we also inspect the confidence histogram, showing that softmax is still overconfident and the difference between the accuracy and mean confidence is high. This suggests that MC dropout is more resilient to noise and, in cases where the performance of a model is low, MC dropout could potentially obtain more precise predictive uncertainties.

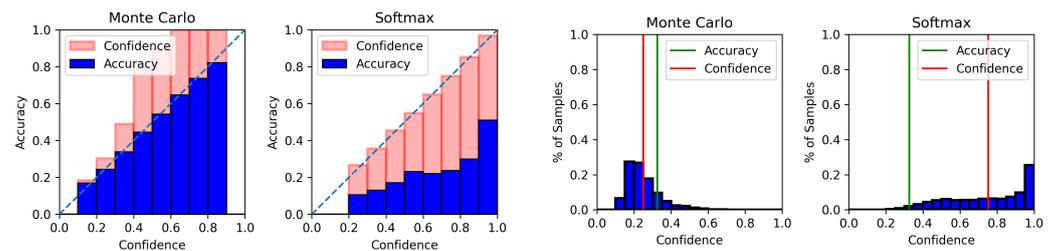


Figure 4. Reliability diagram of 20 Newsgroups dataset (displayed as a stacked bar chart comparing accuracy and confidence) using the BERT-CNN model, with added zero-mean Gaussian noise to the BERT embeddings. Softmax is highly overconfident compared to MC dropout (despite the efficient setting in this paper where only the final layers of the model are used for dropout), as indicated by the large gap between average confidence and accuracy in each bin of the histogram.

5. Discussion and Conclusions

In this paper, we perform an in-depth empirical comparison of using the MC dropout method in an efficient setting and the more straightforward softmax method. By doing a thorough empirical analysis of the two methods, shown in Section 4.3.2, using various metrics to measure their performance on both efficiency and performance levels, we see that in our holdout experiments in Table 3, the two methods perform approximately the same. Looking at the expected calibration error (ECE) experiments, the results again show that the MC dropout and softmax method perform somewhat equally, which we

have shown in Section 4.6. We observe differences in the results as we observe a lower accuracy score, which we show in our noise experiment, which is also shown in Section 4.6. Prior research [7] investigated out-of-distribution analysis and found that softmax, both for sentiment classification and text categorisation tasks, can detect out-of-distribution data points efficiently. It further showcases that in these two tasks, the softmax can also, to some extent, perform well as a confidence estimator. While we show that the two methods perform almost equally, when comparing the predictive performance, the cost of using MC dropout is at a minimum 10 times that of running softmax, even in the efficient setting where only the final layer is dropped out, depending on the post-processing of the uncertainties, as we show in Section 4.4. The post-processing cost of MC dropout can quickly explode when used on larger datasets or if a more expensive method like dropout entropy is used instead of simpler approaches.

Given this, when could it be appropriate to use the more efficient softmax over MC dropout for estimating predictive uncertainty? Our results suggest that when the base accuracy of a model is high, the differences in uncertainty estimation between the two methods is relatively low, likely due to the higher confidence of the softmax method. In this case, if latency or resource efficiency is a concern such as on edge devices, it may be appropriate to rely on a quick estimate using softmax as opposed to a more cumbersome method. However, when model accuracy is expected to be low, softmax is still overconfident compared to MC dropout, so estimates using a single deterministic softmax may be unreliable. The downstream application may also impact this; in critical scenarios such as health care, it may still be more appropriate to use an inefficient method with better predictive uncertainty for improved decision-making. In low-risk applications where models are known to be accurate and efficiency is of concern, we have demonstrated that softmax can potentially be sufficient.

6. Limitations

We highlight a few key limitations of the study to further contextualise the work. First, we note that the study is restricted to neural-network-based methods, while other methods in ML may be useful to study for uncertainty estimation as well. Second, we note that we test a plain softmax method without temperature scaling—while calibrating a useful temperature could induce a cost in terms of time, it would potentially lead to better uncertainty estimation. Finally, we note that we also test an efficient form of MC dropout which only drops out a portion of the network; while this demonstrates that in an efficient setting, softmax can be as good or better at uncertainty estimation than MC dropout, full MC dropout still may have better uncertainty estimation when efficiency is not a concern.

Author Contributions: Conceptualisation, A.N.H., D.W. and I.A.; methodology, A.N.H.; software, A.N.H.; validation, A.N.H.; formal analysis, A.N.H.; investigation, A.N.H.; data curation, A.N.H.; writing—original draft preparation, A.N.H., D.W. and I.A.; writing—review and editing, A.N.H., D.W. and I.A.; supervision, I.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Innovation Fund Denmark grant number 9065-00131B.

Data Availability Statement: All data used is open-source and can be found in the referenced papers.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Reproducibility

Appendix A.1. Computing Infrastructure

All experiments were run on a Microsoft Azure NC6-series server. With the following specifications: 6 Inter Xeon-E5-2690 v3, NVIDIA Tesla K80 with 12 GB RAM and 56 GB of RAM.

Appendix A.2. Hyperparameters

We used the following hyperparameters for training our CNN model and CNN GloVe model: epochs: 1000; batch size: 256 for 20 Newsgroups, IMDb SST-2 and Wiki, and 128 for Amazon; early stopping: 10; learning rate: 0.001. For fine-tuning BERT, we used the following set of hyperparameters: epochs: 3; warm-up steps 500; weight decay 0.01; batch size 8; masked language model probability: 0.15. All hyperparameters are set without performing cross-validation.

Appendix A.3. Dropout Hyperparameters

The performance of the MC dropout method is correlated with the dropout probability. We therefore run our CNN model using BERT embeddings on the 20 Newsgroups dataset with the following dropout probabilities [0.1, 0.2, 0.3, 0.4, 0.5]. In Table A1, we show the results using the five different dropout probabilities, where we see that it stops improving at 0.4 and 0.5 percentage dropout. As such, we use a dropout of 0.5 for our experiments.

Table A1. We test how the dropout probabilities correlate with the performance of MC dropout, using the BERT-CNN model. The results are reported in terms of macro F1.

	0%	10%	20%	30%	40%
0.1	0.8598	0.9010	0.9255	0.9408	0.9483
0.2	0.8599	0.9005	0.9256	0.9408	0.9502
0.3	0.8596	0.9007	0.9245	0.9412	0.9491
0.4	0.8601	0.8996	0.9253	0.9425	0.9502
0.5	0.8591	0.8985	0.9225	0.9406	0.9487

Appendix B. Result Tables

Table A2. Macro F1 score and improvement rate for the IMDb dataset.

BERT	0%	10%	20%	30%	40%
Mean MC	0.9354	0.9668 (1.0335)	0.9829 (1.0508)	0.9901 (1.0585)	0.9930 (1.0616)
DE	0.9354	0.9679 (1.0347)	0.9789 (1.0465)	0.9787 (1.0463)	0.9798 (1.0475)
Softmax	0.9364	0.9691 (1.0349)	0.9847 (1.0516)	0.9913 (1.0586)	0.9940 (1.0615)
PL-Variance	0.9364	0.9678 (1.0335)	0.9837 (1.0506)	0.9901 (1.0574)	0.9933 (1.0608)
GloVe					
Mean MC	0.8825	0.9170 (1.0391)	0.9416 (1.0670)	0.9614 (1.0894)	0.9730 (1.1025)
DE	0.8825	0.9183 (1.0406)	0.9430 (1.0686)	0.9449 (1.0707)	0.9455 (1.0714)
Softmax	0.8824	0.9154 (1.0374)	0.9406 (1.0660)	0.9598 (1.0878)	0.9724 (1.1020)
PL-Variance	0.8824	0.9162 (1.0383)	0.9415 (1.0670)	0.9611 (1.0892)	0.9736 (1.1034)

Table A3. Accuracy score and improvement rate for the Amazon (Sports and Outdoors) dataset.

BERT	0%	10%	20%	30%	40%
Mean MC	0.7466	0.7853 (1.0518)	0.8137 (1.0898)	0.8392 (1.1240)	0.8605 (1.1526)
DE	0.7466	0.7850 (1.0513)	0.8191 (1.0871)	0.8492 (1.1374)	0.8684 (1.1631)
Softmax	0.7474	0.7875 (1.0537)	0.8225 (1.1005)	0.8562 (1.1456)	0.8845 (1.1834)
PL-Variance	0.7474	0.7856 (1.0510)	0.8144 (1.0896)	0.8404 (1.1244)	0.8610 (1.1520)
GloVe					
Mean MC	0.6979	0.7369 (1.0559)	0.7675 (1.0998)	0.7962 (1.1408)	0.8214 (1.1770)
DE	0.6979	0.7366 (1.0555)	0.7716 (1.1056)	0.8019 (1.1490)	0.8102 (1.1610)
Softmax	0.6984	0.7374 (1.0559)	0.7730 (1.1068)	0.8067 (1.1550)	0.8359 (1.1969)
PL-Variance	0.6984	0.7358 (1.0536)	0.7676 (1.0990)	0.7961 (1.1398)	0.8209 (1.1753)

Table A4. Macro F1 score and improvement rate for the Wiki dataset.

BERT	0%	10%	20%	30%	40%
Mean MC	0.9227	0.9569 (1.0370)	0.9742 (1.0557)	0.9824 (1.0646)	0.9878 (1.0705)
DE	0.9227	0.9566 (1.0367)	0.9743 (1.0559)	0.9767 (1.0585)	0.9762 (1.0579)
Softmax	0.9230	0.9561 (1.0358)	0.9745 (1.0558)	0.9834 (1.0655)	0.9869 (1.0692)
PL-Variance	0.9230	0.9566 (1.0364)	0.9748 (1.0561)	0.9827 (1.0647)	0.9869 (1.0693)
GloVe					
Mean MC	0.8559	0.8958 (1.0466)	0.9168 (1.0712)	0.9325 (1.0896)	0.9379 (1.0958)
DE	0.8559	0.8914 (1.0415)	0.9146 (1.0686)	0.9269 (1.0830)	0.9319 (1.0889)
Softmax	0.8539	0.8941 (1.0471)	0.9181 (1.0752)	0.9312 (1.0906)	0.9393 (1.1001)
PL-Variance	0.8539	0.8958 (1.0491)	0.9209 (1.0785)	0.9322 (1.0918)	0.9366 (1.0969)

Table A5. Macro F1 score and improvement rate for the SST-2 dataset.

BERT	0%	10%	20%	30%	40%
Mean MC	0.7407	0.7706 (1.0403)	0.7907 (1.0674)	0.8149 (1.1001)	0.8432 (1.1383)
DE	0.7407	0.7744 (1.0454)	0.8008 (1.0811)	0.8265 (1.1158)	0.8472 (1.1437)
Softmax	0.7442	0.7706 (1.0354)	0.8006 (1.0758)	0.8246 (1.1080)	0.8451 (1.1355)
PL-Variance	0.7442	0.7719 (1.0372)	0.7964 (1.0701)	0.8100 (1.0884)	0.8339 (1.1205)
GloVe					
Mean MC	0.7397	0.7658 (1.0354)	0.7853 (1.0354)	0.8013 (1.0833)	0.8202 (1.1088)
DE	0.7397	0.7648 (1.0339)	0.7940 (1.0735)	0.7998 (1.0812)	0.8204 (1.1091)
Softmax	0.7442	0.7686 (1.0328)	0.7918 (1.0639)	0.8023 (1.0780)	0.8217 (1.0141)
PL-Variance	0.7442	0.7686 (1.0328)	0.7918 (1.0639)	0.8023 (1.0780)	0.8204 (1.1023)

Appendix C. Model Calibration Plots

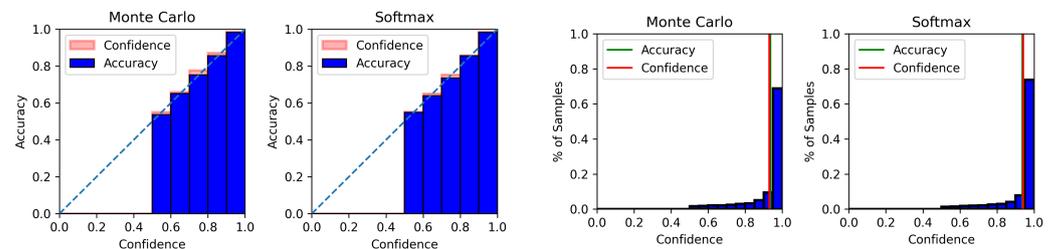


Figure A1. Reliability diagram (left) and confidence histogram (right) of IMDb using BERT-CNN.

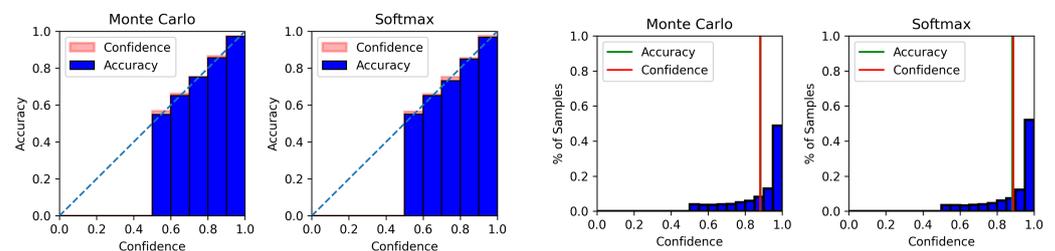


Figure A2. Reliability diagram (left) and confidence histogram (right) of IMDb using GloVe-CNN.

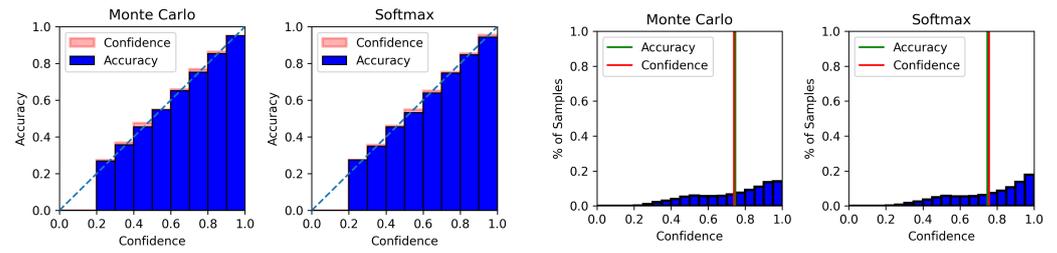


Figure A3. Reliability diagram (left) and confidence histogram (right) of Amazon using BERT-CNN.

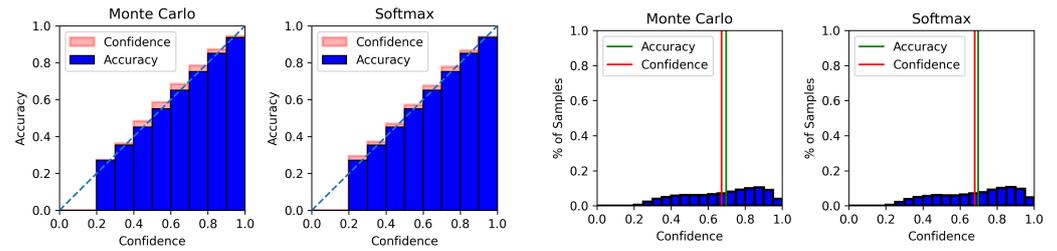


Figure A4. Reliability diagram (left) and confidence histogram (right) of Amazon using GloVe-CNN.

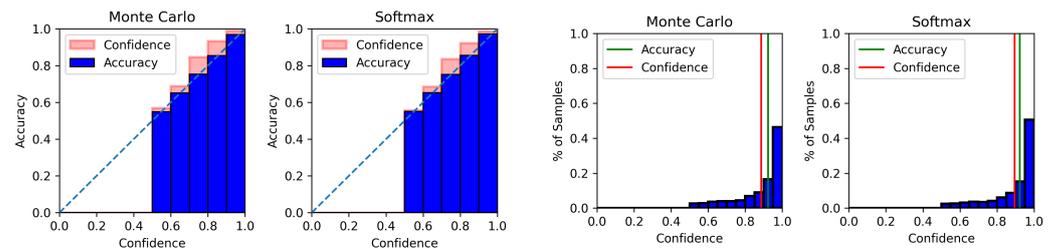


Figure A5. Reliability diagram (left) and confidence histogram (right) of WIKI using BERT-CNN.

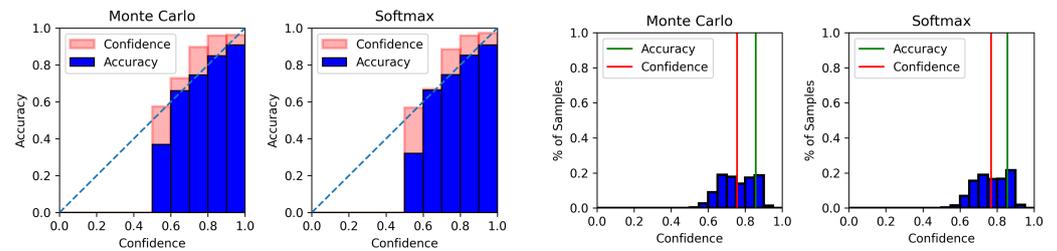


Figure A6. Reliability diagram (left) and confidence histogram (right) of WIKI using GloVe-CNN.

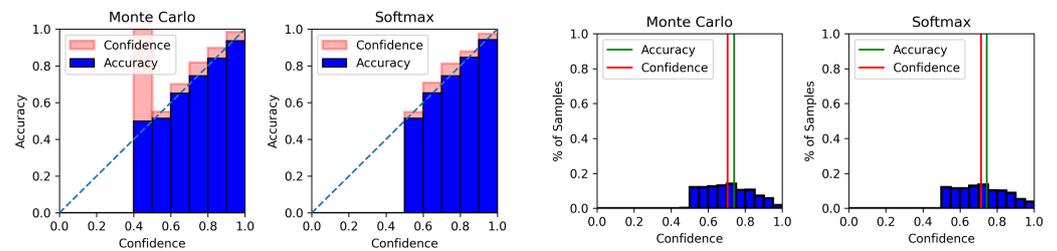


Figure A7. Reliability diagram (left) and confidence histogram (right) of SST-2 using BERT-CNN.

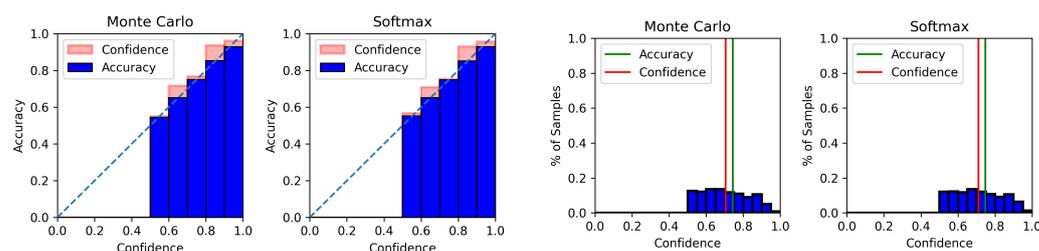


Figure A8. Reliability diagram (left) and confidence histogram (right) of SST-2 using GloVe-CNN.

References

- Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All you Need. In Proceedings of the Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; pp. 5998–6008.
- Lin, T.; Wang, Y.; Liu, X.; Qiu, X. A Survey of Transformers. *arXiv* **2021**, arXiv:2106.04554.
- Tay, Y.; Dehghani, M.; Gupta, J.P.; Aribandi, V.; Bahri, D.; Qin, Z.; Metzler, D. Are Pretrained Convolutions Better than Pretrained Transformers? In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Online, 1–6 August 2021.
- Gawlikowski, J.; Tassi, C.R.N.; Ali, M.; Lee, J.; Humt, M.; Feng, J.; Kruspe, A.; Triebel, R.; Jung, P.; Roscher, R.; et al. A Survey of Uncertainty in Deep Neural Networks. *arXiv* **2021**, arXiv:2107.03342.
- Gal, Y.; Ghahramani, Z. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1050–1059.
- Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; Wierstra, D. Weight Uncertainty in Neural Network. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1613–1622.
- Hendrycks, D.; Gimpel, K. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017.
- Zhang, X.; Chen, F.; Lu, C.T.; Ramakrishnan, N. Mitigating Uncertainty in Document Classification. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 July 2019; Long and Short Papers; Association for Computational Linguistics: Minneapolis, MN, USA, 2019; Volume 1, pp. 3126–3136. [[CrossRef](#)]
- He, J.; Zhang, X.; Lei, S.; Chen, Z.; Chen, F.; Alhamadani, A.; Xiao, B.; Lu, C. Towards More Accurate Uncertainty Estimation In Text Classification. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP); Association for Computational Linguistics, Online, 16–20 November 2020; pp. 8362–8372. [[CrossRef](#)]
- Strubell, E.; Ganesh, A.; McCallum, A. Energy and Policy Considerations for Deep Learning in NLP. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019.
- Patterson, D.A.; Gonzalez, J.; Le, Q.V.; Liang, C.; Munguia, L.; Rothchild, D.; So, D.R.; Texier, M.; Dean, J. Carbon Emissions and Large Neural Network Training. *arXiv* **2021**, arXiv:2104.10350.
- Ovadia, Y.; Fertig, E.; Ren, J.; Nado, Z.; Sculley, D.; Nowozin, S.; Dillon, J.; Lakshminarayanan, B.; Snoek, J. Can you trust your model’s uncertainty? Evaluating predictive uncertainty under dataset shift. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Curran Associates, Inc.: Red Hook, NY, USA, 2019; Volume 32.
- Henne, M.; Schwaiger, A.; Roscher, K.; Weiss, G. Benchmarking Uncertainty Estimation Methods for Deep Learning with Safety-Related Metrics. In Proceedings of the Workshop on Artificial Intelligence Safety, Co-Located with 34th AAAI Conference on Artificial Intelligence, SafeAI@AAAI 2020, New York, NY, USA, 7 February 2020.
- Mozejko, M.; Susik, M.; Karczewski, R. Inhibited Softmax for Uncertainty Estimation in Neural Networks. *arXiv* **2018**, arXiv:1810.01861.
- van Amersfoort, J.; Smith, L.; Teh, Y.W.; Gal, Y. Uncertainty Estimation Using a Single Deep Deterministic Neural Network. In Proceedings of the 37th International Conference on Machine Learning, ICML Virtual Event, 13–18 July 2020; Volume 119, pp. 9690–9700. PMLR.
- Hinton, G.E.; van Camp, D. Keeping the neural networks simple by minimizing the description length of the weights. In Proceedings of the Sixth Annual Conference on Computational Learning Theory, COLT’93, Santa Cruz, CA, USA, 26–28 July 1993; Association for Computing Machinery: New York, NY, USA, 1993; pp. 5–13. [[CrossRef](#)]
- Neal, R. *Bayesian Training of Backpropagation Networks by the Hybrid Monte Carlo Method*; Technical Report; University of Toronto: Toronto, ON, Canada, 1993.
- MacKay, D.J.C. A Practical Bayesian Framework for Backpropagation Networks. *Neural Comput.* **1992**, *4*, 448–472. [[CrossRef](#)]

19. Lakshminarayanan, B.; Pritzel, A.; Blundell, C. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
20. Durasov, N.; Bagautdinov, T.; Baque, P.; Fua, P. Masksembles for uncertainty estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13539–13548.
21. Niculescu-Mizil, A.; Caruana, R. Predicting good probabilities with supervised learning. In Proceedings of the 22nd international conference on Machine learning, ICML'05, Bonn German, 7–11 August 2005; Association for Computing Machinery: New York, NY, USA, 2015; pp. 625–632. [[CrossRef](#)]
22. Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K.Q. On Calibration of Modern Neural Networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1321–1330.
23. Naeini, M.P.; Cooper, G.F.; Hauskrecht, M. Obtaining Well Calibrated Probabilities Using Bayesian Binning. In Proceedings of the AAAI conference on artificial intelligence, Austin, TX, USA, 25–30 January 2015; pp. 2901–2907.
24. Gal, Y.; Ghahramani, Z. Dropout as a Bayesian Approximation: Appendix. *arXiv* **2016**, arXiv:1506.02157.
25. Joo, T.; Chung, U.; Seo, M.G. Being Bayesian about Categorical Probability. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 4950–4961.
26. Lang, K. NewsWeeder: Learning to Filter Netnews. In Proceedings of the Twelfth International Conference on Machine Learning, Tahoe City, CA, USA, 9–12 July 1995; Prieditis, A., Russell, S., Eds.; Morgan Kaufmann: San Francisco, CA, USA, 1995; pp. 331–339. [[CrossRef](#)]
27. McAuley, J.; Leskovec, J. Hidden factors and hidden topics: Understanding rating dimensions with review text. In Proceedings of the 7th ACM conference on Recommender Systems, RecSys'13, Hong Kong, China, 12–16 October 2013; Association for Computing Machinery: New York, NY, USA, 2013; pp. 165–172. [[CrossRef](#)]
28. Maas, A.L.; Daly, R.E.; Pham, P.T.; Huang, D.; Ng, A.Y.; Potts, C. Learning Word Vectors for Sentiment Analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; Association for Computational Linguistics: Portland, OR, USA, 2011; pp. 142–150.
29. Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.D.; Ng, A.Y.; Potts, C. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, Grand Hyatt Seattle, Seattle, DC, USA, 18–21 October 2013.
30. Redi, M.; Fetahu, B.; Morgan, J.T.; Taraborelli, D. Citation Needed: A Taxonomy and Algorithmic Assessment of Wikipedia's Verifiability. In Proceedings of the WWW'19: The Web Conference, San Francisco, CA, USA, 13–17 May 2019.
31. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, Doha, Qatar, 25–29 October 2014.
32. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2019**, arXiv:1810.04805.
33. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings, San Diego, CA, USA, 7–9 May 2015.
34. Zaragoza, H.; d'Alché Buc, F. Confidence Measures for Neural Network Classifiers. In Proceedings of the 7th Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Paris, France, 6–10 July 1998.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.