Revisiting Softmax for Uncertainty Approximation in Text Classification

Anonymous ACL submission

Abstract

Uncertainty approximation in text classification is an important area with applications in domain adaptation and interpretability. The most widely used uncertainty approximation method is Monte Carlo Dropout, which is computationally expensive as it requires multiple forward passes through the model. A cheaper alternative is to simply use a softmax to estimate model uncertainty. However, prior work has indicated that the softmax can generate overconfident uncertainty estimates and can thus be tricked into producing incorrect predictions. In this paper, we perform a thorough empirical analysis of both methods on three datasets with two base neural architectures in order to reveal insight into the trade-offs between the two. We compare the methods' uncertainty approximations and downstream text classification performance, while weighing their performance against their computational complexity as a cost-benefit analysis. We find that, while Monte Carlo produces the best uncertainty approximations, using a simple softmax leads to competitive F1 results 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.

1 Introduction

The pursuit of pushing state-of-the-art performance on machine learning benchmark datasets often comes with an added cost of computational complexity. On top of already complex base models, such as Transformer models (Vaswani et al., 2017; Lin et al., 2021), successful methods often employ additional techniques such as ensembling and uncertainty estimation in order to push performance. 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 (Tay et al., 2021). Here, we turn our attention to predictive uncertainty estimation methods in text classification, which have applications in domain adaptation and can help make models more transparent and explainable, with a focus on Monte Carlo Dropout.

Quantifying predictive uncertainty has been explored using various techniques (Gawlikowski et al., 2021), with the methods being divided into three main categories: Bayesian methods, single deterministic networks, and ensemble methods. Bayesian methods include Monte Carlo (MC) dropout (Gal and Ghahramani, 2016b) and Bayes by back-prop (Blundell et al., 2015). 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.

In this paper, we investigate the cost vs. benefit of choosing simple vs. expensive uncertainty approximation methods for text classification, with the goal of highlighting when and if more complex uncertainty methods should be employed by NLP researchers and practitioners who could benefit from their use. We focus on single deterministic and Bayesian methods. For the single deterministic methods, 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 posits that the softmax output is not the most dependable uncertainty approximation method (Gal and Ghahramani, 2016b; Hendrycks and Gimpel), and as such it has been superseded by newer methods such as MC dropout. 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 (Zhang

et al., 2019; He et al., 2020).

To understand the cost vs. benefit of softmax vs. MC dropout, we perform experiments on three 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 being competitive in performance to MC dropout. As such, given the already high computational cost of deep neural network based methods and recent pushes for more green ML (Strubell et al., 2019; Patterson et al., 2021), we recommend not discarding simple uncertainty approximation methods such as softmax just yet, as they are often surprisingly effective, while being more efficient.

Contribution In summary, our contributions are as follows: 1) An empirical study of MC dropout and softmax for text classification tasks, using two different neural architectures and three datasets; 2) An analysis of the underlying performance of MC dropout and softmax using expected calibration error; 3) A comparison of MC dropout and softmax, using the two methods' measured accuracy/F1 and runtime.

2 Related Work

2.1 Uncertainty Quantification

Quantifying the uncertainty of a prediction can be done using various techniques (Ovadia et al., 2019; Gawlikowski et al., 2021; Henne et al., 2020) such as single deterministic methods (Możejko et al., 2019; van Amersfoort et al., 2020) 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 (Hinton and van Camp, 1993), Sampling (Neal, 1993), and Laplace Approximation (MacKay, 1992). Some of the more notable methods are Bayes by backprop (Blundell et al., 2015) and Monte Carlo Dropout (Gal and Ghahramani, 2016b). 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 (Lakshminarayanan et al., 2017; He et al., 2020; Durasov et al., 2021). Recently, we have seen uncertainty methods being used to develop methods for new tasks (Zhang et al., 2019; He et al., 2020), 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 done in multiple ways, each offering benefits and downsides. Niculescu-Mizil and Caruana (2015) 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 calibration error (ECE, Guo et al., 2017). While ECE measures the underlying confidence of the uncertainty approximation, we have also seen the use of human intervention for text classification (Zhang et al., 2019; He et al., 2020). There, the uncertainty estimates are used to identify uncertain predictions from the model and ask humans to classify these predictions. The human classified data is 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 (Guo et al., 2017; Naeini et al., 2015). 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. 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.

3.1 Bayesian Neural Networks

Before introducing the MC dropout method, we quickly introduce the concept of *Bayesian Neural Networks* (BNN)s. We start by comparing a BNN to a traditional NN. A traditional NN is of frequentist mindset. Therefore, it assumes that the network weights are real but of an unknown value and can be found through maximum-likelihood estimation, and the input data are treated as random variables. The BNN instead views the weights as random variables and infers a posterior distribution $p(\omega|\mathcal{D})$ over ω after observing $\mathcal{D}$, where $\mathcal{D} = (x, y)$ and ω are the weights of the model. The posterior distribution is defined as follows:

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

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, and 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 (Blundell et al., 2015) and Monte Carlo Dropout (Gal and Ghahramani, 2016b).

3.2 Monte Carlo Dropout

We provide a high-level introduction to the method but refer the reader to the literature for the in-depth theory and proofs (Gal and Ghahramani, 2016b,a). MC dropout approximates the posterior $p(\omega|\mathcal{D})$ by leveraging the dropout layers in a model. In particular, it simply allows the dropout layers to remain active during testing and obtains multiple samples by passing an input through the model multiple times with different nodes dropped on each pass (see Figure 1). Mathematically, we introduce $q(\omega)$, 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:

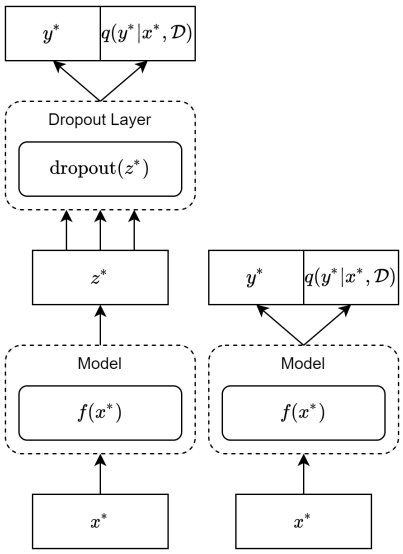

Figure 1: In this figure (left), we show how the MC dropout method functions in a NN and how we can use the representation calculated by the model before the last dropout layer as a reference point, and how we can reuse it to avoid the high costs of recalculating the entire model multiple times. We further show the simplicity of the softmax next to it (right).

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

For the proof of how to get from Eq. (2) to Eq. (3), we refer the reader to (Gal and Ghahramani, 2016b).

As this requires multiple forward passes, this introduces added computational costs. To help alleviate this and provide a fair comparison with the more lightweight softmax, 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.

3.2.1 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:

$$u_i = \frac{1}{K} \sum_{k=1}^{K} \text{Softmax}\left(z_i^k\right), \quad (4)$$

where z_i^k is the logits of the i'th data point of the k'th forward pass. The second method we use to quantify the predictive uncertainty is Dropout Entropy (DE) (Zhang et al., 2019) which uses a combination of binning and entropy:

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

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

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 is one of the most common objectives used in classification tasks for processing the logits of NNs. Softmax is defined as follows:

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

where y_i is the i'th data point and the logits of a NN. The softmax yields a probability distribution over the predicted classes. However, the predicted probability distribution is often overconfident toward the predicted class (Gal and Ghahramani, 2016b; Hendrycks and Gimpel). The issue of softmax's overconfidence can also be exploited (Gal and Ghahramani, 2016b; Joo et al., 2020) – 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 (Guo et al., 2017). In Section 4.6 we measure the model's confidence level and further inspect the distribution of the model's predictions as a part of our model calibration analysis.

4 Experiments and Results

We consider three different datasets and two different 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. 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 three text classification datasets. We use the following three datasets: 20 Newsgroups dataset (Lang, 1995), the Amazon dataset (McAuley and Leskovec, 2013) and the IMDb dataset (Maas et al., 2011). The 20 Newsgroup dataset is a collection of $20,000$ news articles consisting of 20 different classes. We use the 'sports and outdoors' category for the Amazon dataset, which consists of $272,630$ reviews with ratings from 1 to 5. We also use the IMDb dataset, which is a binary classification task consisting of $50,000$ samples. We create the following splits for the datasets: 60%, 20% and 20% for training, validation and testing, respectively, with each set having been selected randomly.

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 (Zhang et al., 2019), which is a CNN using pre-trained GloVe embeddings with a dimension of 200 (Pennington et al., 2014). The second model uses a pre-trained BERT model (Devlin et al., 2019) fine-tuned as masked language model on the dataset under evaluation to obtain contextualized embeddings, which are then input to a CNN with 4 layers. For both models we use the final dropout layer for MC dropout. Both models are optimised using Adam (Kingma and Ba, 2015) 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 of the MC dropout method with

	0%	10%	20%	30%	40%
1	0.8212	0.8721	0.8997	0.9167	0.9367
10	0.8623	0.9008	0.9228	0.9416	0.9495
25	0.8540	0.8978	0.9223	0.9405	0.9507
50	0.8591	0.8985	0.9225	0.9406	0.9487
100	0.8559	0.8966	0.9238	0.9385	0.9485
1000	0.8573	0.9007	0.9253	0.9406	0.9492

Table 1: This table shows how the number of samples affect the performance of the MC dropout method, using the CNN model with BERT embeddings. The results are reported using macro F1.

the CNN model using BERT embeddings 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.

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 done on the test sets. The training of the model is independent of the two methods, since we only use the methods 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 Uncertainty Estimation Performance

We use two main methods to measure uncertainty estimation performance: test data holdout and expected calibration error (ECE). For base model performance, we record the macro F1 score on the 20 Newsgroups and IMDb datasets, and the accuracy on the Amazon dataset.

Test data holdout We provide 4 scores for each method, where each result is based on holding out a proportion of the test data. The uncertainty method identifies the most uncertain predictions, which are then left out of the F1 and accuracy calculations. The holdout is done at 10%, 20%, 30% and 40% of the data. 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 (Zhang et al., 2019).

Expected calibration error To measure model calibration, we use the expected calibration error (ECE) (Guo et al., 2017), which measures the difference between the predictive uncertainties and the labels. 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 range of predictive uncertainty. ECE is defined as:

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

where M is the size of the dataset and acc and $conf$ is the accuracy and mean confidence of the bin B_m.

To observe the difference between the MC dropout and softmax, we create both confidence histograms and reliability diagrams (Guo et al., 2017). 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. The reliability diagrams help us visualise the ECE, by showing the accuracy and mean confidence of each bin, where each bin contains 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.

	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
	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

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.

4.5 Test Data Holdout Results

Table 3, Table 4 and Table 5 show the performance of the two uncertainty approximation methods using the different datasets and embeddings. The tables show the macro F1 score or the accuracy depending on the datasets, and the improvement ratio in the parenthesis. 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 not far from each other 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 beneficial on larger datasets, where the DE calculations are too large to compute.

4.6 Model Calibration Results

To further investigate the differences between MC dropout and softmax, we utilize the expected calibration error (ECE) to observe the differences in the predictive uncertainties. In Table 6, we show the accuracy and ECE on the three datasets using the BERT embeddings.

The results from our holdout experiments in Table 3, 4 and 5 combined with the results from our ECE calculations in Table 6, all point in the direction of MC dropout and softmax performing equally to some extent, with the main difference of the two methods being efficiency as shown in Table 2. To get 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. In Figure 2 and 3 we show the reliability diagrams and the confidence histograms on the 20 Newsgroups dataset using our CNN with both the MC dropout method and softmax utilising BERT and GloVe. We create the reliability diagrams using 10 bins and the confidence histograms with 20. Where 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. Using the reliability diagram, we observe that the difference in confidence and outputs are small. The difference between the two methods is also minimal, including both BERT and GloVe embeddings, suggesting minimal potential gains from using MC dropout. We determine that there is minimal difference by visual inspecting the plots, and by observing the ECE displayed in Table 6. We further observe that in both MC dropout and softmax that the model worsens when we use the GloVe embeddings. As mentioned earlier, we know that the softmax method 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, utilizes the probability space to a greater extent. We include reliability diagrams and confidence histograms for the 2 other datasets in Appendix B.

Inspecting both Table 6 showing the ECE values and the performances in Table 3, 4 and 5, we observed that using our two methods, MC dropout and softmax, we achieved very high F1 scores and accuracies and low ECEs. We hypothesized that high performance could lead to softmax achieving high ECE, due to naturally having high confidence, compared to MC dropout. We added noise to the 20 Newsgroups test embeddings and redid 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

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 (0.1104)	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)

Table 3: Macro F1 score and improvement rate for the 20 Newsgroups 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 4: Macro F1 score and improvement rate for the IMDb 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 5: Accuracy score and improvement rate for the Amazon (Sports and Outdoors) dataset.

	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

Table 6: Accuracy and ECE of the two uncertainty approximation approaches on the three selected datasets.

of a model is low, MC dropout could potentially obtain more precise predictive uncertainties.

5 Discussion and Conclusion

In this paper, we perform an in-depth empirical comparison of using the MC dropout method 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 hold-out experiments, where we select a percentage of the dataset to exclude from the test, that the two methods perform equally or that softmax slightly outperforms MC dropout in some cases. Looking at the ECE experiments, the results again show that the MC dropout and softmax method perform somewhat equally, which we have shown in Section

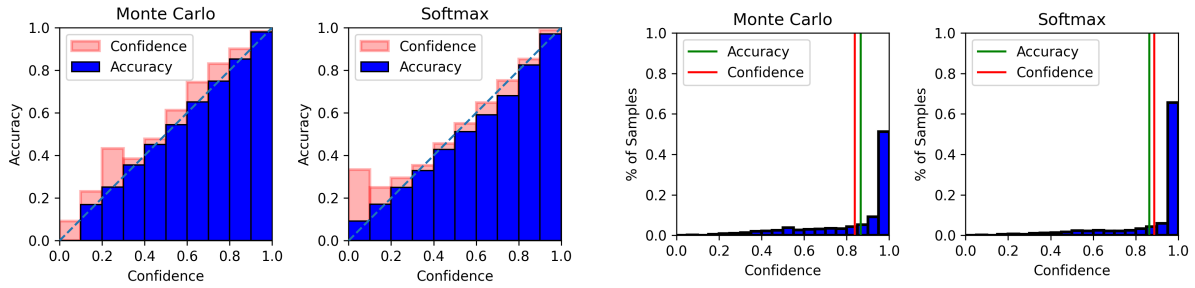

Figure 2: Reliability diagram (left) and confidence histogram (right) of 20 Newsgroups using BERT embeddings. In the reliability diagram, we can observe the difference between the confidence and the output of the model, and the better model has a low gap between confidence and output, which shows the model is better calibrated.

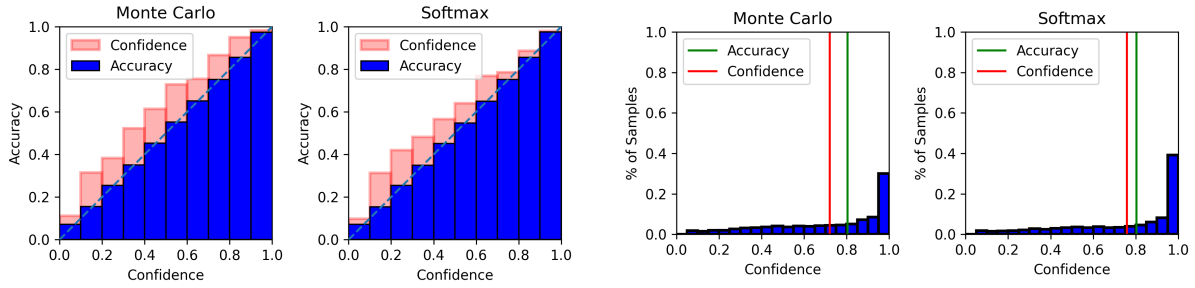

Figure 3: Reliability diagram (left) and confidence histogram (right) of 20 Newsgroups using GloVe embeddings. Comparing the plots of the figure to Figure 2, we see slight differences in both the reliability diagram and the confidence histogram. Most noticeable, we see slight differences in the reliability diagram, where we see more significant gaps between the confidence and the outputs, which indicates a less calibrated model due to the GloVe embeddings.

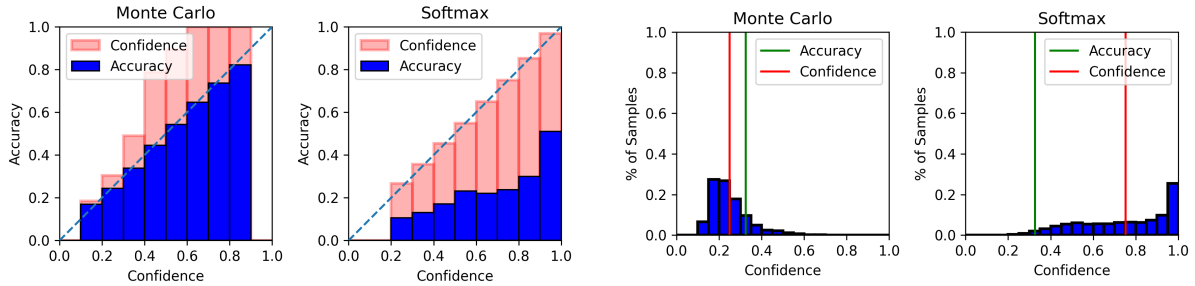

Figure 4: Reliability diagram of 20 Newsgroups dataset using BERT embeddings, with added noise to the BERT embeddings. By adding noise to the test embeddings, we observe how the MC dropout keeps the mean confidence grounded in the confidence histogram, compared to the softmax, which keeps being confident, which can be observed by the distance between the accuracy and mean confidence in the confidence histogram.

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. While the two methods perform equally, the cost of using MC dropout is at a minimum 10 times that of running softmax, 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. Our empirical findings suggest that MC dropout suffers from diminishing returns – the better the model performs, the less can be gained by using MC dropout. By testing the hypothesis by introducing noise in the dataset, we showed that the MC dropout did outperform softmax when the accuracy was lower.

To summarize our findings, we observe that the difference between MC dropout and softmax narrows as the accuracy score increases. Therefore, using MC dropout on a high accuracy dataset provides close to no benefit and using softmax would instead provide a good predictive uncertainty while also being computationally efficient.

References

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight Uncertainty in Neural Network. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1613–1622. PMLR. ISSN: 1938-7228.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Nikita Durasov, Timur Bagautdinov, Pierre Baque, and Pascal Fua. 2021. Masksembles for uncertainty estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13539–13548.

Yarin Gal and Zoubin Ghahramani. 2016a. Dropout as a Bayesian Approximation: Appendix. *arXiv:1506.02157 [stat]*. ArXiv: 1506.02157.

Yarin Gal and Zoubin Ghahramani. 2016b. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1050–1059. PMLR. ISSN: 1938-7228.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, Muhammad Shahzad, Wen Yang, Richard Bamler, and Xiao Xiang Zhu. 2021. A Survey of Uncertainty in Deep Neural Networks. *arXiv:2107.03342 [cs, stat]*. ArXiv: 2107.03342.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1321–1330. PMLR. ISSN: 2640-3498.

Jianfeng He, Xuchao Zhang, Shuo Lei, Zhiqian Chen, Fanglan Chen, Abdulaziz Alhamadani, Bei Xiao, and ChangTien Lu. 2020. Towards more accurate uncertainty estimation in text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8362–8372, Online. Association for Computational Linguistics.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks.

Maximilian Henne, Adrian Schwaiger, Karsten Roscher, and Gereon Weiss. 2020. 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 City, NY, USA, February 7, 2020*, volume 2560 of *CEUR Workshop Proceedings*, pages 83–90. CEUR-WS.org.

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*, COLT '93, pages 5–13, New York, NY, USA. Association for Computing Machinery.

Taejong Joo, Uijung Chung, and Min-Gwan Seo. 2020. Being Bayesian about Categorical Probability. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4950–4961. PMLR. ISSN: 2640-3498.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ken Lang. 1995. NewsWeeder: Learning to Filter Netnews. In Armand Prieditis and Stuart Russell, editors, *Machine Learning Proceedings 1995*, pages 331–339. Morgan Kaufmann, San Francisco (CA).

Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. 2021. A survey of transformers. Cite arxiv:2106.04554.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

David J. C. MacKay. 1992. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, RecSys '13, pages 165–172, New York, NY, USA. Association for Computing Machinery.

Marcin Możejko, Mateusz Susik, and Rafał Karczewski. 2019. Inhibited softmax for uncertainty estimation in neural networks.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. 2015. Obtaining well calibrated probabilities using bayesian binning. 2015:2901–2907.

Radford Neal. 1993. Bayesian Training of Backpropagation Networks by the Hybrid Monte Carlo Method. Technical report.

Alexandru Niculescu-Mizil and Rich Caruana. 2015. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 625–632. Association for Computing Machinery.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. Can you trust your model' s uncertainty? Evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon emissions and large neural network training.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Yi Tay, Mostafa Dehghani, Jai Prakash Gupta, Vamsi Aribandi, Dara Bahri, Zhen Qin, and Donald Metzler. 2021. 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)*, pages 4349–4359, Online. Association for Computational Linguistics.

Joost van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. 2020. Uncertainty estimation using a single deep deterministic neural network.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, page 5998–6008. Curran Associates, Inc.

Xuchao Zhang, Fanglan Chen, Chang-Tien Lu, and Naren Ramakrishnan. 2019. 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, Volume 1 (Long and Short Papers)*, pages 3126–3136, Minneapolis, Minnesota. Association for Computational Linguistics.

A Reproducibility

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 12GB RAM and 56GB of RAM.

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 and IMDb and 128 for Amazon; early stopping: 10; learning rate: 0.001. For fine-tuning BERT we used the following set of hyperparamaters: 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.

A.3 Dropout - Hyperparameter

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 7, we show the results using the 5 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.

	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

Table 7: We test how the dropout probabilities correlate with the performance of MC dropout, using a CNN model with BERT embeddings. The results are reported in terms of macro F1.

B Model Calibration Plots

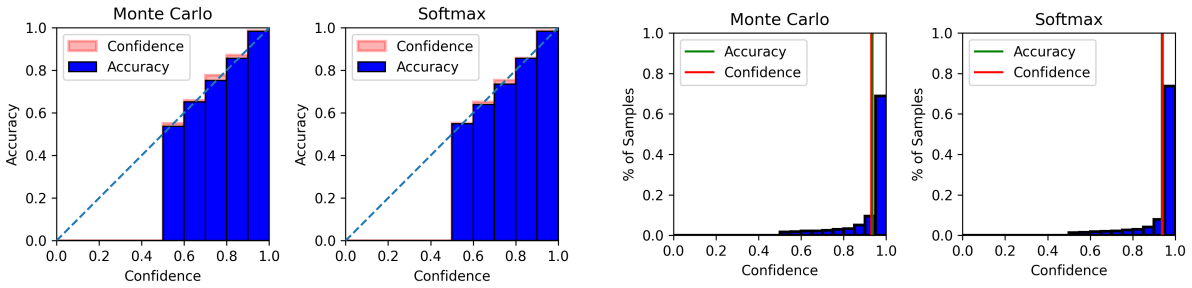

Figure 5: Reliability diagram (left) and confidence histogram (right) of IMDb using BERT embeddings.

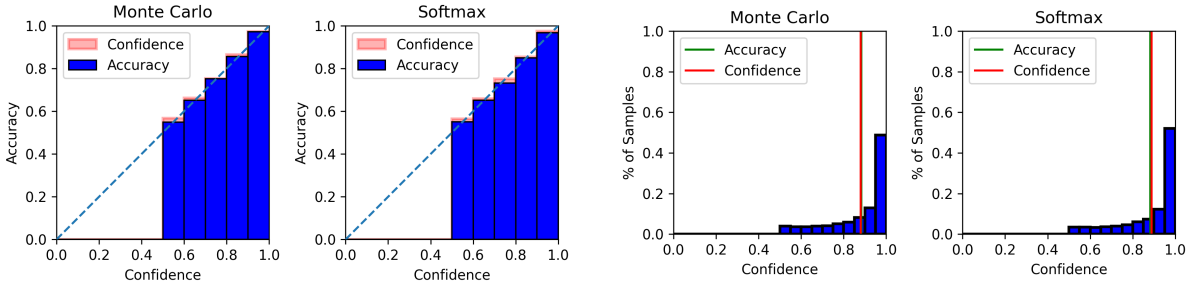

Figure 6: Reliability diagram (left) and confidence histogram (right) of IMDb using GloVe embeddings.

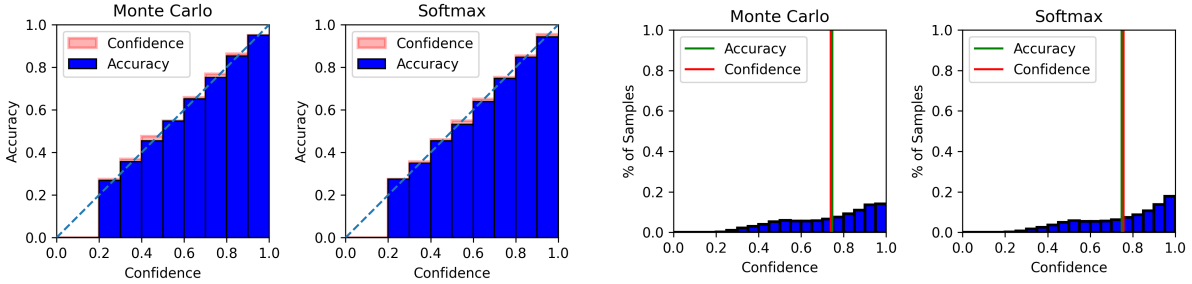

Figure 7: Reliability diagram (left) and confidence histogram (right) of Amazon using GloVe embeddings.

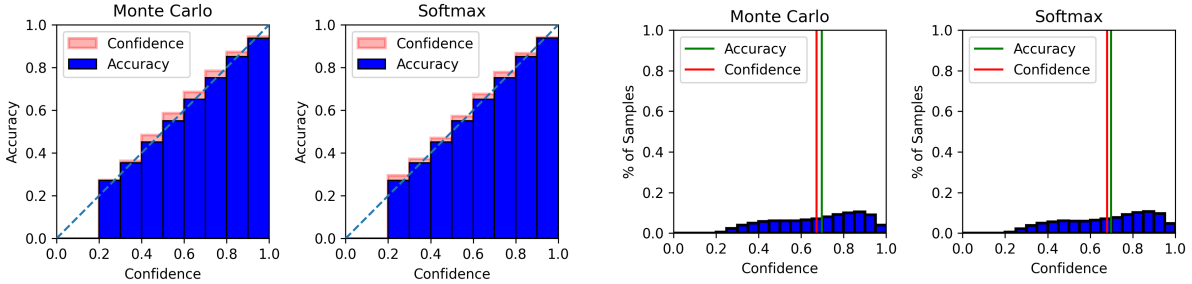

Figure 8: Reliability diagram (left) and confidence histogram (right) of Amazon using GloVe embeddings.