How Certain is Your Transformer?

Artem Shelmanov†, Evgenii Tsymbalov‡, Dmitri Puzyrev‡,*, Kirill Fedyanin†,
Alexander Panchenko‡, and Maxim Panov‡

†Skolkovo Institute of Science and Technology, Moscow, Russia
‡Huawei Industry Video Application Lab, Moscow, Russia
*Mobile TeleSystems (MTS), Moscow, Russia

{a.shelmanov, evgenii.tsymbalov, dmitry.puzyrev, k.fedyanin, a.panchenko, m.panov}@skoltech.ru

Abstract

In this work, we consider the problem of uncertainty estimation for Transformer-based models. We investigate the applicability of uncertainty estimates based on dropout usage at the inference stage (Monte Carlo dropout). The series of experiments on natural language understanding tasks shows that the resulting uncertainty estimates improve the quality of detection of error-prone instances. Special attention is paid to the construction of computationally inexpensive estimates via Monte Carlo dropout and Determinantal Point Processes.

1 Introduction

Quantifying the uncertainty of machine learning models is an important aspect of trustworthy, reliable, and accountable natural language understanding (NLU) systems. Obtaining measures of uncertainty in predictions (also known as uncertainty estimations, UE) helps to detect out-of-domain (Maninlin and Gales, 2018), adversarial, or error-prone instances that require special treatment. For example, such instances can be additionally checked by human experts or another more advanced system or alternatively rejected from classification (Herbei and Wegkamp, 2006). Besides, uncertainty estimation is an essential component of various applications such as active learning (Shelmanov et al., 2021) and outlier/error detection in a dataset (Larson et al., 2019).

Many modern NLU methods take advantage of deep pre-trained models that are based on the Transformer architecture (Vaswani et al., 2017) (e.g., BERT (Devlin et al., 2019) or ELECTRA (Clark et al., 2020)). Obtaining reliable uncertainty estimations for such neural networks (NNs) can, therefore, directly benefit a wide range of NLU tasks, yet implementing UEs, in this case, is challenging due to the huge number of parameters in these deep learning models. The approximations of Bayesian inference based on dropout usage at the inference stage – Monte Carlo (MC) dropout (Gal and Ghahramani, 2016), provide a realizable approach to quantifying UEs of deep models. However, they are usually accompanied by serious computational overhead due to the necessity of performing multiple stochastic predictions. Importantly, training ensembles of independent models (Lakshminarayanan et al., 2017) leads to even more prohibitive overheads.

In this work, we investigate various MC dropout-based approaches to uncertainty quantification of NLU models on the widely-used General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018). The main contributions of our work are two-fold:

• We show that the use of the MC dropout with pre-trained Transformer models significantly improves the quality of UEs in NLU tasks compared to deterministic baselines.

• We are the first to our knowledge to apply a modification of the MC dropout based on determinantal point processes (DPP; Tsymbalov et al. (2020)) to Transformers and show that this approach allows obtaining the UEs competitive to the standard MC dropout at a fraction of its cost. To improve the stability of the DPP-based dropout for Transformer-based models, we extend the method presented in Tsymbalov et al. (2020) by averaging multiple dropout masks sampled with DPP.

2 Related Work

Three dominating approaches to uncertainty estimation in neural networks exist: (i) interpretation of the model’s logits from the uncertainty estimation perspective (Gal, 2016), which is the basic one; (ii) ensembling, where a discrepancy...
between models’ predictions are interpreted as a sample variance (Lakshminarayanan et al., 2017); (iii) Bayesian neural networks (Teye et al., 2018), which have a built-in mechanism to capture uncertainty via a single model.

There are a few recent works that investigate uncertainty quantification for NLP models and use MC dropout techniques. Dong et al. (2018) use Bayesian UEs for the analysis of semantic parser predictions for correctness. Zhang et al. (2019) propose an additional training loss component that facilitates smaller inter-class and bigger intra-class distances in the vector space of the output layer. Experiments with convolutional NNs on text classification datasets show that this modification helps to improve error detection using MC dropout UEs. For quantifying model data uncertainty, Xiao and Wang (2019) use NNs to parameterize a probability distribution (mean and variance) instead of making a prediction directly. For quantifying model uncertainty, the authors leverage the MC dropout. Modeling both types of uncertainties in convolutional and recurrent NNs helped them to improve the performance in regression and classification NLP tasks. Kochkina and Liakata (2020) apply UEs to the problem of rumor verification.

3 Uncertainty Estimation of Deep Transformer Neural Networks

In this section, we describe types of dropout, uncertainty estimation methods, and the Transformer-based neural classifier used in our experiments.

3.1 Types of Dropout

We use two types of dropout described below.

Monte Carlo Dropout The dropout (Srivastava et al., 2014) has emerged as a powerful and universal regularization technique applicable to most DL architectures, with the Transformers not being an exception. Despite originally being an empirical, engineering way to fight the overfitting, it then obtained a theoretical explanation as a special case of Bayesian NNs, where activations are drawn from the Bernoulli distribution (Gal, 2016). This allows to represent a vector of outputs $x_h$ of the $h$-th layer of the network as a function of its weights $W_h$, activation function $\sigma$, and a dropout mask $M_h$:

$$x_h = \sigma(x_{h-1} \mid W_h, M_h), M_h \sim Bernoulli(1-p),$$

where $p \in [0; 1]$ is the dropout rate.

This theoretical explanation enables the use of the dropout not only at the training stage but also at the inference stage via sampling of multiple masks $M^{(t)}_h, t = 1, \ldots, T$ for each dropout layer of the network $h$ and subsequently providing an ensemble of models parameterized by these masks: $f_r(x) = f(x \mid \{M^{(t)}_h\})$. The obtained UEs are relatively fast, convenient, and applicable to various tasks, such as regression (Tsybalkov et al., 2018), image classification (Gal and Ghahramani, 2016), and active learning (Gal et al., 2017; Siddhant and Lipton, 2018).

Monte Carlo Dropout with Determinantal Point Processes The models obtained from the standard dropout masks usually show a high degree of correlation in predictions between them, limiting the power of the resulting ensemble. Recently, it was proposed to improve the diversity of predictions by considering the correlations between neurons and sampling the diverse neurons via the mechanism of Determinantal Point Processes (DPP; Kulesza and Taskar (2012)), an approach for sampling diverse elements from a set of points. This setup was proposed by (Tsybalkov et al., 2020) and evaluated for the simple multilayer perceptrons and CNNs. In this work, we aim to extend this approach to Transformer models.

DPP-based dropout masks $M^{DPP}_h$ for the $h$-th layer are constructed using the correlation matrix $C_h$ between neurons as a likelihood kernel for the DPP: $M^{DPP}_h \sim DPP(C_h)$. The probability to select a set $S$ of activations on the layer $h$ is given by

$$P[M^{DPP}_h = S] = \frac{\det C^S_h}{\det(C_h + I)},$$

where $C^S_h$ is a square submatrix of $C_h$ obtained by keeping only rows and columns indexed by the sample $S$. The matrix of correlations between activations of the $h$-th layer $C_h$ is estimated empirically based on some set of points, which represents the data distribution well enough (i.e. training set). The key feature of the approach is that DPP tends to sample neurons with low correlations between them, which in turn improves the overall diversity of the obtained models. More information about DPP is presented in Appendix B.

To improve the stability of the DPP-based dropout for Transformer-based models, we create a final dropout mask by sampling from DPP and averaging multiple initial masks.
3.2 Uncertainty Estimates

Let $T$ be a number of stochastic passes, i.e., the number of dropout masks to be sampled. We use the three following UEs (also known as acquisition methods) for the classification with $C$ classes:

- **Sampled maximum probability**:\[ 1 - \max_c \bar{p}_T(y = c \mid x), \]
  where $\bar{p}_T$ is an average probability for the class $c$ prediction over multiple stochastic passes $t = 1, \ldots, T$.

- **Probability variance** averaged over classes:\[ \frac{1}{T} \sum_{t=1}^{T} \left( \frac{1}{C} \sum_{c=1}^{C} p_t(y = c \mid x) - \bar{p}_T(y = c \mid x) \right)^2. \]

- **Bayesian Active Learning by Disagreement (BALD)** proposed by Houlsby et al. (2011) describes the mutual information between outputs and model parameters:
  \[ H(x) + \frac{1}{T} \sum_{t=1}^{T} \sum_{c=1}^{C} p(y = c \mid x) \log(p(y = c \mid x)), \]
  where $H(x)$ is the entropy of the ensemble mean.

We would like to note that all these estimates can be used for any ensembling technique, including the MC dropout and the DPP-based dropout.

3.3 Classification Models

In this work, we focus on the ELECTRA (Clark et al., 2020) model, which is a recent successor to BERT (Devlin et al., 2019). It is based on the same Transformer architecture but takes advantage of the harder “replaced token detection” objective instead of the “masked language model” objective. This gives better pre-training capabilities and makes ELECTRA the state-of-the-art Transformer in natural language understanding benchmarks. We should note that ELECTRA is regularized with multiple dropout layers, which facilitates the usage of the MC dropout. For example, the body of the “ELECTRA-base” model has 37 dropout layers.

We also experiment with DistilBERT (Sanh et al., 2019), which is a smaller Transformer obtained from the middle-size BERT (Devlin et al., 2019) via a distillation procedure (Hinton et al., 2015). This model provides the faster inference and has smaller memory requirements but retains 97% of the language understanding capabilities of the original model according to Sanh et al. (2019).

4 Experiments

4.1 Experimental Setup

We evaluate the UEs on the basis of their ability to detect misclassification. High UEs should indicate potential errors in the model output, while low uncertainties should correspond to correctly classified instances. In this vein, we transform the original task into a binary classification task by comparing predictions of a model with the ground truth labels in the validation dataset. Uncertainty estimates on the validation dataset are treated as the outputs of the binary classifier that is trained to look for potential errors. We calculate the ROC AUC score using the new ground truth labels and UEs and use this score as the main evaluation metric.

The baseline in this task is the UE calculated based on the maximal probability of the original deterministic model. We compare it to the estimates obtained using multiple stochastic predictions with activated dropout layers. Three variants of estimates are calculated: 1) based on the model, in which MC dropout is applied to all dropout layers; 2) based on the model with the MC dropout applied only to the last layer; 3) based on the model with the DPP-based sampling applied to the last dropout layer. For calculating these UEs, we conduct 20 stochastic predictions. The dropout rate in these passes for the MC dropout is 0.1, which is shown to be optimal in the preliminary experiments. For the DPP dropout, we sample and average multiple masks produced by DPP. In experiments with SST-2 and ELECTRA, we average as many masks so at least 30% of neurons remain active during the pass (this roughly can be considered as a “dropout rate” of 0.7). For MRPC, we choose the “dropout rate” equal to 0.2 and for CoLA: 0.4. For DistilBERT, we use the “dropout rate” of 0.4 in all tasks.

We train three versions of models with different random seeds. For each model, another five random seeds are used to produce predictions for stochastic methods. Multiple models and predictions are used for estimating the standard deviation and conducting the statistical significance testing.

4.2 Datasets

We evaluate UEs and dropout variants on the widely used NLU benchmark GLUE (Wang et al., 2018). Specifically, we perform experiments on three tasks: Stanford Sentiment Treebank (SST-2; Socher et al. (2013)), Corpus of Linguistic Acceptability (CoLA; Warstadt et al. (2019)), and
Table 1: The misclassification detection performance (ROC AUC) (±SD) for the baseline with the ELECTRA model and performance improvements over the baseline for various UE methods. Statistically significant improvements (p-value ≤ 0.05) are highlighted.

<table>
<thead>
<tr>
<th>Dropout Type</th>
<th>Model Acquisition</th>
<th>Dropout Layers</th>
<th>SST-2</th>
<th>MRPC</th>
<th>CoLA</th>
</tr>
</thead>
<tbody>
<tr>
<td>No (baseline)</td>
<td>Max. probability</td>
<td>-</td>
<td>79.7±3.4</td>
<td>78.6±4.1</td>
<td>78.7±2.0</td>
</tr>
<tr>
<td>MC</td>
<td>Sampled max. prob.</td>
<td>last</td>
<td>-0.1±0.2</td>
<td>-0.5±0.4</td>
<td>-0.1±0.1</td>
</tr>
<tr>
<td>MC</td>
<td>Probability variance</td>
<td>last</td>
<td>-1.9±1.0</td>
<td>-3.0±0.7</td>
<td>-1.4±0.7</td>
</tr>
<tr>
<td>MC</td>
<td>BALD</td>
<td>last</td>
<td>-5.0±1.7</td>
<td>-5.6±1.3</td>
<td>-3.9±1.3</td>
</tr>
<tr>
<td>DPP</td>
<td>Sampled max. prob.</td>
<td>last</td>
<td>3.2±2.4</td>
<td>0.0±0.7</td>
<td>-0.4±0.7</td>
</tr>
<tr>
<td>DPP</td>
<td>Probability variance</td>
<td>last</td>
<td>2.7±3.1</td>
<td>1.5±2.3</td>
<td>-1.1±1.3</td>
</tr>
<tr>
<td>DPP</td>
<td>BALD</td>
<td>last</td>
<td>0.7±2.7</td>
<td>2.1±3.1</td>
<td>-2.0±2.1</td>
</tr>
<tr>
<td>MC</td>
<td>Sampled max. prob.</td>
<td>all</td>
<td>3.2±1.6</td>
<td>5.5±2.6</td>
<td>3.2±0.6</td>
</tr>
<tr>
<td>MC</td>
<td>Probability variance</td>
<td>all</td>
<td>4.7±2.1</td>
<td>7.2±3.1</td>
<td>2.9±0.4</td>
</tr>
<tr>
<td>MC</td>
<td>BALD</td>
<td>all</td>
<td>5.2±2.4</td>
<td>7.5±3.3</td>
<td>2.8±0.4</td>
</tr>
</tbody>
</table>

Microsoft Research Paraphrase Corpus (MRPC; Dolan and Brockett (2005)). The SST-2 task is to predict the sentiment of a given sentence (positive/negative). The SST-2 dataset was randomly subsampled to 2% of the original size to emulate the situation with a small amount of training data. The CoLA task is to determine whether the given sentence is grammatical or not. The MRPC task is to predict whether two given sentences are semantically similar or not. We select these three datasets for their compact size.

4.3 Model and Training Details

We use the middle-size pre-trained ELECTRA-base model with 110 million parameters and the DistilBERT model with 66 million parameters obtained from the middle-size BERT. The implementation of the models is provided by the Huggingface Transformers library (Wolf et al., 2020). For fine-tuning models, we follow the approach described by Clark et al. (2020) and Devlin et al. (2019): train for 4 epochs with 10% warm-up and a linear learning rate scheduler. For all models and tasks, we use the same learning rate equal to 5e-5. For ELECTRA and SST-2 and MRPC tasks, the batch size is 16. For ELECTRA and CoLA, the batch size is 32. For DistilBERT, the batch size is 32 for all tasks. Although calibrating these hyperparameters can yield some performance improvements, the aforementioned settings allow achieving good results across all tasks.

4.4 Results and Discussion

ROC AUC scores for the misclassification detection task and ELECTRA are presented in Table 1. The results for DistilBERT are presented in Table 3 in Appendix A. While the classifier performance does not significantly variate across multiple versions of the fine-tuned models, the difference in the misclassification detection performance is statistically significant. Therefore, we present the absolute values of the performance only for the baseline (UE based on maximum probability), while for other methods, we present the improvement over the baseline across multiple runs. Tables with results also present the standard deviation of scores.

We note that the UE based on the maximum probability of the deterministic model is a strong baseline. Overall, Transformers are able to indicate their potential mistakes with just the probability from the softmax layer. Applying the MC dropout to all dropout layers in the network always gives a reliable boost in the misclassification detection. For SST-2 and MRPC tasks, UE based on BALD demonstrates better performance than sampled maximum probability and variance, while on CoLA, all UEs perform comparably well. The biggest improvement can be achieved for MRPC and ELECTRA: up to 7.5% ROC AUC.

On the contrary, the UEs based on the MC dropout applied only to the last layer do not perform well. We see that the misclassification detection performance always deteriorates compared to the baseline, especially, for variance and BALD. UEs that take advantage of the DPP-based masks applied to the last dropout layer are somewhere in the middle in terms of quality compared to the MC dropout variants. Although this method also does not give any improvement for CoLA, unlike the last layer MC dropout, DPP gives a significant
advantage over the baseline on the SST-2 task for both models and on the MRPC task for ELECTRA. We note that although DPP-based sampling and the last layer MC dropout diverge in terms of “dropout rate” (e.g., in the experiment on the SST-2 task with ELECTRA, 0.7 for the DPP dropout versus 0.1 for the MC dropout), this aspect does not explain the performance difference. Applying dropout rates higher than 0.1 to the MC dropout downgrades the performance of the misclassification detection due to the overall decrease of the model quality, while for DPP, only 30% of neurons is more than enough to retain the model performance and obtain better UEs on the SST-2 task.

Despite the fact that the DPP-based approach appears to be worse than applying the MC dropout on all layers, it is much faster since it is applied to only the last dropout layer. For practical applications, obtaining UEs normally should not cause a significant overhead compared to the standard model inference time. This strikes the methods based on the MC dropout since they require multiple stochastic predictions. However, for most of the pre-trained Transformers, if only the last dropout is replaced with the MC variant, the outputs of the massive Transformer “body” are not affected during the stochastic predictions. This means that the body outputs can be calculated only once, and only the last linear layer with the softmax activation should be recalculated multiple times. As the last layer contains less than 1% of total parameters, this favors the UEs that do not use stochastic inference on dropout layers except the last. Compared to masks generated uniformly with the MC dropout, sampling masks with DPP has some insignificant computation overhead, but, as we showed, it can give a useful contribution to the misclassification performance (for MRPC and SST-2) even if it is used only in the last dropout layer.

We performed an investigation of computation time overhead for calculating UEs with various MC dropout options for the development dataset. The results for ELECTRA are presented in Table 2. The computations were conducted with the Nvidia 2080ti GPU and the Intel Xeon 5217 CPU. We use BALD as an acquisition function, but other functions have comparable execution time. The MC dropout placed on all layers of Transformers gives better improvements, but it causes roughly 2,000% overhead (in the case of 20 stochastic passes), with less than 10% overhead for the last layer MC and DPP. Therefore, DPP can provide a better trade-off between computation time and performance of error detection.

<table>
<thead>
<tr>
<th>UE Method</th>
<th>Inference time, sec.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SST-2</td>
</tr>
<tr>
<td>Deterministic, −</td>
<td>3.07 ± 0.03</td>
</tr>
<tr>
<td>MC dropout, all</td>
<td>65.5 ± 0.7</td>
</tr>
<tr>
<td>MC dropout, last</td>
<td>3.17 ± 0.06</td>
</tr>
<tr>
<td>DPP dropout, last</td>
<td>3.33 ± 0.02</td>
</tr>
</tbody>
</table>

Table 2: The inference time of the ELECTRA model on the development dataset with BALD UE.

5 Conclusion

In this work, we evaluated several UEs for the state-of-the-art Transformer model ELECTRA and the speed-oriented DistilBERT model in the text classification tasks. To obtain estimates, we leverage multiple stochastic passes using the MC dropout, and the DPP-based dropout proposed by (Tsybulya et al., 2020). We show that by activating all dropouts in the model for stochastic predictions, one can beat the baseline deterministic uncertainty estimate by the significant margin in the binary misclassification detection task. We also demonstrate that replacing the last dropout layer with the DPP dropout can yield significant improvements over the baseline in some cases, but less than the usage of the MC dropout on all dropout layers. Despite being inferior compared to the latter, the DPP dropout can provide a better trade-off between computation time and performance of error detection, which can be important for practical use cases.

In future work, we are seeking to improve UEs quality obtained using the DPP dropout with the help of calibration (Safavi et al., 2020) and conduct experiments on sequence tagging tasks.

Acknowledgments

We thank the reviewers for their valuable feedback. The development of uncertainty estimation algorithms for Transformer models (Section 3) was supported by the joint MTS-Skoltech lab. The development of a software system for the experimental study of uncertainty estimation methods and its application to NLP tasks (Section 4) was supported by the Russian Science Foundation grant 20-71-10135. The Zhores supercomputer (Zacharov et al., 2019) was used for computations.
References


## A Results for DistilBERT

<table>
<thead>
<tr>
<th>Dropout Type</th>
<th>Model Acquisition</th>
<th>Dropout Layers</th>
<th>SST-2</th>
<th>MRPC</th>
<th>CoLA</th>
</tr>
</thead>
<tbody>
<tr>
<td>No (baseline)</td>
<td>Max. probability</td>
<td>-</td>
<td>76.6±2.6</td>
<td>76.6±0.8</td>
<td>73.0±1.7</td>
</tr>
<tr>
<td>MC</td>
<td>Sampled max. prob.</td>
<td>last</td>
<td>-0.0±0.1</td>
<td>-0.1±0.3</td>
<td>0.0±0.1</td>
</tr>
<tr>
<td>MC</td>
<td>Probability variance</td>
<td>last</td>
<td>-0.5±0.4</td>
<td>-0.6±0.7</td>
<td>-0.2±0.4</td>
</tr>
<tr>
<td>MC</td>
<td>BALD</td>
<td>last</td>
<td>-1.7±0.6</td>
<td>-1.6±1.1</td>
<td>-1.1±0.7</td>
</tr>
<tr>
<td>DPP</td>
<td>Sampled max. prob.</td>
<td>last</td>
<td>0.6±1.0</td>
<td>0.2±0.6</td>
<td>-0.1±0.2</td>
</tr>
<tr>
<td>DPP</td>
<td>Probability variance</td>
<td>last</td>
<td>0.6±1.3</td>
<td>0.4±1.2</td>
<td>-0.4±0.6</td>
</tr>
<tr>
<td>DPP</td>
<td>BALD</td>
<td>last</td>
<td>0.5±1.6</td>
<td>0.2±1.5</td>
<td>-0.8±1.0</td>
</tr>
<tr>
<td>MC</td>
<td>Sampled max. prob.</td>
<td>all</td>
<td>0.6±0.2</td>
<td>2.1±0.6</td>
<td>1.4±0.7</td>
</tr>
<tr>
<td>MC</td>
<td>Probability variance</td>
<td>all</td>
<td>2.0±0.8</td>
<td>2.4±1.0</td>
<td>1.3±1.0</td>
</tr>
<tr>
<td>MC</td>
<td>BALD</td>
<td>all</td>
<td>2.3±1.0</td>
<td>2.4±1.2</td>
<td>1.1±1.0</td>
</tr>
</tbody>
</table>

Table 3: The misclassification detection performance (ROC AUC) (±SD) for the maximal probability baseline with the DistilBERT model and performance improvements over the baseline for various UEs. Statistically significant improvements (p-value ≤ 0.05) are highlighted.

## B Determinantal Point Processes

Determinantal point processes (DPPs) are specific probability distributions over a set of points. They allow choosing the subset of points enforcing the diversity between the samples. The DPPs were introduced for the needs of statistical physics (Macchi, 1975), and found their applications in machine learning (Kulesza and Taskar, 2012)

For example, consider the situation where we observe $N$ news from different outlets during one specific day. Let us also assume that we can measure the corresponding texts’ pairwise similarity. In this case, DPPs allow choosing a number $n \ll N$ of most non-similar news for the day, giving a good representation of the agenda. Most importantly, DPPs have efficient implementation for the exact sampling and several even more efficient approximate solutions. We also note that DPP sampling is stochastic, i.e., it provides a different result for each repetition. That is an essential property for the uncertainty estimation problems we consider in this work.

Formally, let us assume that the kernel matrix $K$ of pairwise similarities between the considered points $X$ is given. DPPs are similar to the algorithm of finding maximum volume submatrix of $K$ (Goreinov et al., 2010; Çivril and Magdon-Ismail, 2009) as geometrically determinant of the matrix is equal to the scaling volume of a corresponding linear transformation. In this case, a large volume is good because it corresponds to orthogonal vectors (i.e. non-similar vectors). Likewise, DPPs sample points $S$ with probabilities:

$$P[S \subset X] = \det K_S,$$

where $K_S$ is the submatrix of the matrix $K$ corresponding to points $S$.

As probability takes values between 0 and 1, the matrix $K$ needs to be positive semidefinite and should not have minors with determinant larger than 1. In practice, usually only some unnormalized likelihood matrix $L$ is given. The standard approach is to normalize it in the following way:

$$K = L(L + I)^{-1}.$$

In this case, we can directly calculate the submatrix probabilities:

$$P[X = S] = \frac{\det L_S}{\det(L + I)}.$$