A Novel Gradient Accumulation Method for Calibration of Named Entity Recognition Models

Grégor Jouet, Clément Duhart, Jacopo Staiano, Francis Rousseaux, Cyril de Runz

A Novel Gradient Accumulation Method for Calibration of Named Entity Recognition Models

Grégor Jouet*†§, Clément Duhart*,
*Pôle Universitaire Léonard de Vinci, Research Center
La Défense, France
{gregor.jouet, clement.duhart}@devinci.fr

Jacopo Staiano†
†reciTAL
Paris, France
{jacopo, gregor}@recital.ai

Francis Rousseaux‡
‡URCA CReSTIC, Moulin de la Housse
Reims, France
francis.rousseaux@univ-reims.fr

Cyril de Runz§
§University of Tours, LIFAT, BdTLN
Tours, France
cyril.derunz@univ-tours.fr, gregor.jouet@etu.univ-tours.fr

***Abstract*—The adoption of deep learning models has brought significant performance improvements across several research fields, such as computer vision and natural language processing. However, their "black-box" nature yields the downside of poor explainability: in particular, several real-world applications require – to varying extents – reliable confidence scores associated to a model's prediction. The relation between a model's accuracy and confidence is typically referred to as *calibration*. In this work, we propose a novel calibration method based on gradient accumulation in conjunction with existing loss regularization techniques. Our experiments on the Named Entity Recognition task show an improvement of the performance/calibration ratio compared to the current methods.**

***Index Terms*—calibration, ner, uncertainty, noise injection**

I. Introduction

For classification tasks, the output layer of a neural network typically performs a `softmax` over the computed logits in order to provide per-class decision probabilities. Nonetheless, these likelihoods are not necessarily correlated with the model's prediction confidence, a fact that can lead to misconceptions from the final user's point of view [9, 29]. The relation between the classifier's likelihoods and their actual confidence level is typically referred to as the *calibration* of the model. The Expected Calibration Error (ECE) [9] is one of the most popular calibration measures experimented with in various use cases, e.g. Question-Answering (QA) [12, 24], Computer Vision (CV) [23, 24] and Natural Language Processing (NLP) in general [13, 16].

Current literature presents two popular techniques for model calibration applied during training/fine-tuning of the entire model: Noise Injection (NI) [31] and Loss Regularization (LR) [14, 26, 38]. Commonly, these techniques are applied on a pretrained foundation model [2] such as BERT [5] in NLP applications.

In this work, we propose a novel calibration method applied only on the network's head without perturbing the foundation model's weights. Our approach is based on a gradient accumulation technique with a dedicated training objective and on the loss regularizer introduced by Xin et al. [38]. The proposed method is summarized in Figure 1. We evaluate the proposed method in terms of calibration and inference performance on the following well-established Named Entity Recognition (NER) datasets: Conll2003 [33], Wikiann [27], NCBI Disease [6], WNUT17 [27] and GUM [40].[1]

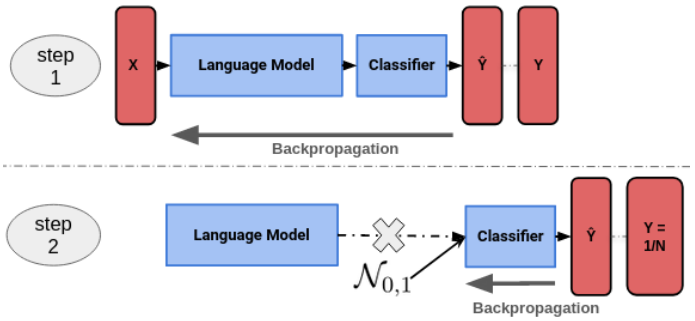

Figure 1. The gradient accumulation method general idea: the first step is a forward pass on the dataset element, the second step is the training of the classifier on noise data.

This work can give new perspectives on the calibration of fine-tuned models for NLP applications. This could lead to a better understanding of a model's outputs and ultimately to a higher trust level in the systems relying on these models.

Our contributions are as follows:

- A novel method for model calibration on token classification (e.g. NER) tasks based on gradient accumulation and a custom training objective, depicted in Figure 1.
- A benchmark of existing calibration techniques combined with our gradient accumulation method on several NER datasets.
- An open framework for extending this method to other classification tasks.

In the following, we first introduce important concepts regarding calibration and NER in Section II. Then, we go

[1]We publicly release the source code and scripts to reproduce this work at: https://github.com/DeVinci-Innovation-Center/ijcnn2022

over several recent advancements in model calibration in Section III. We then introduce the proposed Gradient accumulation method and explain the motivations behind its design choices and its implementation in Section IV. Finally, in Section V we showcase the results achieved by our method and compare them to several baselines.

II. Background

This section introduces the foundation of our method and details the specific use case we target throughout this paper.

A. Calibration

Calibration is a measurement of the discrepancy between the accuracy and confidence of a model's output. Several metrics have been proposed and used to measure calibration, such as reliability diagrams [9, 25] or the Brier score [39] which has been adjusted over time [15]. More recently, the research community has converged to adopt the Expected Calibration Error has as the metric of choice for the measurement of calibration [18, 20, 21, 35]. The ECE [9] uses bins to sort a model's outputs by confidence and measure the average difference between the accuracy and confidence of samples in each bin.

Formally, with N bins B_n containing samples within a certain confidence range, the ECE is defined as:

$$ECE = \sum_{n=1}^{N} \frac{|B_n|}{n} |acc(B_n) - conf(B_n)| \tag{1}$$

Where $|B_n|$ is the number of samples in the B_n bin. A perfectly calibrated model would be very confident (i.e. it would associate high probability to its output) when its prediction is correct, and conversely return probabilities close to $\frac{1}{N}$ on wrong predictions. The accuracy function is the fraction of correctly classified samples in the B_n bin. The confidence score represents the average confidence of samples in the bin B_n. Various methods have been introduced to measure such confidence. The ECE uses the *max softmax value* as the confidence of a sample. The methods used to calibrate a model often rely on the introduction of perturbations which make the model more robust, less affected by a dataset shift, and less susceptible to miscalibration; unfortunately, these are found to also often impact the performance. This motivates our approach: to prevent performance drop, calibration should be learned by the classification head of a NLP pipeline.

Miscalibration can have a direct negative impact on the explainability and user experience of the downstream process. To tackle this problem several sources of miscalibration have been identified such as model capacity [9], dataset quality [32] or out of domain training samples. Another source of miscalibration is the one-hot label encoding that tends to build overconfident output distribution even in case of low confidence.

B. Named Entity Recognition

NER is a popular token classification task: each token from the input text is classified as belonging to predefined categories; typically, entities are single words or groups of words referring to e.g. locations, persons, dates.

Various NLP pipelines rely, more or less explicitly, on this task. In QA [10], tokens and entities can be seen respectively as the words and their relation to the answer. In Document Representation [28], NER can be used to retrieve the text in proximity of entities of interest (e.g. persons, organizations).

Under the popular *BIO* schema, illustrated in Figure 2, the tokens are classified as the Beginning, the Inside, or being Outside of a specific entity. Naturally, in most NER datasets, the O label is over-represented, and thus the label distributions are heavily unbalanced; this makes NER tasks a challenging use case for calibration problems.

Figure 2. Sample text with NER token annotations using the *BIO* schema.

C. Transformer Integration

Nowadays, the Transformer architecture [36] and BERT [5] are considered as foundation models for task-specific fine-tuning or building more complex architectures. These language models provide dense contextual representations of the input text which have proven highly effective on several downstream tasks; nonetheless, they are found to be sensitive to miscalibration, especially in out of domain applications [4]. For this reason, a complementary calibration is necessary to make the model usable in the wild: the input samples from real world use cases can be out of the training distribution, and in these cases an appropriately calibrated model should output results reflecting the unknown nature of the sample.

Most recent applications leverage these representations with a Multi Layer Perceptron (MLP), using it as the classification head [13, 12]. In this work, we adopt the same setup to evaluate our proposed method.

III. Related Work

The current literature addresses the calibration issue mainly through Loss Regularization (LR) and Noise Injection (NI).

A. Loss Regularization

Loss Regularization (LR) is a family of techniques relying on a complementary loss term to regularize the weights or obtain a model with specific properties. LR has been used to tackle the calibration problem. For instance, the AVUC loss [14] penalizes samples whose accuracy and confidence do not match. The loss is computed from the number of correctly and incorrectly classified samples, as well as the number of strongly (high probability) and weakly (low probability) classified samples. The objective is to have all the accurate samples strongly classified and all inaccurate samples weakly

classified. The loss works for any classification task, but it lacks in stability and can hinder the training process.

Another approach to improve calibration with LR is to use Out Of Domain (OOD) examples. Mitrose et al. [22] compared in and out of domain examples using the cosine distance and used it as a regularization loss. Taking inspiration from the ECE, Tomani et al. [34] suggested an adversarial loss term. A different training procedure is suggested by Noh et al. [26] where examples are processed several times by the model with a different a dropout mask to produce different gradients which are then averaged. These approaches do not directly address the calibration problem, but have been used in recent literature to calibrate models [30].

Xin et al. [38] introduce a simple and effective loss, based on the uncertainty difference between samples selected by their loss value. For each sample x_i and x_j, their respective loss values e_i and e_j, and an uncertainty measure $g(x)$ the loss $\mathcal{L}_{reg}$ is applied during training:

$$\mathcal{L}_{reg} = \sum_{1<i,j<n} \Delta_{i,j} \mathbf{1}[e_i > e_j] \tag{2}$$

$$\Delta_{i,j} = max(0, g(x_i) - g(x_j))^2 \tag{3}$$

This loss is effective but has the downside of working on pairs of samples which puts the time complexity of the loss at $\mathcal{O}(n^2)$ where n is the batch size which can increase the training time under certain conditions.

B. Noise Injection

Noise Injection (NI) relies on the alteration of a model's inputs or internal neuron activations via the addition or multiplication of noise. NI has been used to improve a model generalization capability or to prevent adversarial attacks [8].

The idea of modifying input data to improve a model accuracy was applied with great success to the domain of image classification [31]. The concept is more difficult to apply to the NLP field but attempts have been made using synonyms or word replacement approaches [7].

Other NI techniques focus on modifying the data flow inside the model regardless of the input data. This is mainly used to tackle the overfitting problem [41] as the data signal can't create a single path to the output, eliminating trivial solutions and preventing overfitting. NI is also used to increase the model accuracy and general performance. In the context of adversarial attacks, the noise is used to improve the robustness of models so small perturbations do not have significant negative impact on the results [11].

These different results have also been studied for their regularization properties. Gaussian Noise Injection (GNI) [19] has been shown to have regularizing properties that improve calibration; these results have been further studied and applied to different tasks [3].

IV. Gradient Accumulation Proposal

Our proposed method addresses the miscalibration induced by the one-hot label encoding and the out of domain application.

A. Requirements

To address the miscalibration issue, we consider our proposal should meet the following requirements:

- The method should not require a model adaptation, additional implementations, nor adaptation of the language model. The proposed method could potentially be used to train any network, and not to redo the training on data, nor to redesign/re-implement the network. The method should therefore be agnostic to the used model to guarantee its generalization and exploitation to other tasks.
- The method should have a low training overhead. This ensures the training time stays within reason, even on large scale datasets. The idea is that the overhead training should help to better calibrate the classifier head for the application domain at training time.
- The Gaussian noise sample training must only be applied to the classifier head. Applying this method to the entire model could hinder the performance of the pre-trained model and precipitate performance loss at the expanse of calibration.

B. General Principle

In order to meet the previous requirements, we propose a straightforward method which consists in accumulating on the classification head the training gradients for each sample with those from a Gaussian noise sample. This technique helps the system to maintain calibrated probabilities on the one-hot label vector for out of domain samples.

We consider this technique as a model regularization based on uniform labels for the loss function, which can be seen as a complementary training objective on out-of-domain data.

To resume, we accumulate the gradient from a training sample on one-hot vector, representing a data point with absolute certainty, with a Gaussian noise sample with uniform labels -which contains no information. This contrast between an absolutely certain and absolutely uncertain data point is averaged by the gradient accumulation, explaining the improvement of the calibration.

C. Implementation

As previously explained, we train the classification head on synthetic noise data with equiprobable target labels. The classifier is trained on the dataset at hand and synthetic data sequentially.

First, a sample is fed-forward to the entire model: Language Model (LM) and classification head, then a backpropagation step computes gradients for the entire model on that sample. Finally, synthetic data is injected in the classification head with equiprobable labels as targets. We then perform another backpropagation step, accumulating the gradients. This accumulates the gradients for the *normal* forward pass with a sample from the dataset and the gradients from the synthetic data on the classification head. We summarize this procedure in Figure 1.

This implementation choice stems from the requirements explained in Section IV-A. The simple and effective idea

behind this method is to have a label for *negative*, impossible to classify examples. These samples serve as counter examples to one-hot encoded labels of the dataset. Since it is hard to create noise text data that would match a real-world use case, we prefer to train the head classifier on noise data rather than the whole model. Because we want only the classifier to learn calibration data, we chose to feed noise to the classification head and not the entire model.

More formally, given a dataset $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}$ with y_i being the target vector for N classes. We define a model as a LM $\mathbf{x} \mapsto LM(\mathbf{x})$ composed with a classifier $\mathbf{z} \mapsto C(\mathbf{z})$ The $\mathbf{z}$ component is a vector with the same dimension as the output of the LM. We note it $\mathcal{M} = \mathbf{x} \mapsto C(LM(\mathbf{x}))$. We define $\hat{\mathbf{z}}$ as a gaussian noise vector with the same dimension as the output of the LM.
From these elements we obtain the two different losses:

$$\mathcal{L}_{task} = MSE[\mathcal{M}(\mathbf{x}_i), \mathbf{y}_i] \tag{4}$$

$$\mathcal{L}_{calib} = \lambda(\frac{1}{N} - max[C(\hat{\mathbf{z}})])^2 \tag{5}$$

$\mathcal{L}_{task}$ is the loss associated to the classification task, $\mathcal{L}_{calib}$ is the loss used when training the classification head on noise input. We use the Mean Square Error (MSE) loss in our experiments but other loss functions compatible with classification tasks can be used. λ is a hyperparameter, chosen before training.

Algorithm 1 Training procedure for one step

Input A model $\mathcal{M}$ and dataset point $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}$

Output A model $\mathcal{M}$ with accumulated gradients ∇_{LM} and ∇_C for its Language Model (LM) and classification head.

1: Do a forward pass: $\hat{\mathbf{y}}_i = \mathcal{M}(\mathbf{x}_i)$
2: Compute $\mathcal{L}_{task}$
3: Apply backpropagation on $\mathcal{M}$ using $\mathcal{L}_{task}$. This computes ∇_{task} for LM and C
4: Generate gaussian noise input $\hat{\mathbf{z}} \sim \mathcal{N}(0; 1)$
5: Compute $C(\hat{\mathbf{z}})$ then $\mathcal{L}_{calib}$
6: Apply backpropagation on C computing ∇_{calib}.
Now, the LM has gradients $\nabla_{LM} = \nabla_{task}$ and C has gradients $\nabla_C = \nabla_{task} + \nabla_{calib}$
7: $\nabla_C \leftarrow \frac{\nabla_C}{2}$
8: Return $\mathcal{M}$

The training procedure for a single element of the dataset is described in the Algorithm 1. The model is trained on its task and noise at the same time and gradients for both the task and noise passes are accumulated on the classification head. Since we accumulate two gradients on the classifier but only one on the LM, we divide ∇_C by 2 in step 7. We repeat this step for all the samples in the dataset. Effectively, at the classification head, each sample from the dataset will have a negative counterpart sampled from a Gaussian distribution. At the end of the training process, we will have simultaneously trained the classifier on both tasks by the same amount.

D. Combination with abstention

Our method can be combined with previous work, specifically the *Abstention* regularization loss showcased by Xin et al. [38]. Combining the two methods requires doing multiple passes of the same sample from the dataset and accumulating the gradients, on the classification head only, for the different passes. With the *Abstention* method, this implies backpropagating the regularization loss on the classification head only. Effectively, we are adapting the *Abstention* mechanism on an NER task. This adaptation is possible because the two methods work on classification tasks: the *Abstention* method was originally used on sentence classification. To adapt it to a NER context, we compute $\mathcal{L}_{reg}$ on each token and use the mean value of these losses as the new regularization. Since we do an additional backpropagation step for the *Abstention* pass, we divide the gradient of the classification head by 3 instead of 2 in the Algorithm 1 when this method is used. The classification head gradients become the mean of the dataset sample pass, the abstention pass and the pass with our method.

V. Results

In this section, we test our method on multiple datasets and compare it to different baselines. We first explain the experimental conditions in which we ran the experiments. We introduce a new metric to reflect the performance over calibration ratio which is an important aspect of the calibration objective. Finally, we present our results and compare them to the current state of the art.

A. Experimental Setup

Several datasets are commonly used as a baseline for the NER task. The Conll2003 dataset [33] is widely used in the community and is a popular choice for Deep Learning approaches, as it comprises several hundreds of annotated news articles. The NCBI Disease dataset [6] is a NER dataset in the medical domain, to enable disease named entity recognition based on a large, manually annotated corpus of medical publications. It is regularly used as a baseline in NER applied to the medical field. The Wikiann dataset [27] was originally designed for cross-lingual name tagging and is based on a large corpus from Wikipedia. We do not use the cross-lingual aspect of the dataset and only use the English article for NER. The GUM dataset [40] contains a rich and varied ensemble of annotation types and is a challenging dataset because of this diversity. Finally, the WNUT 2017 dataset [1] is a NER dataset on emerging and rare entities, this feature is the reason why we chose to include this dataset in spite of the very low performance of all models (F1 30-58%), the rare and unseen characteristic of the data is the key element measure by the ECE metric, this is effectively, in the author's opinion the dataset which leverages the method the best and is the closest thing to a real-world application. These different datasets are available online and can be downloaded using the HuggingFace Dataset library[2] increasing the reproducibility of our work.

[2]https://huggingface.co/docs/datasets/

Table I
SUMMARY OF RESULTS WITH A CLASSIFICATION HEAD OF DEPTH = 1 AND VARYING WIDTH (w)

Method	ConLL2003			NCBI Disease			Wikiann			GUM			Wnut17		
	F1↑	ECE↓	APCR↑	F1↑	ECE↓	APCR↑	F1↑	ECE↓	APCR↑	F1↑	ECE↓	APCR↑	F1↑	ECE↓	APCR↑
Baseline															
$w = 64$	**.938**	.012	74.83	.850	.014	51.18	**.839**	.069	10.13	**.624**	.194	2.01	.541	.045	6.46
$w = 128$	**.939**	**.012**	75.12	.857	.014	52.69	.835	.069	10.00	.612	.204	1.83	.557	.046	6.75
$w = 200$	.936	.013	68.91	**.857**	.012	53.35	.830	.070	9.75	.617	.203	1.87	**.564**	.044	7.17
$w = 256$	**.936**	.013	69.25	.849	.014	49.46	.831	.070	9.84	.610	.199	1.86	**.544**	.045	6.45
AVUC															
$w = 64$	.834	**.006**	**106.88**	.477	.011	19.13	.557	**.037**	8.29	.250	**.061**	1.02	.309	**.028**	3.39
$w = 128$	.845	.010	71.07	.520	.014	18.72	.564	**.049**	6.38	.317	**.049**	**2.03**	.441	**.028**	6.83
$w = 200$	.845	**.007**	**101.13**	.517	.012	21.17	.563	**.040**	7.92	.320	**.043**	**2.38**	.444	**.026**	**7.67**
$w = 256$	.846	**.007**	98.13	.523	.013	21.21	.562	**.039**	7.94	.334	**.041**	**2.67**	.462	**.026**	**8.12**
Abstention (ABS)															
$w = 64$	.932	.013	65.58	.860	.013	55.64	.826	.072	9.44	.604	.204	1.78	.409	.053	3.13
$w = 128$	.938	**.011**	**75.03**	.854	.013	53.59	.831	.070	9.84	.612	.199	1.87	.346	.051	2.31
$w = 200$	.931	.013	62.97	.849	.014	50.09	.828	.071	9.63	.602	.203	1.78	.404	.048	3.40
$w = 256$	.931	.013	63.45	**.857**	.013	54.57	.833	.069	9.96	.618	.204	1.87	.516	.053	4.98
∇Acc. (Ours)															
$w = 64$	.932	.013	65.44	.848	.014	49.67	.835	.069	10.08	.609	.199	1.86	**.582**	.044	**7.69**
$w = 128$	.934	.013	65.60	**.858**	**.013**	55.13	**.839**	.066	**10.57**	.612	.198	1.89	**.561**	.043	**7.24**
$w = 200$	**.937**	.012	68.16	.842	.014	50.28	**.833**	.069	10.01	.609	.200	1.84	.564	.044	7.16
$w = 256$	.943	.012	73.15	.850	.013	52.61	**.836**	.068	10.17	.619	.197	1.94	.569	.043	7.47
∇Acc. (Ours) + ABS															
$w = 64$	.930	.008	104.52	**.868**	**.010**	**74.98**	.829	.065	**10.57**	.614	.184	**2.05**	.542	.040	7.22
$w = 128$	.933	.013	64.67	**.858**	.013	**53.84**	.825	.072	9.44	**.614**	.200	1.88	.522	.046	5.86
$w = 200$	.934	.008	97.58	.844	**.011**	**62.35**	.830	.065	**10.58**	**.624**	.185	2.10	.487	.042	5.63
$w = 256$	.932	.008	**106.31**	.844	**.009**	**73.45**	.832	.063	**10.98**	**.623**	.187	2.07	.497	.042	5.87

To validate our method, we use different metrics: first, we want the model to retain its performance; consistently with the literature on NER, we use the F1 score defined as:

$$F1 = 2\frac{recall * accuracy}{recall + accuracy}$$

The accuracy and recall are measured at the token level of the NER task, the F1 score represents the performance of the NER model to classify tokens and not word compounds. The F1 is a scalar metric between 0 and 1, the higher the better. The measure of the calibration of the classifier is made with the ECE metric [9], as previously detailed in the Section II; the ECE represents the discrepancy between the accuracy and `softmax` output of the classifier. The ECE is a scalar metric between 0 and $+\infty$. A perfectly calibrated model will yield an ECE of 0. As we will detail in the Section V-B, calibration can be achieved at the expense of the model performance. To reflect this phenomenon, we introduce a new metric which accounts for both the model's performance and the calibration value. The Adjusted Performance to Calibration Ratio (APCR) is defined as:

$$APCR = \frac{F1^2}{ECE}$$

The APCR is a ratio between the performance and calibration of a model. To better reflect the performance drop of models, we adjusted the metric and used the square of the F1 score. Since the performance drop can be significant and invalidate a result as the model becomes unusable in practice, we want a metric that reflects this implication strongly. The APCR provides a preview of the overall performance of the method. The APCR metric is a scalar $\geqslant 0$, higher values are better. APCR is not defined when $ECE = 0$, however, perfectly calibrated models do not arise in practice. We consider the ECE to be strictly positive in this paper.

We conduct our experiments in different settings to validate our approach: the *raw* setting represents a training of the model without any regularization and is used as the baseline for all the different results. The *AVUC* setting only uses the AVUC [14] regularization. As described in the original paper, we backpropagate the loss on the entire model, LM and classification head. The *Abstention* setting refers to the approach introduced by Xin et al. [38], applied as described in the original paper. The $\nabla Acc.$ setting refers to our method as described in Section IV and finally the $\nabla Acc.$+Abstention setting refers to the combination of our approach and the *Abstention* method and is implemented as described in Section IV-D.

Since our method is applied to the classification head specifically, it is important to test different architectures and capacities. We conducted our experiments on a MLP with different depth and width. The width is the number of neurons in a single layer and the depth is the number of layers in the MLP. The classification head is usually a single layer perceptron (depth=1), which is the adopted setting in the Table I. We chose values between 64 and 256 for the classifier width and values between 1 and 6 for its depth. The method's behavior for depth over 1 are reported on Figure 4.

Finally, we provide details on the hyperparameters used. In our experiments, the LM is *BERT-base* from the HuggingFace [37]. We measure the ECE on the test fraction of each dataset with the *netcal* [17] library while the F1 score is

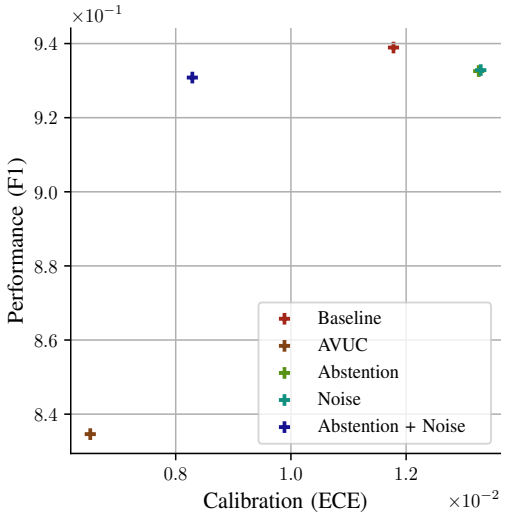

Figure 3. Performance VS Calibration of several methods on the Conll2003 dataset: each model's ECE and F1 score are plotted to visualize the performance/calibration trade-off between the different methods. The calibrated and performant models are on the top left.

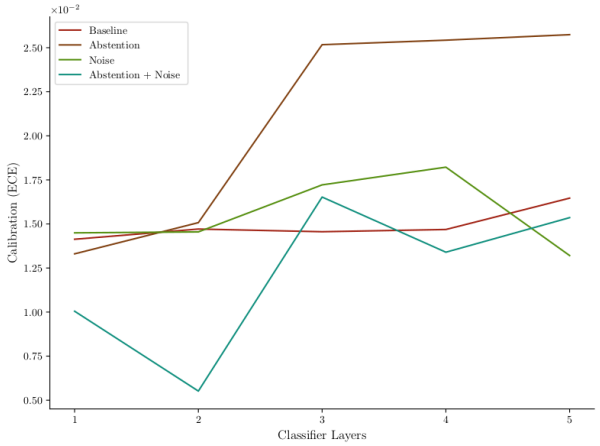

Figure 4. ECE Calibration for different methods on the *Ncbi_Disease* dataset w.r.t classifier depth. The classifier width remains 128 throughout, we increase the depth of the classifier.

computed by the internal metrics provided by HuggingFace. We choose $\lambda = 2.5^{-2}$ for all experiments for the loss scaler. The learning rate is linearly decreased over the training by a scheduler, starting at $5e^{-5}$. The vector size of $\mathbf{z}$ is 768 for all experiences. For all training we chose a batch size of 3. Training is performed on a Nvidia Tesla V100 on 10 epochs.

B. Performance vs. Calibration

Model calibration plays an important role in explainability and improves the user's trust in the predictions, but the model's performance remains a primordial metric. If the performance drops significantly, the model becomes unusable regardless of its calibration. In practice, we notice a trade-off between the performance of the model and the calibration measured by the ECE. The reason for this trade-off can stem from the potency of the method or the strength applied to the regularizer, i.e the loss scale.

The most potent calibration methods can impact the model's performance the most. Therefore, the ECE metric on its own is not enough to get information about the model's usability in a real world setting as the method could have had a detrimental effect on the model's performance. For this reason, we introduced the APCR metric, to have a joint estimation of the model's performance and interpretability by end users.

C. Discussion

The results of our experiments are reported in Table I. The last two rows depict our method and its combination with the Abstention mechanism [38], respectively. We obtain satisfactory results on all datasets and architectures: the APCR, representing the trade-off between performance and accuracy, is better 12 times out of 20. The F1 score is also equal or better in 12 case out of 20. As we expected, the ECE metric is better only in 4 cases out of 20. As presented, the AVUC metric yields better calibration but models that are ill-suited for a real world application due to their performance. This shows our method gives a better trade-off and models better suited for real world application. Even though we provided motivation for our APCR metric, we can observe the raw ECE value is not always better and that our method does not strongly constrain the model's calibration. We can also observe the impact of the AVUC regularization on the model's performance, which despite yielding better calibration, severely hinders the model's accuracy.

Because the classifier head is initialized with Gaussian noise and the weights of the language model are trained with normalization components such as Batch Normalization, we chose uniform Gaussian noise as our noisy input for the classification head. The two training objectives pull the gradient in different directions: the first pass trains the classifier for the given task, the second pass trains the classifier to give equiprobable outputs for noisy inputs. We chose to inject the noise only at the classification head because the calibration of the model is determined only by the final layers, the language model gives high dimensional representations and only the classifier is responsible for the calibration of the model.

To illustrate our results, we plotted the Performance vs. Calibration of several methods for the Conll2003 dataset in Figure 3. We can observe the trade-off with a set of techniques more accurate but less calibrated and on the other side, better calibrated but less performant methods. The combination of our method and Abstention appears to be the best trade-off between performance and calibration.

We experimented with different kinds of classification head architectures: Table I shows results with different width of a 1 layer MLP, while Figure 4 depicts the evolution of the calibration w.r.t the depth of an MLP of width 128. We observe that all methods have a tendency to converge when the depth increases, except the AVUC method, which seems to degrade. This could be interpreted as the calibration information being insufficient for the model capacity, or the model capacity

getting too important for the task, leading to overfitting.

VI. Conclusion

We presented a novel gradient accumulation method for calibrating NER models in order to address the miscalibration issue. To address this issue and to gain in generalization, the method should be agnostic to the trained model and network's architecture, only have a low training overhead, and exploit synthetic noise data only for the network head training. Our proposal consists in training a classifier on domain data and synthetic Gaussian noise data, and then combining accumulating the gradients.

We tested this method on several NER datasets. The comparison with the initial model (baseline) showed that our proposal, with and without Abstention, improves, most of the time, the calibration in terms of ECE, and the performance in terms of F1 score. To analyze the performance/calibration trade-off of a model, we introduced the APCR metric combining F1 scores and ECE. According to this metric, our approach provided globally a better balance between calibration and performance than the current state-of-the-art approaches. Therefore, the experimentation has shown that our approach yields results more suitable for real-world applications.

Future works include testing the method on a wider variety of tasks such as sentiment classification, language generation and on other foundation-based architectures. Further investigation is needed on the impact of the loss scale on the APCR. The impact of other parameters such as the loss or learning rate used can be further analyzed in future works.

References

[1] Gustavo Aguilar et al. "A Multi-task Approach for Named Entity Recognition in Social Media Data". In: *Proc. of the 3rd W-NUT*. Copenhagen, Denmark: ACL, 2017, pp. 148–153. DOI: 10.18653/v1/W17-4419.

[2] Rishi Bommasani et al. "On the Opportunities and Risks of Foundation Models". In: *ArXiv* (2021). URL: https://arxiv.org/abs/2108.07258.

[3] Alexander Camuto et al. "Explicit Regularisation in Gaussian Noise Injections". In: *NeurIPS 2020*. Ed. by Hugo Larochelle et al. virtual, 2020.

[4] Shrey Desai and Greg Durrett. "Calibration of Pre-trained Transformers". In: *Proc. of EMNLP*. Online: Association for Computational Linguistics, 2020, pp. 295–302. DOI: 10.18653/v1/2020.emnlp-main.21.

[5] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proc. of NAACL-HLT*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423.

[6] Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. "NCBI Disease Corpus: A Resource for Disease Name Recognition and Concept Normalization". In: *Journal of Biomedical Informatics* 47 (2014), pp. 1–10. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2013.12.006.

[7] Steven Y. Feng et al. "A Survey of Data Augmentation Approaches for NLP". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Online: Association for Computational Linguistics, 2021, pp. 968–988. DOI: 10.18653/v1/2021.findings-acl.84.

[8] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples". In: *Proc. of ICLR*. Ed. by Yoshua Bengio and Yann LeCun. 2015.

[9] Chuan Guo et al. "On Calibration of Modern Neural Networks". In: *Proc. of ICML*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 1321–1330.

[10] Jiafeng Guo et al. "Named Entity Recognition in Query". In: *Proc. ACM SIGIR2009*. SIGIR '09. New York, NY, USA: Association for Computing Machinery, 2009, pp. 267–274. ISBN: 978-1-60558-483-6. DOI: 10.1145/1571941.1571989.

[11] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. "Parametric Noise Injection: Trainable Randomness to Improve Deep Neural Network Robustness Against Adversarial Attack". In: *CVPR 2019*. Computer Vision Foundation / IEEE, 2019, pp. 588–597. DOI: 10.1109/CVPR.2019.00068.

[12] Zhengbao Jiang et al. "How Can We Know When Language Models Know? On the Calibration of Language Models for Question Answering". In: *Transactions of the Association for Computational Linguistics* 9 (2021), pp. 962–977. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00407.

[13] Taehee Jung et al. "Posterior Calibrated Training on Sentence Classification Tasks". In: *Proc. of ACL*. Online: Association for Computational Linguistics, 2020, pp. 2723–2730. DOI: 10.18653/v1/2020.acl-main.242.

[14] Ranganath Krishnan and Omesh Tickoo. "Improving model calibration with accuracy versus uncertainty optimization". In: *NeurIPS2020*. Ed. by Hugo Larochelle et al. 2020.

[15] Meelis Kull and Peter Flach. "Novel Decompositions of Proper Scoring Rules for Classification: Score Adjustment as Precursor to Calibration". In: vol. 9284. 2015, pp. 68–85. ISBN: 978-3-319-23527-1. DOI: 10.1007/978-3-319-23528-8_5.

[16] Ananya Kumar, Percy Liang, and Tengyu Ma. "Verified Uncertainty Calibration". In: *NeurIPS2019*. Ed. by Hanna M. Wallach et al. 2019, pp. 3787–3798.

[17] Fabian Küppers et al. "Multivariate Confidence Calibration for Object Detection". In: *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2020.

[18] Kimin Lee et al. "Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples". In: *Proc. of ICLR*. OpenReview.net, 2018.

[19] Yinan Li and Fang Liu. "Adaptive Gaussian Noise Injection Regularization for Neural Networks". en. In: *Advances in Neural Networks – ISNN 2020*. Ed. by

Min Han, Sitian Qin, and Nian Zhang. Lecture Notes in Computer Science. Springer International Publishing, 2020, pp. 176–189. ISBN: 978-3-030-64221-1. DOI: 10.1007/978-3-030-64221-1_16.

[20] Wesley J. Maddox et al. "A Simple Baseline for Bayesian Uncertainty in Deep Learning". In: *NeurIPS 2019*. Ed. by Hanna M. Wallach et al. 2019, pp. 13132–13143.

[21] José Mena, Oriol Pujol, and Jordi Vitrià. "A Survey on Uncertainty Estimation in Deep Learning Classification Systems from a Bayesian Perspective". In: *ACM Computing Surveys* 54.9 (2021), 193:1–193:35. ISSN: 0360-0300. DOI: 10.1145/3477140.

[22] John Mitros and Brian Mac Namee. "On the Importance of Regularisation & Auxiliary Information in OOD Detection". In: *ICONIP 2021* (2021). DOI: 10.1007/978-3-030-92310-5_42.

[23] Allan H. Murphy. "A New Vector Partition of the Probability Score". In: *Journal of Applied Meteorology and Climatology* 12.4 (June 1973). Publisher: American Meteorological Society Section: Journal of Applied Meteorology and Climatology, pp. 595–600. DOI: 10.1175/1520-0450(1973)012⟨0595:ANVPOT⟩2.0.CO;2.

[24] Khanh Nguyen and Brendan O'Connor. "Posterior calibration and exploratory analysis for natural language processing models". In: *Proc. of EMNLP*. Lisbon, Portugal: Association for Computational Linguistics, 2015, pp. 1587–1598. DOI: 10.18653/v1/D15-1182.

[25] Alexandru Niculescu-Mizil and Rich Caruana. "Predicting good probabilities with supervised learning". In: *Proc. of ICML*. Ed. by Luc De Raedt and Stefan Wrobel. Vol. 119. ACM International Conference Proceeding Series. ACM, 2005, pp. 625–632. DOI: 10.1145/1102351.1102430.

[26] Hyeonwoo Noh et al. "Regularizing Deep Neural Networks by Noise: Its Interpretation and Optimization". In: *NeurIPS 2017*. 2017, pp. 5109–5118.

[27] Xiaoman Pan et al. "Cross-lingual Name Tagging and Linking for 282 Languages". In: *Proc. of ACL*. Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 1946–1958. DOI: 10.18653/v1/P17-1178.

[28] Desislava Petkova and W. Bruce Croft. "Proximity-Based Document Representation for Named Entity Retrieval". In: *CIKM 2007*. CIKM '07. Lisbon, Portugal: Association for Computing Machinery, 2007, pp. 731–740. ISBN: 9781595938039. DOI: 10.1145/1321440.1321542.

[29] Schaekermann, Mike. "Human-AI Interaction in the Presence of Ambiguity: From Deliberation-based Labeling to Ambiguity-aware AI". PhD thesis. 2020.

[30] Seonguk Seo, Paul Hongsuck Seo, and Bohyung Han. "Learning for Single-Shot Confidence Calibration in Deep Neural Networks Through Stochastic Inferences". In: *CVPR2019*. Computer Vision Foundation / IEEE, 2019, pp. 9030–9038. DOI: 10.1109/CVPR.2019.00924.

[31] Connor Shorten and Taghi M. Khoshgoftaar. "A Survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* 6.1 (2019), p. 60. ISSN: 2196-1115. DOI: 10.1186/s40537-019-0197-0.

[32] Jasper Snoek et al. "Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift". In: *NeurIPS 2019*. Ed. by Hanna M. Wallach et al. 2019, pp. 13969–13980.

[33] Erik F. Tjong Kim Sang. "Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition". In: *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*. 2002.

[34] Christian Tomani and Florian Buettner. "Towards Trustworthy Predictions from Deep Neural Networks with Fast Adversarial Calibration". In: *AAAI*. 2021.

[35] Juozas Vaicenavicius et al. "Evaluating model calibration in classification". In: *AISTATS 2019*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 2019, pp. 3459–3467.

[36] Ashish Vaswani et al. "Attention is All you Need". In: *NeurIPS2017*. Ed. by Isabelle Guyon et al. 2017, pp. 5998–6008.

[37] Thomas Wolf et al. "Transformers: State-of-the-Art Natural Language Processing". In: *Proc. of EMNLP2020: System Demonstrations*. Online: Association for Computational Linguistics, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[38] Ji Xin et al. "The Art of Abstention: Selective Prediction and Error Regularization for Natural Language Processing". In: *Proc. of ACL*. Online: Association for Computational Linguistics, 2021, pp. 1040–1051. DOI: 10.18653/v1/2021.acl-long.84.

[39] Bianca Zadrozny and Charles Elkan. "Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers". In: *Proc. of ICML2021*. Ed. by Carla E. Brodley and Andrea Pohoreckyj Danyluk. Morgan Kaufmann, 2001, pp. 609–616.

[40] Amir Zeldes. "The GUM Corpus: Creating Multilayer Resources in the Classroom". In: *Language Resources and Evaluation* 51.3 (2017), pp. 581–612. DOI: http://dx.doi.org/10.1007/s10579-016-9343-x.

[41] Richard M. Zur et al. "Noise injection for training artificial neural networks: A comparison with weight decay and early stopping". In: *Medical Physics* 36.10 (2009), pp. 4810–4818. ISSN: 0094-2405. DOI: 10.1118/1.3213517.