DuluthNLP at SemEval-2021 Task 7: Fine-Tuning RoBERTa Model for Humor Detection and Offense Rating

Samuel Akrah
Department of Computer Science
University of Minnesota Duluth
Duluth, MN 55812 USA
akrah001@d.umn.edu

Abstract

This paper presents the DuluthNLP submission to Task 7 of the SemEval 2021 competition on Detecting and Rating Humor and Offense. In it, we explain the approach used to train the model together with the process of fine-tuning our model in getting the results. We focus on humor detection, rating, and offense rating, representing three out of the four subtasks that were provided. We show that optimizing hyper-parameters for learning rate, batch size and number of epochs can increase the accuracy and F1 score for humor detection.

1 Introduction

Humor detection poses a challenge to humans, not least because of the mix of irony, sarcasm, and puns which underlie humor. To understand the funniness of humor requires a certain grasp of context, culture and, for some, even country.

If rating the funniness of humor is any challenge, ranking its offensiveness is even more so, especially when doing so requires an appreciation of the sensibilities of the humor target – whether race, religion, and/or gender – and, in most cases, context.

It is little wonder humor detection has been central to NLP tasks in the past few years. The last few SemEvals have featured tasks focused exclusively on either detecting the funniness of humor (Hossain et al., 2020; Van Hee et al., 2018; Potash et al., 2017) or detecting offense (Zampieri et al., 2019). What SemEval-2021 task 7 seeks to do is combine the detection of both humor and offense for a given corpus.

Our approach uses pretrained RoBERTa model (Liu et al., 2019b) trained on a RoBERTa classifier implemented by HuggingFace (Wolf et al., 2019). The intuition here is that RoBERTa model achieves state of the art performance for tasks requiring contextual information. Our goal is to measure the effect of varying three hyperparameters – batch size, learning rate, epoch size - whilst maintaining default values for others. Our results show that varying the three hyperparameters can increase performance for humor detection, humor ranking, and offense rating. The codebase for our participation of this SemEval Task is available on github [1]

2 Related Work

Earlier works on humor detection and rating showed modest gains. With the advent of attention mechanism (Vaswani et al., 2017), though, and the transformer model a few years later (Dai et al., 2019), not only has interest in the NLP community on humor detection has soared, but performance on humor and offense detection has increased (Weller and Seppi, 2019).

This has been particularly so in the last few years, where humor detection and offense rating have been featured in some of SemEval tasks. SemEval-2019 Task 6 on offense rating (Zampieri et al., 2019) attracted 800 participants and 115 submissions, the interest prompting a second SemEval-2020 Task 12 the following year (Zampieri et al., 2020). Around the same period, humor rating have attracted similar interest (Hossain et al., 2020) in SemEval Tasks. To the best of our knowledge, however, SemEval-2021 Task 7 (Meaney et al., 2021) is the first to measure humor and offense for a given task.

Most of the winning teams (Morishita et al., 2020; Rozental and Biton, 2019; Wiedemann et al., 2020), for both humor and offense rating alike, implemented BERT and its variants, including Albert and RoBERTa, in their model, an approach that tended to yield the best results. And more often than not, the teams exploit ensembles of BERT,

[1] https://github.com/akrahdan/SemEval2021

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 1196–1203
Bangkok, Thailand (online), August 5–6, 2021.

GPT-2, RoBERTa and their variants (Morishita et al., 2020), whilst others stick to a single pretrained model.

Our approach in this task is to use RoBERTa model, an approach that will fine-tune a select number of hyperparameters and to measure the model performance for every change of hyperparameter set.

3 System overview

In this section, we review our system's adoption of the pretrained RoBERTa model (Liu et al., 2019a) for SemEval tasks. We also describe the Bayesian hyperparameter optimization technique, which we used in our hyperparameter sweeps for selecting optimal values for learning rate, batch size, and epoch cycles.

3.1 Model description

Our system's adoption of RoBERTa model is based on its ability to achieve state-of-the-art performance for most NLP tasks with minimal effort, including, in our case, humor detection. The RoBERTa model, itself a re-implementation of BERT (Devlin et al., 2019), is first pre-trained on unlabeled text corpus and subsequently fine-tuned on downstream tasks with labeled data.

The RoBERTa model is a significant improvement over the BERT model, and it differs from BERT for its usage of dynamic masking for training, Next Sentence Prediction (NSP), and a larger mini-batch size (which, it has been observed, correlates with performance (Liu et al., 2019a)).

Again, the RoBERTa model outperforms BERT for its size and diversity of data used in pretraining, with its 160GB of training data drawn from multiple sources compared to BERT's 16GB of training data.

Our system implements the RoBERTa model with a classification layer on top using HuggingFace transformer model[2].

We adopt Bayesian optimization to automate the selection of optimal hyperparameter values for the training and evaluation of the three Subtasks. Details of the Bayesiann optimization method are found in Appendix A

3.2 Sweeps

We use Bayesian optimization to run hyperparameter sweeps for our model, but not before manually selecting a sensible list of hyperparameter values in fine-tuning the RoBERTa model (Liu et al., 2019a) on the SemEval tasks, including a learning rate of 0.000025, a batch size of 4, all for 16 epochs. The initial weights are based on standards followed by BERT and RoBERTa (Devlin et al., 2019; Liu et al., 2019a). The remaining parameters are based on open source implementation by HuggingFace[3] (Wolf et al., 2020).

Whilst the initial approach of selecting sensible defaults for hyperparameters achieved state of the art results, the random, manual process was painful, the results sometimes unpredictable. Applying Bayesian optimization, however, in running a sweep over a range of hyperparameter values helped in the selection of hyperparameters, including epoch size, batch size, and learning rates, across the 24 layers of RoBERTa$_{LARGE}$ model. Our system's implementation of the Bayesian Optimization is based on the open source wandb client by Weights & Biases[4] (Biewald, 2020)

4 Experimental Setup

In this section, we present the experimental setup for the our model and hyperparameter sweeps for the SemEval tasks.

4.1 Implementation

For all experiments for the RoBERTa model (using RoBERTa$_{BASE}$ and RoBERTa$_{LARGE}$ varients) , we use the PyTorch[5] implementation of it in the HuggingFace transformer open source library[6] (Wolf et al., 2020) together with the simpletransformer[7] (Rajapakse, 2019) wrapper library. We maintain the default weights and hyperparameters whilst only changing the learning rate, batch size and finetuning for different values of epochs. All experiments were run on V100 GPUs with 16GB memory.

4.2 Data Processing

The training and evaluation data, as shown in Table 1 is based on the training, development and test data supplied for the SemEval-2021 Task 7 (Meaney et al., 2021). Train data for each Substask is the same but the the number of annotations are

[2]https://huggingface.co/transformers/

[3]https://github.com/huggingface/transformers

[4]https://github.com/wandb/client

[5]https:
pytorch.org

[6]https://github.com/huggingface/transformers

[7]https://simpletransformers.ai

Task	Type	Metric	Train
Task 1a	Classification	F1-Score	8000
Task 1b	Regression	RMSE	4932
Task 2a	Regression	RMSE	8000

Table 1: A Summary of Subtasks and dataset. Both development and test set have the same have sizes of 1000.

different, with 8000 annotations for Subtask 1a and 2a; and 4932 annotations for Subtask 1b. Data splits between training and evaluation is 80% and 20%.

The annotators for the dataset for the tasks were a diverse group of individuals from differing age group (18-70), genders, political views and income levels – their backgrounds reflecting their perceptions of jokes or humor. For each text in the dataset, annotators were asked to rank as either humorous or not, and to rate the humor level on a scale of 1 to 5.

The subjectivity level of each text is also captured as a controversy score. Each text is labeled as controversial if the variance of its humor rating is greater than the median variance of all texts. Otherwise, it is labeled as not controversial.

As a way of combining humor and offense detection in the same task, a first in SemEval tasks, annotators were asked to classify humor as either offensive or not, and, if offensive, to rate the offensiveness on a a scale of 1 to 5. Non-offensive humor received a zero rating.

Overall, the SemEval task divides into four subtasks. Task 1a, a binary task, predicts if a given text should be considered humorous. The second Task 1b, a regression task, assigns a rating between 1 to 5 to text considered humours, and 0 otherwise. The third Task 1c, itself a binary task, gives a controversy score to a text. The fourth task predicts the general offensiveness of a text on a scale of 0 to 5. Our system's implementation only experiments with three of the Subtasks, including Task 1a, Task 1b, and Task 2a.

4.3 Hyperparameter tuning

Our approach to hyperparameter tuning involves two steps – one manual (implemented during the evaluation phase) , the other using Bayesian optimization (implemented during post-evaluation). In the first step, though, we experiment with a range of hyperparameter values on Task 1a, and the results applied to train our model on the various subtasks. But during the second step, implemented during the post-evaluation phase, we implement hyperparameter sweeps on each task.

In the first step, we manually select from a range of tunable hyperparameters, with batch sizes $\in \{4, 16\}$, learning rates $\in \{2e-5, 4e-5, 1e-4\}$ and we fine-tune for epochs in $\in \{6, 9, 12, 16\}$. The remaining hyperparameters, including dropout rates, and the parameter weights, are based on the default values for RoBERTa model implementation in the HuggingFace transformer library.

Using the results of the first step as our initial defaults for batch size and learning rate, we consider a fine-grained hyperparameter sweep using the Bayesian optimization across the 24 pretrained layers of the RoBERTa$_{LARGE}$ model. We select a range of learning rates between $0 \ \ \& \ \ 1e-3$ for the pretrained layers. We fine-tune for a range of 6 to 40 epochs, applying early stopping and using accuracy as the evaluation metric on the valuation set for Task 1a, and RMSE as the evaluation metric for Task 1b and Task 2a.

Runs on hyperparameter sweeps are taken on all the three Subtasks - Task 1a, 1b, and 2a. We then use the results of learning rates across the pretrained layers, together with the batch size to train our model for the selected number of epochs on each Subtask. Table 5 shows the results of each hyperparameter sweep.

5 Results

In this section we present the results of our SemEval tasks and our analysis for each step.We present, in the first step, our baseline method and results. We then follow up with the results obtained evaluation phase; here, we analyse the impact of the manual sweep and finetuning on the pretrained RoBERTa model on the subtasks. In the last step, and using the results of the codalab scores, we analyse the impact of hyperparameter sweeps on the scores during the post-evaluation phase.

Tasks	Metric	Scores evaluation
Task 1a	F1-Score	0.939
	Accuracy	0.926
Task 1b	RMSE	0.646
Task 2a	RMSE	0.506

Table 2: Official evaluation scores for each task.

Tasks	Metric	Scores post-evaluation
Task 1a	F1-Score	0.957
	Accuracy	0.947
Task 1b	RMSE	0.5802
Task 2a	RMSE	0.469

Table 3: Post-evaluation scores. These were scores generated during the post-evaluation stage after multiple hyperparameter sweeps.

5.1 Baseline

For our baseline method for the regression tasks, we use a very simple linear regression class by scikit-learn. For the classification task, though, we use logistic regression, also by scikit-learn, but with binarized ngram counts, a method proposed by Wang and Manning (2012). The baseline result for the classification task is 89%. RMSE baseline results for the regression tasks, for both Task 1b and 2a, are 0.54 and 0.74 respectively.

5.2 Official Evaluation

As the evaluation results in Table 2 shows, the F1-Score of 0.939 for **Task 1a** is very high, and that is even for manual values of learning rate, epoch and and batch size. What this shows is that using the recommended learning rate and batch size to fine-tune pretrained RoBERTa model on humor classification tasks can achieve very high results. On the the hands, the same hyperparameter values achieved average RMSE metric score for **Task 1b** and average F1 score for **Task 2a**, which suggest that our approach of using the same hyperparameter values for all subtasks is not working.

5.3 Post Evaluation

During the post-evaluation phase the team carried out an extensive hyperparameter finetuning with the bayesian optimization. Table 3 also shows substantial gain in the F1-score, 0.95, for **Task 1a**, RMSE scores (0.5802 and 0.469) for **Task 1b** and **Task 2a** respectively, after applying the results of the Bayes-optimized hyperparameter sweep in Figure 5 during the second step in the post-evaluation stage.

6 Error Analysis

In an attempt to measure our model's predictions against the annotations by humans, we calculate the confusion matrix, comparing the predicted results with the values in the gold test for Subtask 1a. Figure 1, the confusion matrix during the evaluation phase, shows comparable results for false positives and false negatives. In Figure 2, however, the number of false positives (34) are almost twice the number of false negatives, which is so because the train set has more label 1 data (4932) than label 0 data (3068), a slight difference that can lead to the false positives. Again, the total number of true positives (596), almost twice the number of true negatives (351), shows the model is biazed towards positive labels, which will make it difficult to generalize.

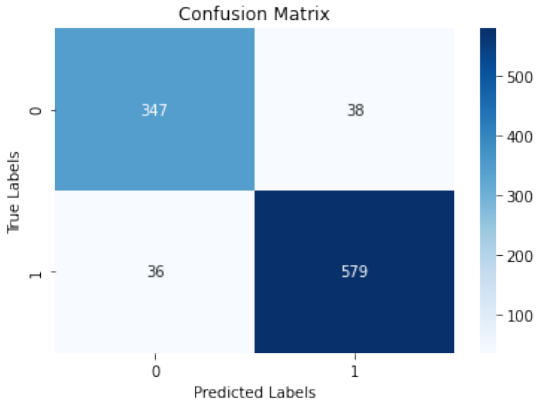

Figure 1: Confusion matrix for Subtask 1a during evaluation phase

Moreover, as shown in Table 4, about half of the number of the false positive predictions are also offensive, in part because most of the labeled texts in the train set that are labeled as humorous are also labeled as offensive.

The higher number of accurate predictions for both the evaluation phase(926) and post-evaluation phase(947) shows our model is efficient in detecting humor.

However, the RMSE scores for humor rating – that is Task 1b, including even the improved RMSE score of 0.58 during the post evaluation phase – still lags behind the RMSE results for Task 2a , the offense-rating subtask. And what this suggest is that our model performs better with offense rating than with humor rating. And that might suggest that the higher scores associated with Task 1a are based on the model's ability to detect offense, which explains why most of the false positive texts also contain offensive content.

ID	text	is humor	offense
9200	Black trans-masculine barbers in NY where ya at? We tryna sumn.	0	0.55
9307	What 's a Black Competitive Swim Skills Clinic? ' feeling motivated	0	1.55
9756	Being told that because I'm a Reform Jew that I'm not actually a Jew and that my smicha as a reform rabbi isn't recognized is a pretty shitty thing to hear from someone.	0	1

Table 4: Examples of gold test results wrongly labeled by our model

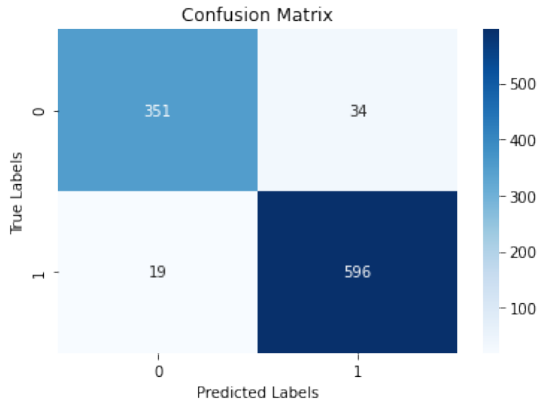

Figure 2: Confusion matrix for Subtask 1a during post-evaluation phase

7 Discussions and Future Works

One major limitation of our approach was that the hyperparameter runs, during the evaluation phase, were experimented only on Task 1a, a binary classification task, and the results applied on the two regression tasks to train our model, which may explain the subpar results for the Task 1b and Task 2a. However, the steps taken during the post-evaluation phase, by independently running the sweeps on each Subtask, showed substantial increase in performance.

In addition, the RoBERTa model, as implemented by HuggingFace, is used as is, without any modification to either the classification layer on top of the RoBERTa model or any of the pretrained layers. In the future, it will be worth pursuing how modifying either of the layers will impact on humor and offense detection.

Overall, however, our system shows that getting optimal values for learning rate, batch and epoch size can yield higher performance for humor detection.

8 Ethical Considerations

The training of RoBERTa, along with other language models such as BERT and its variants, has been shown to be costly, both for its effect on the environment and finance (Strubell et al., 2019). Again, the embeddings used for these language models tend to amplify racial, sexist, and homophobic biases. Mindful of these tendencies, our model experiments included steps to minimize bias and reduce energy cost.

What SemEval-2021 Task 7 (Meaney et al., 2021) intends to achieve is not only to rank humor but also to rate humor offensiveness, the first of any SemEval task. To achieve this, the dataset contains as much as humor as hate, which covers racial slurs, gender bias, trans/homophobic comments, etc. Knowledge of what ranks as offensive in humorous text can help our system moderate humorous content.

To ensure that dataset used for training and development do not over-represent hegemonic viewpoints, Meaney et al. (2021), organizers for the SemEval-2021 Task 7, employed annotators from disparate backgrounds, in age, gender, political views, to ensure that humor ratings and rankings, a subjective process, reflected the varied viewpoints.

Annotators were limited to English speakers, however, which implies that the system's ability to detect and identify humor is largely reflect views inherent in the English language.

Training and testing were carried out on 1 V100 GPUs with less than 16GB of memory, a step taken to ensure minimal, if any, impact on the environment.

The model, however, is prone to classifying offensive content as humorous, which may suggest that applications based on our model will be more likely to rate as humorous any content that might be deemed offensive.

References

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13(null):281–305.

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ian Dewancker, Michael McCourt, and Scott Clark. 2016. Bayesian optimization for machine learning : A practical guidebook. *CoRR*, abs/1612.04858.

Nabil Hossain, John Krumm, Michael Gamon, and Henry Kautz. 2020. SemEval-2020 task 7: Assessing humor in edited news headlines. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 746–758, Barcelona (online). International Committee for Computational Linguistics.

F. Hutter, H. H. Hoos, and K. Leyton-Brown. 2011. Sequential model-based optimization for general algorithm configuration. In *Proc. of LION-5*, page 507–523.

Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. 2013. An evaluation of sequential model-based optimization for expensive blackbox functions. In *Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '13 Companion, page 1209–1216, New York, NY, USA. Association for Computing Machinery.

Hairong Liu, Mingbo Ma, Liang Huang, Hao Xiong, and Zhongjun He. 2019a. Robust neural machine translation with joint textual and phonetic embedding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3044–3049, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *Computing Research Repository*, abs/1907.11692.

J.A. Meaney, Steven R. Wilson, Luis Chiruzzo, Adam Lopez, and Walid Magdy. 2021. Semeval 2021 task 7, hahackathon, detecting and rating humor and offense. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Terufumi Morishita, Gaku Morio, Shota Horiguchi, Hiroaki Ozaki, and Toshinori Miyoshi. 2020. Hitachi at SemEval-2020 task 8: Simple but effective modality ensemble for meme emotion recognition. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1126–1134, Barcelona (online). International Committee for Computational Linguistics.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. SemEval-2017 task 6: #HashtagWars: Learning a sense of humor. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 49–57, Vancouver, Canada. Association for Computational Linguistics.

T. C. Rajapakse. 2019. Simple transformers. `https://github.com/ThilinaRajapakse/simpletransformers`.

Alon Rozental and Dadi Biton. 2019. Amobee at semeval-2019 tasks 5 and 6: Multiple choice CNN over contextual embedding. *CoRR*, abs/1904.08292.

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.

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. SemEval-2018 task 3: Irony detection in English tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Sida Wang and Christopher Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94, Jeju Island, Korea. Association for Computational Linguistics.

Orion Weller and Kevin D. Seppi. 2019. Humor detection: A transformer gets the last laugh. *CoRR*, abs/1909.00252.

Gregor Wiedemann, Seid Muhie Yimam, and Chris Biemann. 2020. Uhh-lt at semeval-2020 task 12: Fine-tuning of pre-trained transformer networks for offensive language detection.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *CoRR*, abs/1903.08983.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 task 12: Multilingual offensive language identification in social media (OffensEval 2020). In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1425–1447, Barcelona (online). International Committee for Computational Linguistics.

A Appendix

The Bayesian optimization is used in hyperparameter optimization techniques for getting a combination of hyperparameters that returns the best performance as measured by a validation set. More formally, for an expensive function $f : \mathcal{X} \rightarrow \mathbb{R}$, hyperparameter optimization can be represented below:

$$\mathbf{x}_{opt} = \underset{\mathbf{x}\in\mathcal{X}}{arg\,max} f(\mathbf{x})$$

The objective function, $f(x)$, represents the score to maximize – in our case, say, the accuracy score – over the validation set; $\mathbf{x}_{opt}$ is the set of hyperparameters that will yield the highest value value of the score – or, in the the case of error rate or RMSE, the lowest value.

The Bayes optimization (or search) is one of the standard algorithms for hyperparameter sweeps; the others are grid and random search (Bergstra and Bengio, 2012). Grid search creates a grid of hyper-parameters and runs through all the range of hyperparameter values. Whilst the grid search is better than manual selection, it is computationally expensive; it is also inefficient because the choice of the next hyper-parameter values in a cycle run is not informed by previous values.

The Bayes search, on the other hand, keeps record of previous results, and uses the results to build a probabilistic model for mapping hyperparameter values to a probability of score on the objective function (Dewancker et al., 2016).

Our system implements the Sequential Model-Based Optimization (SMBO) method, which is a succinct formalization of Bayesian Optimization (Hutter et al., 2011, 2013). Sequential Model-Based Optimization technique iterates between fitting a model and then using that model to determine the next location to evaluate. The pseudocode in Algorithm 1, adopted from SigOpt[8] (Dewancker et al., 2016), encapsulates the technique.

Algorithm 1 Sequential Model-Based Optimization

Input: $f, \mathcal{S}, \mathcal{X}, \mathcal{M}$
$\mathcal{D} \leftarrow$ InitSamples(f, $\mathcal{X}$)
for $i \leftarrow \mathcal{D}$ to T **do**
  $p\left(y \mid \mathbf{x}, \mathcal{D}\right) \leftarrow$ FitModel($\mathcal{M}, \mathcal{D}$)
  $\mathbf{x}_i \leftarrow \underset{\mathbf{x}\in\mathcal{X}}{arg\,max}\, \mathcal{S}(\mathbf{x}, p\left(y \mid \mathbf{x}, \mathcal{D}\right))$
  $\mathbf{y}_i \leftarrow f(\mathbf{x}_i)$
  $\mathcal{D} \leftarrow \mathcal{D} \cup (\mathbf{x}_i, \mathbf{y}_i)$
end for

From a domain $\mathcal{X}$ of tunable hyperparameters, the system would select a small set to initialize a probabilistic regression model $\mathcal{M}$. Afterwards, subsequent locations are selected sequentially from the domain by optimizing the acquisition function $\mathcal{S}$, which uses the current model – the Gaussian process, in our implementation – as a surrogate of the objective function f. Each suggested result is then used to produce the real result $\mathbf{y}_i = f(\mathbf{x}_i)$, the values of which are appended to $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots (\mathbf{x}_i, \mathbf{y}_i)\}$; the updated values are then used by the regression during the next iteration.

[8] https://sigopt.com

hyperparams		Task 1a
lr	$layer_{0-5}$	$1.556e^{-5}$
	$layer_{6-11}$	$4.38e^{-6}$
	$layer_{12-15}$	$4.62e^{-5}$
	$layer_{16-23}$	$3.24e^{-5}$
epochs		9
lr		$4.5e^{-5}$

(a) Task 1a

hyperparams		Task 1b
lr	$layer_{0-5}$	$3.55e^{-5}$
	$layer_{6-11}$	$9.38e^{-6}$
	$layer_{12-15}$	$2.76e^{-5}$
	$layer_{16-23}$	$2.79e^{-5}$
epochs		6
lr		$2.1e^{-5}$

(b) Task 1b

hyperparams		Task 2a
lr	$layer_{0-5}$	$7.86e^{-6}$
	$layer_{6-11}$	$1.02e^{-5}$
	$layer_{12-15}$	$1.38e^{-5}$
	$layer_{16-23}$	$2.08e^{-5}$
epochs		18
lr		$3.74e^{-5}$

Table 5: Hyperparameter Sweep Results for learning rates for the 24 layers of RoBERTa$_{LARGE}$ model, learning rate, epoch and batch sizes of the model.