Post-Hoc Interpretation of Transformer Hyperparameters with Explainable Boosting Machines

Kiron Deb* **Xuan Zhang*** **Kevin Duh**
Johns Hopkins University
{kdeb1, xuanzhang}@jhu.edu, kevinduh@cs.jhu.edu

Abstract

Hyperparameter tuning is important for achieving high accuracy in deep learning models, yet little interpretability work has focused on hyperparameters. We propose to use the Explainable Boosting Machine (EBM), a glassbox method, as a post-hoc analysis tool for understanding how hyperparameters influence model accuracy. We present a case study on Transformer models in machine translation to illustrate the kinds of insights that may be gleaned, and perform extensive analysis to test the robustness of EBM under different data conditions.

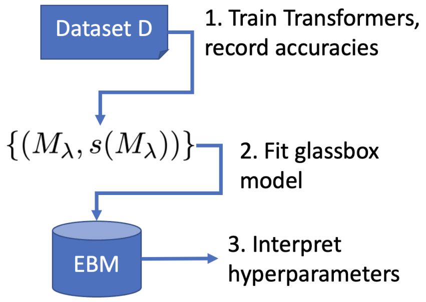
Dataset D
1. Train Transformers, record accuracies
$\{(M_\lambda, s(M_\lambda))\}$
2. Fit glassbox model
EBM
3. Interpret hyperparameters

Figure 1: Proposed framework for post-hoc intepretation of hyperparameters with EBM.

1 Introduction

Deep neural networks have revolutionized the field of AI, bringing about impressive improvements in accuracy at various tasks. There is now a growing interest in interpreting what the model is doing that leads to these high accuracies (Bastings et al., 2021). A better understanding is useful in many ways: it can provide researchers a more in-depth view of the problem, assist developers to debug the model, or give users a way to act on the model result.

Our goal is to improve our understanding of neural network *hyperparameters*. While there are many research efforts on explaining a model's prediction or interpreting a model's parameters, there has been little work on hyperparameters. Hyperparameters like number of layers and learning rate are important factors that impact model performance. In practice, many engineering hours are spent on tuning hyperparameters. We believe methods and tools for interpreting hyperparameters are needed to help practitioners tune more effectively; there are also applications in the growing field of AutoML (Hutter et al., 2019), where our understanding of hyperparameters can help guide researchers design more effective search spaces.

In this paper, we advocate a *post-hoc interpretation framework* for hyperparameters. This framework requires that a set of neural network models with different hyperparameters are trained, and that their resulting accuracy metrics are recorded. Then, a glassbox model is fit on this data to reveal trends in hyperparameters. We use Explainable Boosting Machines (EBM, Lou et al. (2013)) as the glassbox model; it is a Generalized Additive Model similar to Boosted Trees, except that its additive feature function is visualizable in 1-D or 2-D plots, making it well-suited for understanding hyperparameters.

In the following, we first develop further the idea of post-hoc interpretation of hyperparameters and contrast it with other types of interpretability research (Section 3). Then we briefly describe the EBM, which is the glassbox model used in our interpretability framework (Section 2). Section 4 present a case study on Transformers in machine translation tasks, to illustrate how our framework can be used to understand which hyperparameters are important, how its influence changes according to different hyperparameter values, and whether pairwise interactions are present. Finally, Section 5 analyzes the robustness of EBM: it helps characterize under what conditions are the interpretability results valid.

The contribution of this paper is two-fold: First,

* These authors contributed equally to this work.

December 8, 2022 ©2022 Association for Computational Linguistics

we advocate a framework for understanding hyperparameters with EBMs and present a case study on machine translation transformers to illustrate its usefulness. Second, we perform extensive experiments on EBMs to characterize the conditions where interpretability results are robust.

2 Explainable Boosting Machine

Let's define input x as a feature vector representing the hyperparameter setting model M_λ, and y as the scalar output response variable $s(M_\lambda)$. We use EBM as introduced in (Lou et al., 2012, 2013; Caruana et al., 2015) and implemented in Nori et al. (2019). EBM is a generalized additive model with the form:

$$g(y) = \beta_0 + \sum_j f_j(x_j) + \sum_{ij} f_{ij}(x_i, x_j), \quad (1)$$

where g is a link function that transforms the model to either a regression or classification setting (identity or logit, respectively). f_j is a feature function for feature x_j that is learnt through bagging and gradient boosting. Each f_j is trained separately at a time in round-robin fashion. Additionally, EBM also includes pairwise terms f_{ij} to increase accuracy and enable analysis of pairwise interactions between features. In our experiments, we focus on 6 Transformer hyperparameters, so x is a vector of dimension 6 and the EBM model $F(\cdot)$ is a sum of 6 single-hyperparameter functions f_j, up to $(6 \times 5)/2 = 15$ pairwise functions f_{ij}, and a bias term β_0.

An attractive aspect of EBM is that $f_j(x_j)$ is based on a single feature, and can be of arbitrary shape. See examples of f_j in Figure 3: on the left, we see that $f_{j=1}(x_1)$ decreases in score as the learning rate hyperparameter increases; on the right, we see a different $f_{j=2}(x_2)$ increase in score slightly as BPE hyperparameter from 10k to 30k, then drop sharply when BPE increases to 50k. Since the $f_j(\cdot)$ are summed *linearly* to predict the response variable (accuracy or BLEU score), we can obtain an intuitive understanding of how each hyperparameter impacts the final accuracy. In other words, since EBM is an additive model, it is straightforward to infer the contribution of each feature function; at the same time, the ability to learn arbitrary shapes for the feature function allows for enhanced interpretability. Refer to the aforementioned papers for details of how the EBM is trained.

3 Interpreting Hyperparameters

Proposed framework: Our goal is to gain insights about hyperparameters for a class of deep neural networks. We require the existence of a set of models with different hyperparameter settings trained on the same dataset. For example, assume a set of Transformer (Vaswani et al., 2017) models $\{M_\lambda\}, \lambda \in \Lambda$ where Λ represents the hyperparameter space, M_λ represents a model with a specific hyperparameter setting (e.g. 6-layer encoder, 2-layer decoder, 8 heads, 256 word embedding size); each model has an accuracy metric $s(M_\lambda)$, and a glassbox model is fit on pairs $P \triangleq \{(M_\lambda, s(M_\lambda))\}$. Assume there is a person building the models (model builder) and a person analyzing the models after the fact (model analyzer); they may or may not be the same person. Our framework consists of three steps:

1. On a dataset D, the model builder trains N models $\{M_\lambda\}$ and record their accuracy metric $s(M_\lambda)$. The metric can be any scalar in $\mathbb{R}$; for this paper, we focus on machine translation and use the development set BLEU score.

2. The model analyzer fits an EBM on $P \triangleq \{(M_\lambda, s(M_\lambda))\}$. The EBM is a function $F(\cdot)$ that maps from hyperparameter space to BLEU score, $F : \Lambda \rightarrow \mathbb{R}$. In practice, a small subset of P is held-out to measure EBM's generalization, and we would proceed only if we trust that the EBM has not over-fit or under-fit.

3. The model analyzer visualizes the internal features of EBM to glean insights about hyperparameters.

The overall framework is shown in Figure 1. Step 1 is critical because it provides the data for EBM fitting. How large must N be, and are there requirements for the samples from Λ to be independent, identically distributed (i.i.d.)? Neural models can be expensive to train, so we assume that Step 1 is the result of whatever hyperparameter search was performed by the model builder. Thus, the model analyzer may not have full control over the models available for analysis. Section 5 characterizes under what conditions is EBM robust over different sizes and distributions of P.

Step 2 is the core component of our framework. Different glassbox regression models are possible,

Type	Goal	Example Result
Prescriptive	Model building	Given past experience, we recommend setting embedding size to 256 and attention head to 8 on dataset D.
Descriptive (this work)	Post-hoc understanding	Given N models that are trained on dataset D, we find that embedding size influences BLEU more than attention heads.

Table 1: Two kinds of goals for Interpretability Research on hyperparameters.

but we choose EBM due to its excellent visualization ability. Note that while there is a considerable amount of work on interpreting a Transformer's parameters such as attention weights (Kobayashi et al., 2020; Abnar and Zuidema, 2020; Tay et al., 2021; Lim et al., 2018), these methods are not readily applicable due to the non-differentiability and heterogeneity of hyperparameters. Thus, an external model $F : \Lambda \to \mathbb{R}$ that treats hyperparameters as input features is more amenable. This external model is essentially finding hyperparameter "features" that are predictive of accuracy. As long as this model is glassbox in the sense that it's internals are viewable, then we are able to interpret the results in Step 3.

Broader context: We would like to provide context on what our framework does and does not do. In the Explainable AI literature, one way to characterize explainablity/interpretability research is to ask where the method sits on the local vs. global and self-explaining vs post-hoc continuum (Danilevsky et al., 2020). Local methods explain the model's behavior on a specific input, whereas global methods inspects the model generally. Our framework is global in the sense that it identifies hyperparameter trends based on accuracy on a batch of inputs. Self-explaining methods generate explanations as part of the model's prediction process, whereas post-hoc method builds an external model after the predictions have been made. Our framework sits squarely in the post-hoc camp because we work on top of trained Transformers, but it is interesting to note that the glassbox EBM employed can be called a self-explaining method.

In terms of research on hyperparameters, there is a branch of work (Bahar et al., 2017; Britz et al., 2017; Araabi and Monz, 2020) aiming at finding the optimal choices of hyperparameter values. In those work, hyperparameters are usually manually tuned based on experience and massive experiments are conducted to gather results. Those work would make recommendations on which hyperparameter combinations to use in general. We call this approach *prescriptive*; they are useful to inform the building of specific models.

In contrast, our framework is *descriptive*: models have already been trained, and we are interested in understanding the relationship between hyperparameters and accuracy. In other words, rather than predicting whether to set embedding size to 256 or 512, we are more interested in seeing how accuracy changes according to various embedding sizes and understanding whether other hyperparameters like number of layers would interact. This is an example of post-hoc analysis, which is also used in medicine (TDI, 2022; Srinivas et al., 2015) – after the effectiveness of a new treatment is tested, post-hoc analysis on both the failed and successful trials are conducted. It is not the intent of the original study, but it is the support for further trials. The distinctions between the two kinds of interpretation work are summarized in Table 1.

Post-hoc interpretation on hyperparameters is well-suit to the following two scenario: (a) Suppose a practitioner has already performed extensive hyperparameter tuning, and has deployed the best model. It would be a waste to throw away all the data pairs P. Running post-hoc interpretation allows us to extract more knowledge out of the data. Knowledge about which hyperparameters are important, for example, may inform future hyperparameter tuning experiments; it may also assist AutoML researchers to design more efficient search spaces for hyperparameter optimization and neural architecture search. (b) Suppose a researcher proposes a new neural network model. Providing a post-hoc analysis of hyperparameters is akin to showing feature ablation experiments. In sum, our work can be considered as an effort to unpack “blackbox" deep learning models at the level of hyperparameters.

4 Case Study: Post-hoc Interpretation

We now provide a case study on Transformer hyperparameters for machine translation to illustrate the kinds of insight we can learn from the proposed

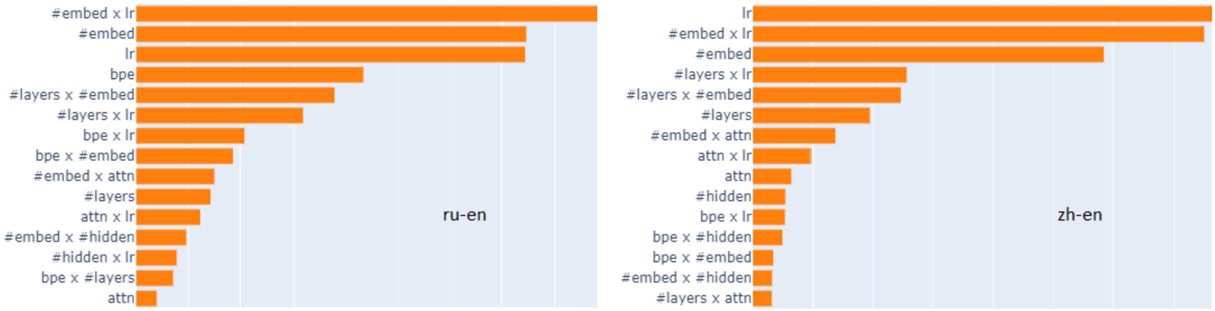

Figure 2: Hyperparameter contribution rank on ru-en (left) and zh-en (right). Hyperparameters are ordered by importance score – for ru-en, *#embed x lr* is the most important, while *attn* is the least important. Hyperparameters that are not included in the plots are in lower ranks than shown ones.

post-hoc interpretation framework.

4.1 Dataset and Setup

Machine Translation (MT) Datasets Our experiments are conducted on a tabular dataset published by Zhang and Duh (2020), which contains 1,983 pairs of hyperparameter configurations and BLEU scores in total. To obtain those pairs, they trained all the Transformers to convergence on 6 MT corpora. Those 6 MT datasets are distinct on sizes – ranging from 24K training samples to 4M; domains – either in a single domain like TED Talks or in mixed-domain; language pairs – including Chinese-English (zh-en), Russian-English (ru-en), Japanese-English (ja-en), English-Japanese (en-ja), Swahili-English (sw-en) and Somali-English (so-en). The large size of the tabular dataset enables efficient post-hoc investigation. Its diversity also allows further study on the generalization of the observations.

Following Zhang and Duh (2020), we will be focusing on the effect of 6 different hyperparameters.

- **Preprocessing configurations:** number of BPE symbols (bpe).
- **Training settings:** initial learning rate (lr) for the Adam optimizer.
- **Architecture designs:** number of layers (layers), embedding size (#embed), number of hidden units in each layer (#hidden), number of heads in self-attention (attn).

These hyperparameters appear frequently in MT literature as a part of the description of experiment setups. Practitioners aiming at a better model spend a large amount of time tuning them manually. We are interested in examining whether they are really equally important and deserve the efforts. We will answer these questions in the following section.

EBM Setup We adopt the implementation of EBM from Nori et al. (2019). To be specific, we train a EBM regressor on each of the language pair, which results in 6 models. Due to space limitation, we will only show results on selected language pairs, the rest can be found in Appendix.

4.2 Findings

In this section, we show how EBM can be used to interpret Transformer hyperparameters and report three types of findings.

4.2.1 Hyperparameter Importance

EBM learns an importance score for each feature, which indicates how much the model performance would change with varying feature values. It is computed as the absolute expected value of f_j over the dataset. Figure 2 plots the hyperparameter importance ranking on ru-en and zh-en. As shown in the figure, hyperparameters are not equally important and there is a large discrepancy between features. On ru-en, *#embed* and *lr* are the most critical hyperparameters in determining Transformer's performance followed by *bpe*; while adjusting *#layers*, *attn* and *#hidden* (not shown in the figure) would only slightly affect the results. On zh-en, *lr* and *#embed* are also at the top of the listing, but the overall ranking is different from ru-en. Some important hyperparameters for ru-en, e.g. *bpe*, rank low on zh-en. Some insignificant hyperparameters for ru-en, e.g. *#hidden*, are elevated to higher positions on zh-en.

https://github.com/interpretml/interpret

In summary, there are only a limited number of critical hyperparameters for Transformers, and it would be more efficient to focus more on tuning them when developing a model. Across 6 language pairs, *#attn* is always ranked low and can be probably dropped from future hyperparameter search.

4.2.2 Single Hyperparameter Analysis

Besides the macro view of contributions of all the hyperparameters, EBM also provides a micro view studying how the segments within each hyperparameter relate. Figure 3 depicts the single feature function extracted from the trained EBM model on en-ja. As *lr* increases from 0.0003 to 0.001, BLEU score decreases significantly. While it is not the case for *bpe*, where the BLEU score does not change monotonically – it rises a little when *bpe* increases from 10k to 30k, then drops notably when *bpe* becomes 50k. This finding tell us both 10k and 30k are positively correlated with BLEU and the difference is not so distinct, but 50k is definitely not desirable.

4.2.3 Pairwise Interactions

EBM can automatically detect and include pairwise interaction terms in its modeling. Figure 4 shows an example of how two hyperparameters interact to determine Transformer's performance. On en-ja, *#embed* with size of 1024 and *lr* with the size of 0.0003 produce the highest BLEU score among all the combinations. On the contrary, *#embed* 1024 and *lr* 0.001 output the worst Transformer. This is consistent with Figure 3 Left – larger *lr* worsens the performance.

However, this does not hold true for *#embed* 256 and 512: given these values, there is not so strong of a (negative) correlation between *lr* and BLEU score. This seems to imply that while lr is sensitive for a large *#embed* 1024, it is less sensitive when *#embed* is small. We do have to interpret this result carefully because there may be confounding factors from the individual feature functions f_j that are added, but this is illustrative of the potential insights we may gain from this case study.

Theoretically, the EBM formulation can allow for higher-order interactions (e.g. three-way). This may be a promising direction for future work.

5 Analysis of EBM Robustness

To ensure the validity of our post-hoc interpretation framework, we need to analyze the robustness of EBM to different kinds of data sizes and distributions. Specifically, one important requirement for our framework is the availability of $P \triangleq \{(M_\lambda, s(M_\lambda))\}$; one might not be able to fully control how this data is acquired. It may be a by-product of an extensive grid search, a manual and focused hyperparameter tuning guided by an engineer's intuition, or an AutoML experiment. This implies the that hyperparameters may not be sampled uniformly from the space Λ, and the number of samples for EBM fitting may not be very large.

In order to gain better understanding of EBM's robustness under different conditions, we conduct four experiments. We first study how EBM's fitting ability would be affected if the size or the distribution of training data changes. We then make connections to Hyperparameter Optimization (HPO), and examine EBM's performance on data generated by sampling from two different HPO methods. Finally, we investigate the generalization ability of EBM. To be more specific, we test whether a EBM model trained on one dataset can perform well on another dataset.

The experiments following are all conducted on sw-en except for the one in Section 5.4. We split the sw-en dataset, the largest dataset among the six datasets provided by Zhang and Duh (2020) which contains 767 (configuration, BLEU score) pairs, into a train set with 614 samples and a test set with 153 samples. An EBM regressor is trained on a subset of the train set and its performance on the test set is reported. We repeat the process 5 times with different random seeds to generate 5 different train-test splits. Thus, results reported below are all averaged over 5 runs.

5.1 Varying Data Sizes

In practice, it is often infeasible to get a tabular dataset as large as the one in Zhang and Duh (2020), where around 2,000 Transformers are trained. This raises the question how EBM would perform with insufficient training data. In other words, it is in doubt if its interpretations on hyperparameters (e.g. observations shown in Section 4.2) are trustworthy when it is trained with less data.

In order to answer the questions above, we create datasets with different sizes by randomly sampling from the train set of sw-en. We experimented with subsets ranging from containing only 5% of the training samples, that is 31 samples, to the whole

Data here refers to the (hyperparameter configuration, BLEU score) pairs, instead of the sentence pairs that are used to train a MT Transformer.

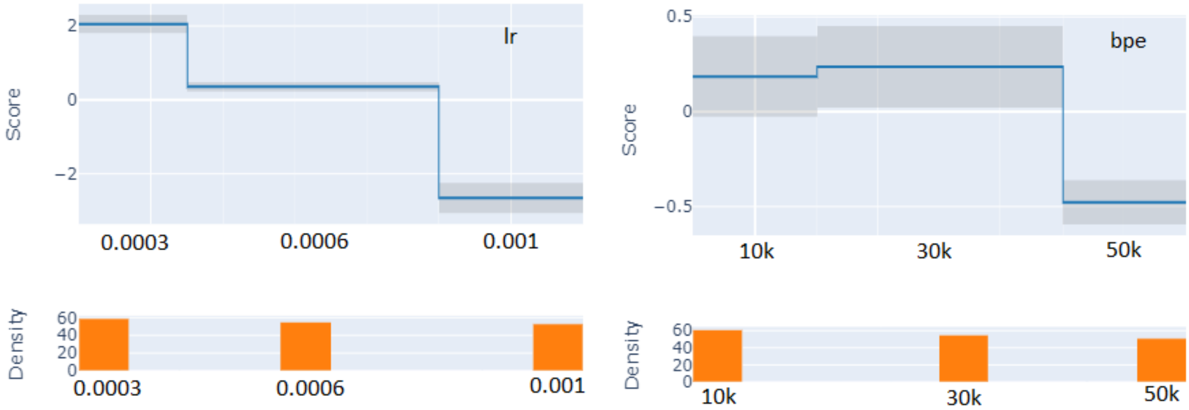

Figure 3: Single hyperparameter feature function on en-ja. Left: initial learning rate. Right: bpe symbols. Higher *score* indicates a higher chance to get a high BLEU score. *Density* refers to the number of samples in the dataset.

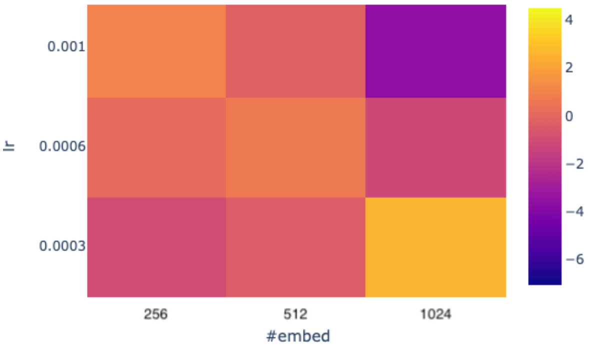

Figure 4: Pairwise interaction between embedding size and initial learning rate on en-ja. Higher *score* (yellow) indicates higher odds to get higher BLEU scores.

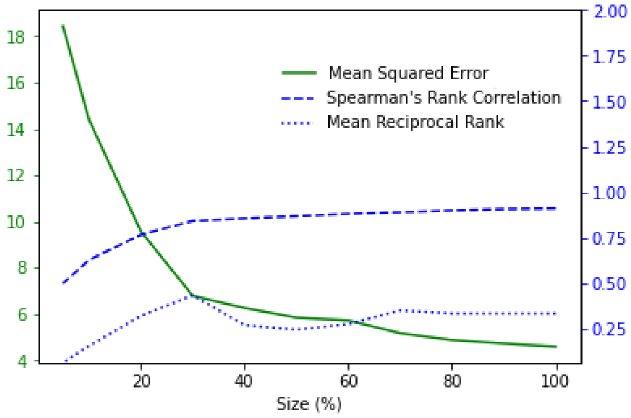

Figure 5: EBM's fitting ability with varying data sizes of sw-en. Subsets are generated by randomly sampling from the train set. Results are averaged over 5 runs with different random seeds.

train set, i.e. 614 samples.

We use the following metrics to measure EBM's performance:

- **Mean Squared Error (MSE)** We calculate the average of the squared difference between the actual BLEU scores and EBM regressor's predictions given hyperparameter configurations. As a widely used measure of an estimator's quality, MSE is useful when compared between estimators. To be more specific, when there are multiple MSE scores, a lower one indicates a stronger estimator. While when there is only a single MSE score, it is hard to judge whether it is low enough to testify a good EBM model. Thus, we propose the following metrics as complements to MSE.

- **Spearman's Rank Correlation Coefficient (SRC)** We measure SRC between the ranking of real BLEU scores and EBM's predictions. For the purpose of interpreting hyperparameters, it is not necessary that EBM would predict the exact BLEU scores. Instead, it is more important that it recovers the ranking. For SRC, higher is better.

- **Mean Reciprocal Rank (MRR)** In some cases, for example, in hyperparameter search, one might be more interested in getting the best configuration and would be less concerned with the ranking of all the configurations. Reciprocal rank is defined as $\frac{1}{rank}$, where $rank$ is the position of the best configuration predicted by EBM in the real ranking. MRR, in our case, is the average over 5 runs. It is better if MRR is closer to 1.

Size(%)	5	10	20	30	40	50	60	70	80	90	100
Mean	18.43	14.40	9.58	6.79	6.26	5.84	5.72	5.17	4.88	4.73	4.59
Std	3.51	1.56	0.84	0.54	0.41	0.57	0.32	0.16	0.31	0.13	0.10

Table 2: The mean and standard deviation of MSE on sw-en test set. EBM is trained on subsets of train set with various sizes and data compositions. Each subset is sampled 5 times with different random seeds.

We plot EBM's performance with varying data sizes in Figure 5. It can be observed that although MSE rises drastically when the data size shrinks from 30% to 5%, it remains roughly at the same level when the size is larger than 30%. This means that a relatively accurate EBM model can be obtained with only 185 samples, and data sizes smaller than that would worsen the model significantly.

Same trend is also shown in other metrics and 30% is the turning point for all the lines. SRC ends up getting close to 1 when the data size increases, suggesting EBM's great ability to recover the ranking. MRR stops at $\frac{1}{3}$, that means EBM mistakes the third best configuration as the best one. However, the difference between the BLEU score of the top three and top one is small, which is only 0.41.

5.2 Varying Data Distributions

Section 5.1 shows that a comparably good EBM model can be obtained by training on as few as 185 samples. Would this stay true if those 185 samples are replaced with other 185 samples? In other words, would EBM be robust to varying data distributions?

We evaluate EBM models trained with different data compositions and data sizes. Results are summarized in Table 2. As the amount of training data increases, the standard deviation of MSE decreases gradually, i.e. the EBM model becomes more robust. When given limited data, EBM is more prone to underfitting and generalize poorly to the test set. It can be inferred that hyperparameter interpretations produced by EBM models trained with more samples are more trustworthy and accurate than those trained with limited data.

5.3 Connections to HPO

The goal of HPO is to find an optimal hyperparameter configuration with as few evaluations of the model as possible. Most of the HPO methods can be classified as sequential model-based optimization (SMBO). SMBO employs 1) a **surrogate model** to approximate the underlying function between hyperparameters and model performance, 2) an **acquisition function** to propose hyperparameter configurations to explore. SMBO works iteratively. Considering applying HPO to MT Transformer, at each iteration, the acquisition function selects configuration candidates to query the Transformer; after training and evaluation, a BLEU score is returned; the surrogate model then fits on the resulting pairs of configurations and BLEU scores and make predictions on unevaluated configurations, which are further used by the acquisition function as bases for making candidate suggestions. Thus, the fitting ability of the surrogate model is crucial to the success of the HPO method.

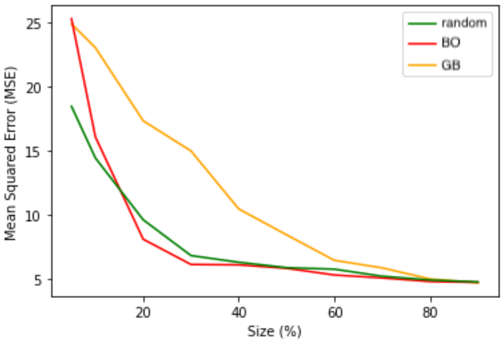

Figure 6: The performance of EBM trained on sw-en data sampled by BO, GB and randomly sampling. EBM is evaluated on a held-out test set, which takes up 20% of the sw-en data.

In this section, we focus on investigating how EBM would fit the sampling by HPO methods, where sampling refers to the candidates proposed by acquisition function along one complete run of HPO – this is related to Section 5.2, since HPO sampling generates another unique data distribution for EBM to train on.

We experiment with two HPO methods, Bayesian Optimization (BO) and a Graph-Based HPO method (GB, Zhang and Duh (2020)). For BO, we use Gaussian Processes as surrogate model and expected improvement as acquisition function.

100% refers to using all the samples in the train set, which takes up 80% of the original sw-en dataset. MSE here is not determined because we also randomly sampled train set from the whole dataset multiple times.

For GB, we use Matérn52 kernel and expected influence. We run BO and GB separately on sw-en and record the sampling order of hyperparameter configurations. We then compare the performance of EBM models trained on the first $n\%$ data points in the sampling with those trained on randomly sampled data. Results are plotted in Figure 6.

BO and *random* show similar trends with MSE falling sharply when the training data increases from 5% to 30%. While GB drops at a slower pace with MSE always staying the highest among the three. The discrepancy between the curves testifies the discrepancy between the sampling of BO and GB. Compared to *random* and BO, the distribution of GB sampling is more skewed. At size 15%, BO surpasses *random* and maintains the lowest MSE till size 100%. This suggests that BO sampling makes EBM a better model than random sampling.

EBM can be used in combination with HPO in two ways: 1) During the run of HPO, EBM can be adopted as an analysis tool. By fitting the HPO sampling, it can provide insights on hyperparameter importance (Section 4.2.1) and make suggestions on hyperparameter values (Section 4.2.2). The HPO algorithm can then adjust its search space accordingly for later runs. But one should be cautious when the HPO algorithm in employment generates poor sampling distribution like GB does. 2) EBM can also be adopted as an alternative surrogate model considering its good fitting ability.

5.4 Transferability

So far, we have examined EBM's behaviours on specific language pairs. We have trained isolated EBM models on 6 MT tasks. Next, we explore whether EBM can leverage knowledge learned from one task and transfer it to another. Specifically, we evaluate each trained model on the test set of each of the language pair respectively. Figure 7 summarizes the results.

EBM faces difficulty on some of the transfers, for example, from sw-en (y-axis) to so-en (x-axis) and from so-en to sw-en. Meanwhile, there are also some successful transfers, for example, from en-ja to ru-en and from ru-en to en-ja. Surprisingly, EBM trained on en-ja generalizes so well on ja-en and ru-en that MSE obtained on those two test sets is even lower than that obtained on en-ja's test set. However, overall, there does not exist a single dataset that can produce a good EBM that can generalize well on all the other datasets.

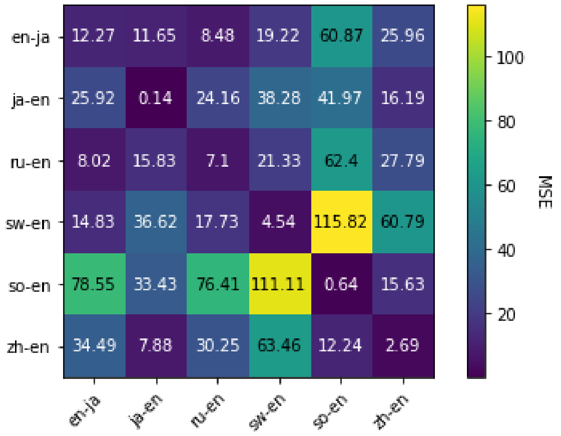

Figure 7: MSE of EBMs trained on one dataset (x-axis) and tested on another (y-axis).

An interesting future work is to implement interpretability tools to analyze when transfer works and when it does not.

6 Related Work

Previous work that explores the effect of choices of hyperparameters can be mainly divided into two categories: the prescriptive approach aims to offer advice on the configurations by large-scale experimental runs and those developing tools to improve the understanding of the hyperparameters, as cited in Section 3. Our work follows the descriptive approach, which seeks to interpret trends from a set of already-trained models. Related are some studies that measure hyperparameter importance: Hutter et al. (2014) and Sharma et al. (2019) applied a functional ANOVA framework to assess the importance, while Probst et al. (2019) adopted a variant term, hyperparameter tunability, conditioned on the difference on the performance of default and optimal settings of hyperparameters.

Exploration of the hyperparameter space also appears in research on HPO interpretability (Pfisterer et al., 2019; Freitas, 2019; Xanthopoulos et al., 2020). Moosbauer et al. (2021) attempted to interpret the HPO process with a variant of the partial dependence plot and showed what the surrogate model learned about the search space and how the final model is found.

7 Conclusions

In this work, we propose a framework for interpreting the hyperparameters of a set of Transformer models. Our framework work uses EBM as a post-hoc analysis tool, and we show that as a glassbox model, EBM is effectively at interpreting hyperparameters. While the computational needs of generating training data for EBM may seem large at first glance, we emphasize that we advocate for *post-hoc* analysis. In other words, the analysis is performed on the results of whatever hyperparameter search the model builder needs to perform to deploy a model.

Our MT case study demonstrates the kinds of insights one can glean regarding the relationship between hyperparameter configurations and Transformer performance; for example, we discover that that not all hyperparameters are equally important, and some hyperparameters exhibit non-monotic corelation with BLEU scores. Further, we conducted a series of analyses to test the robustness of EBM's fitting ability under varying data sizes and distributions. We show that EBM fits well under limited data, yet struggles with transfer across different MT datasets. It should also be noted that the conclusions drawn from MT tasks might not be applicable to other Transformer-based tasks.

Hyperparameter tuning is often viewed as a critical yet un-intuitive part of the model building process. We hope that our proposal provides a first step in unveiling the mysterious masks of hard-to-interpret hyperparameters in deep learning models.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online. Association for Computational Linguistics.

Ali Araabi and Christof Monz. 2020. Optimizing transformer for low-resource neural machine translation. *arXiv preprint arXiv:2011.02266*.

Parnia Bahar, Tamer Alkhouli, Jan-Thorsten Peter, Christopher Jan-Steffen Brix, and Hermann Ney. 2017. Empirical investigation of optimization algorithms in neural machine translation. *The Prague Bulletin of Mathematical Linguistics*, 108(1):13–25.

Jasmijn Bastings, Yonatan Belinkov, Emmanuel Dupoux, Mario Giulianelli, Dieuwke Hupkes, Yuval Pinter, and Hassan Sajjad, editors. 2021. *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, Punta Cana, Dominican Republic.

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. 2017. Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*.

Rich Caruana, Paul Koch, Yin Lou, Marc Sturm , Johannes Gehrke, and Noemie Elhadad. 2015. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *KDD'15, August 10-13, 2015, Sydney, NSW, Australia*. ACM.

Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen. 2020. A survey of the state of explainable AI for natural language processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 447–459, Suzhou, China. Association for Computational Linguistics.

Alex A Freitas. 2019. Automated machine learning for studying the trade-off between predictive accuracy and interpretability. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pages 48–66. Springer.

Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. 2014. An efficient approach for assessing hyperparameter importance. In *International conference on machine learning*, pages 754–762. PMLR.

Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors. 2019. *Automated Machine Learning - Methods, Systems, Challenges*. Springer.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Association for Computational Linguistics.

Robert Lim, Kenneth Heafield, Hieu Hoang, Mark Briers, and Allen Malony. 2018. Exploring hyper-parameter optimization for neural machine translation on gpu architectures. *arXiv preprint arXiv:1805.02094*.

Yin Lou, Rich Caruana, and Johannes Gehrke. 2012. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, page 150–158, New York, NY, USA. Association for Computing Machinery.

Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. 2013. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, page 623–631, New York, NY, USA. Association for Computing Machinery.

Julia Moosbauer, Julia Herbinger, Giuseppe Casalicchio, Marius Lindauer, and Bernd Bischl. 2021. Explaining hyperparameter optimization via partial dependence plots. *Advances in Neural Information Processing Systems*, 34.

Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. 2019. Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*.

Florian Pfisterer, Janek Thomas, and Bernd Bischl. 2019. Towards human centered automl. *arXiv preprint arXiv:1911.02391*.

Philipp Probst, Anne-Laure Boulesteix, and Bernd Bischl. 2019. Tunability: importance of hyperparameters of machine learning algorithms. *The Journal of Machine Learning Research*, 20(1):1934–1965.

Abhinav Sharma, Jan N van Rijn, Frank Hutter, and Andreas Müller. 2019. Hyperparameter importance for image classification by residual neural networks. In *International Conference on Discovery Science*, pages 112–126. Springer.

Titte R Srinivas, Bing Ho, Joseph Kang, and Bruce Kaplan. 2015. Post hoc analyses: After the facts. *Transplantation*, 99.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2021. Synthesizer: Rethinking self-attention for transformer models. In *International conference on machine learning*, pages 10183–10192. PMLR.

ALS TDI. 2022. What is post-hoc analysis in a clinical trial.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Iordanis Xanthopoulos, Ioannis Tsamardinos, Vassilis Christophides, Eric Simon, and Alejandro Salinger. 2020. Putting the human back in the automl loop. In *EDBT/ICDT Workshops*.

Xuan Zhang and Kevin Duh. 2020. Reproducible and efficient benchmarks for hyperparameter optimization of neural machine translation systems. *Transactions of the Association for Computational Linguistics*, 8:393–408.

A Hyperparameter Importance

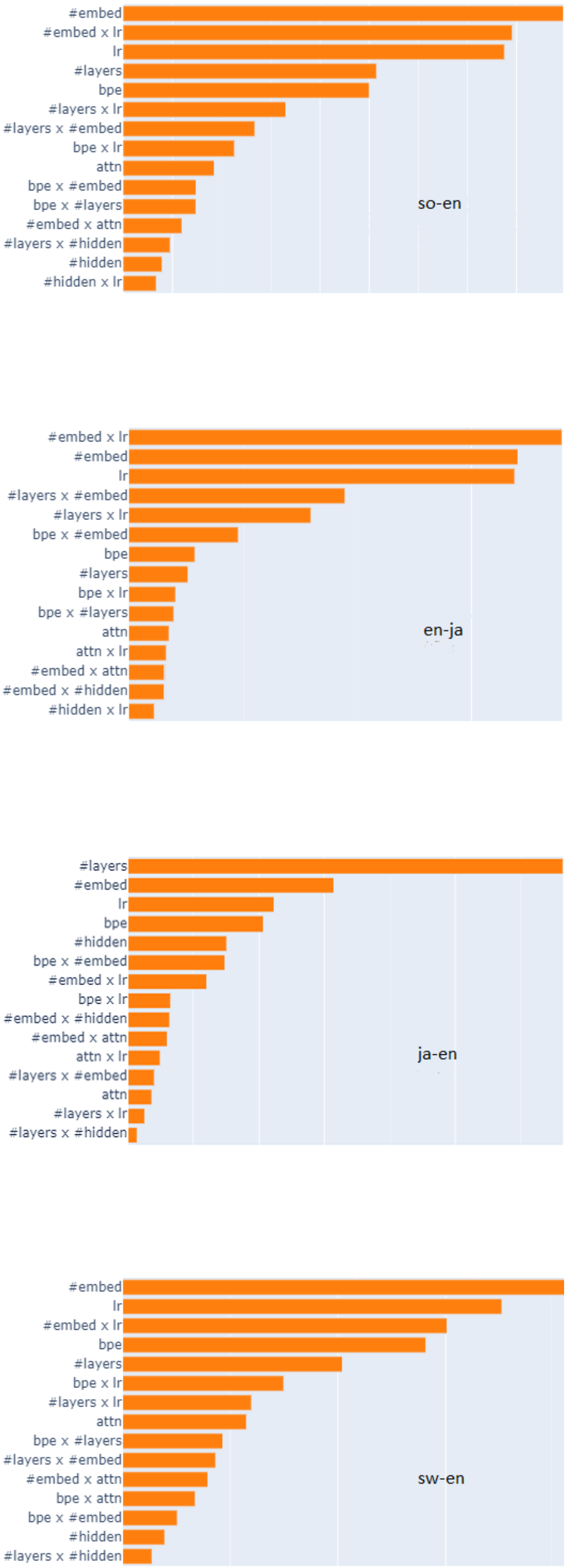

Figure 8: Hyperparameter contribution rank on so-en, en-ja, ja-en and sw-en.

B Single Hyperparameter Analysis

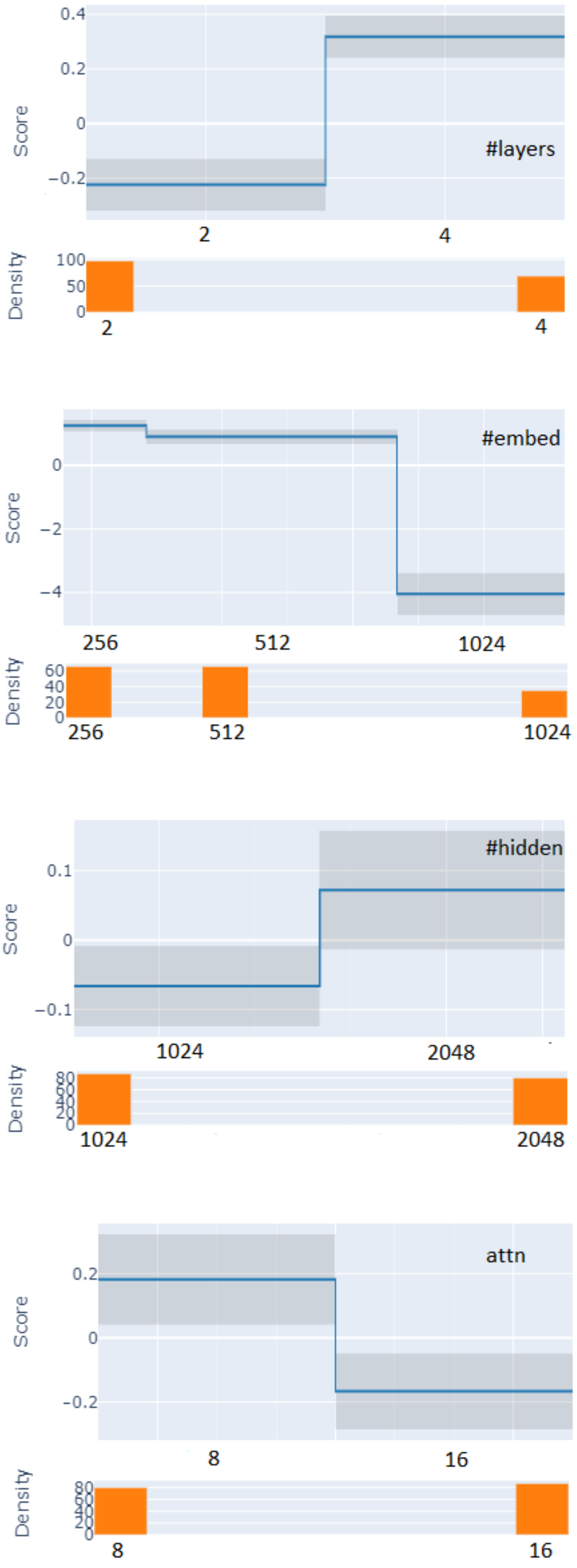

Figure 9: Single hyperparameter feature functions for en-ja. Higher *score* indicates a higher chance to get a high BLEU score. *Density* refers to the number of samples in the dataset.

C Pairwise Interactions

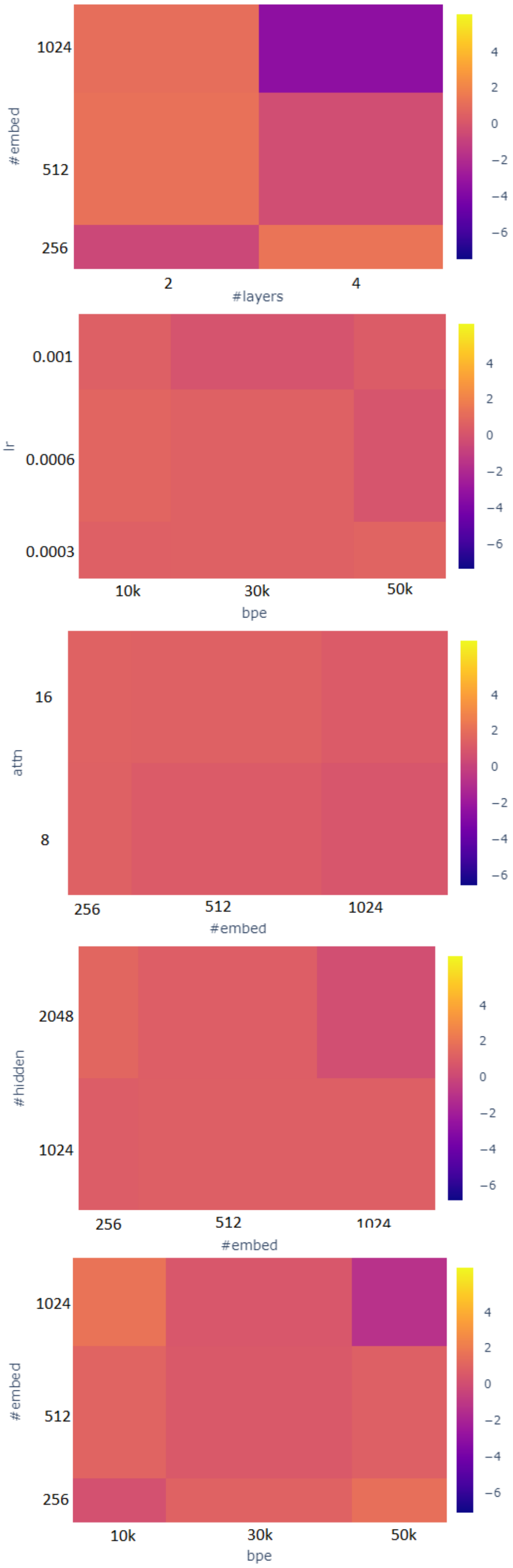

Figure 10: Pairwise interaction between features on en-ja. Higher *score* (yellow) indicates higher odds to get higher BLEU scores.