Bias-Variance Decomposition for Ranking

Pannaga Shivaswamy
Netflix Inc.
Los Gatos, CA, USA
pshivaswamy@netflix.com

Ashok Chandrashekar
Netflix Inc.
Los Gatos, CA, USA
achandrashekar@netflix.com

ABSTRACT

In machine learning and statistics, bias and variance of supervised learning models are well studied concepts. In this work, we try to better understand how these concepts apply in the context of ranking of items (e.g., for web-search or recommender systems). We define notions of bias and variance directly on pairwise ordering of items. We show that ranking disagreements between true orderings and a ranking function can be decomposed into bias and variance components akin to the similar decomposition for the squared loss and other losses that have been previously studied. The popular ranking metric, the area under the curve (AUC), is shown to admit a similar decomposition. We demonstrate the utility of the framework in understanding differences between models. Further, directly looking at the decomposition of the ranking loss rather than the loss used for model fitting can reveal surprising insights. Specifically, we show examples where model training achieves low variance in the traditional sense, yet results in high variance (and high error) on the ranking task.

CCS CONCEPTS

• **Information systems** → **Learning to rank; Recommender systems.**

KEYWORDS

learning to rank; bias-variance; interpretability

ACM Reference Format:

Pannaga Shivaswamy and Ashok Chandrashekar. 2021. Bias-Variance Decomposition for Ranking. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21)*, March 8–12, 2021, Virtual Event, Israel. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3437963.3441772>

1 INTRODUCTION

The goal of many real-world ranking and recommender systems is to rank or recommend candidate items (such as movies, products, URLs etc.). Many such systems are trained using supervised machine learning models [2, 9, 13]. Various types of loss functions along with various architectures are used for training such models which are eventually used for ranking candidates. The loss functions used in the machine learning models are convenient proxies for the eventual objective of ranking.

This work is licensed under a Creative Commons Attribution International 4.0 License.

WSDM '21, March 8–12, 2021, Virtual Event, Israel.

© 2021 Copyright held by the owner/authors(s).

ACM ISBN 978-1-4503-8297-7/21/03.

<https://doi.org/10.1145/3437963.3441772>

Bias and variance of supervised machine learning models are well established concepts [14]. For many loss functions used in machine learning, there exist ways to decompose error into a bias term and a variance term along with additional noise terms. Such decomposition leads to better understanding of the trade-offs involved in training machine learning models. The classical understanding of bias-variance characteristics is that simple models have high bias (due to the lack of ability to fit well to the data) but low variance (because similar models are obtained each time a model is trained from sampled data), whereas more complex models often exhibit low bias but high variance. It is important to note that bias and variance are defined on a distribution of models, not on a single model. Since supervised learning models are trained on a randomly drawn instance of training set, it results in a distribution over the models. If we were to retrain a model with a different draw of the training set, we get another model from the distribution of models. We make similar assumption about model training in this paper as well, but the trained models are assumed to be used in a ranking task.

In this paper, we consider bias and variance directly on pairwise ranking tasks rather than for the surrogate objective or loss function used for training a model used for the task. A model employed for the ranking task might be trained using a pointwise, a pairwise or a listwise or any other objective using surrogate losses. Regardless, we directly look at the bias-variance of the ultimate ranking task at hand. Directly looking at the ranking task can provide insights into the behavior of ranking itself (which is what we primarily care about in a ranking task) rather than the proxy used for learning a model. We illustrate the importance of studying the bias and variance characteristics of the ranking task directly by demonstrating and explaining a seemingly counterintuitive result on a real-world dataset. We show that, even when models have low variance in the traditional sense (i.e., when viewed from the point of view of the surrogate loss that is optimized), the rankings produced can exhibit high variance. Both bias and variance on ranking that we study in this paper have simple, intuitive interpretations. Bias is a measure of distance between the ideal ranking and the rankings produced by a distribution of models. On the other hand, high variance in our framework implies that a model might give one ordering for a pair of examples one time but the opposite ordering another time it is trained.

The main contributions of this paper are as follows. First, we define bias and variance for pairwise ranking. We then show that the ranking error can be decomposed into bias and variance. The framework developed here is general enough that it admits several interesting extensions that are shown in the paper. In particular, we show that the popular ranking metric AUC (Area Under the Curve) also admits a decomposition similar to the one on the pairwise ranking error. Finally, in the experiments section, we show the

bias-variance curves for several popular ranking approaches. In particular, we show that the bias-variance curves for the ranking task, at times, can look very different from bias-variance curves obtained for the proxy loss. We also show an application of using our framework to quantitatively recover the bias of a contrived biased model. Directly looking at the bias-variance of the ultimate objective of ranking is itself a departure from the standard bias-variance analysis practiced in the field. Typically, such analysis is performed on *the same* objective that is also used for optimization.

2 RELATED WORK

Bias-variance decomposition for squared error is a classic result [14], but there are many other variants in the literature. For example, there are bias-variance frameworks for the 0/1 loss in the context of binary and multi-class classifications [18]. There have also been unified approaches for the squared loss and the 0/1 loss [10, 11]. More recently, versions of decomposition for the cross-entropy loss [23, 29] using KL-divergence have been proposed. There is also work on bias-variance in the context of Information Retrieval [8, 30]. Zhang et al. [30] define a notion of bias and variance based on average precision and add them up as a new metric and show its usefulness in practice rather than a decomposition of an original metric. Cormack and Grossman [8] treat rankings as embeddings in a Euclidean space and they define notions of bias and variance based on the embeddings. Bias-variance has been studied in the context of neural networks as well [12]. More recent work [4, 22] looks at the bias-variance trade-offs for deep networks where it has been observed that the traditional understanding does not necessarily hold. These works try to shed more light on how bias and variance change as the architectures are varied in depth or width.

Typically, a model that is employed for ranking items is often trained using a pointwise loss [19] where predictions are at individual query, document level. In contrast, with pairwise approaches [5, 15, 25, 26], a ranking loss is optimized where the model tries to correctly predict ordering among pairs of URLs for given queries. Listwise approaches learn to predict by looking at objectives at the entire list of items rather than at individual items or pairs of items [6, 7]. However, the real objective of all these approaches is to perform well on the ultimate ranking task at hand. Various loss functions are mere approximations that lead to elegant and computationally feasible algorithms for obtaining ranking functions.

In the ranking literature, there are many works about unbiased learning to rank. The main goal in these approaches is to design a learning algorithm that can overcome different types of biases in the data, for example position bias [16, 28], selection bias [27], etc. There is also more recent literature on bias in the societal sense from an algorithmic fairness perspective [3, 21]. The goal of many of these approaches is to overcome some specific notion of bias.

In contrast to all these works, the goal of this paper is not about developing new learning to rank algorithms to overcome specific notions of bias. Rather, the goal of this work is to directly look at pairwise ranking based bias-variance decomposition.

3 PRELIMINARIES

We introduce the following terminology that we use in the rest of this paper. Examples $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ are drawn i.i.d. from

a fixed but unknown distribution. An example of x_i is a feature representation for a query, document pair or a query, URL pair. y_i is a relevance label obtained from implicit or explicit feedback on the document/URL. y_i could be a binary label, or a real valued label or an ordinal ranking. The main requirement on $\mathcal{Y}$ is that we are able to make comparisons between y_i and y_j and determine the order. We are primarily interested in pairs of examples $(x_i, y_i), (x_j, y_j)$ where we can either say $y_i > y_j$ (y_i better compared to y_j) or $y_i \leq y_j$ (y_j at least as good as y_i). We assume that there exists an unknown function $f^* : \mathcal{X} \rightarrow \mathcal{R}$ from which the labels are generated such that $\mathbf{P}[y_i > y_j | f^*, x_i, x_j]$ is independent of everything else. Our formulations assume probabilistic labels. For example, in the context of search/recommendation, the statement $\mathbf{P}[y_i > y_j] = 0.8$ means that the item (e.g. document or URL) i is preferred over item j , eighty percent of the times. Deterministic labels are a specific case, where we can replace $\mathbf{P}[y_i > y_j]$ with $\mathbf{I}[y_i > y_j]$ to get the corresponding results for the deterministic cases.

The ranking error, bias and variance will be defined on a set of pairs indices of examples $S = \{(i, j)\}$. We denote the cardinality of the set S by m , i.e., $|S| = m$. The set S could be the set of all pairs of examples, it could be all pairs of examples for a query, or some other grouping of pairs that can be defined based on the application; the choice of S is specific to the application (we give an example in Section 6.2). For deriving our main results, we work with a generic set S without a specific choice. For brevity, we denote any summation of any $g(\cdot)$ over the elements of S , as an empirical mean as follows:

$$\frac{1}{m} \sum_{(i,j) \in S} g(\cdot) = \hat{\mathbf{E}}_S[g(\cdot)]. \quad (1)$$

Most of the results in this paper could have been derived with just one pair of examples and then extending them to expectation over the set S . However, we find it more informative to directly show empirical expectations over S for clarity.

For any definition of ranking error, bias and variance, it is desirable to have scale and offset invariance. These properties will be satisfied in our definitions of error, bias and variance.

4 BIAS-VARIANCE ON PAIRWISE RANKING

We start by considering an item (i, j) from the set S that denotes a pair of examples (x_i, y_i) and (x_j, y_j) . We consider the indicator function $\mathbf{I}[y_i > y_j | f^*, x_i, x_j]$ that quantifies whether the example i is preferred over example j . We then consider the empirical mean of these indicator functions over the set S to obtain:

$$\begin{aligned} p_S(y) &= \frac{1}{m} \sum_{(i,j) \in S} \mathbf{I}[y_i > y_j | f^*, x_i, x_j] \\ &= \hat{\mathbf{E}}_S[\mathbf{I}[y_i > y_j | f^*, x_i, x_j]]. \end{aligned}$$

The second equation follows the notation introduced in equation (1). Note the conditioning on f^*, x_i and x_j for comparisons $y_i > y_j$. In the rest of the paper, we assume this dependence exists, but do not show this conditioning explicitly for brevity. We subscript by S to explicitly show the dependence on the set. With slight abuse of notation, the y in $p_S(y)$ is to indicate the dependence on the labels.

Similarly, we define the following quantity over a predictive model $f : \mathcal{X} \rightarrow \mathcal{R}$ rather than with true labels:

$$p_S(f) := \hat{\mathbb{E}}_S [\mathbb{I}[f(x_i) > f(x_j)]].$$

We now define bias by taking the difference between $p_S(f)$ (based on a learned function) and $p_S(y)$ (based on true ordering) and taking expectation over the random variables involved:

$$\begin{aligned} bias_S(f, y) &= \mathbb{E}_{f,y} [p_S(f) - p_S(y)] \\ &= \hat{\mathbb{E}}_S [\mathbb{E}_{f,y} [\mathbb{I}[f(x_i) > f(x_j)] - \mathbb{I}[y_i > y_j]]]. \end{aligned} \quad (2)$$

Since $\hat{\mathbb{E}}_S$ simply denotes an average, we took the expectation over f and y inside. When we learn a model from randomly drawn training data, it results in a distribution over the learned function f which is denoted by the subscript f . The expectation over y denotes the expectation over label distribution. We now have:

$$\begin{aligned} bias_S(f, y) &= \hat{\mathbb{E}}_S [\mathbb{E}_{f,y} [\mathbb{I}[f(x_i) > f(x_j)] - \mathbb{I}[y_i > y_j]]] \\ &= \hat{\mathbb{E}}_S [\mathbb{E}_f [\mathbb{I}[f(x_i) > f(x_j)]] - \mathbb{E}_y [\mathbb{I}[y_i > y_j]]] \\ &= \hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)] - \mathbb{P}[y_i > y_j]]. \end{aligned} \quad (3)$$

Since the first term inside $\mathbb{E}_{f,y}$ is independent of y given the distribution over f , we only need the expectation over f . The second term is independent of f since the labels on given instances depend on an unknown function f^* and independent of everything else. On the third line, we used the fact that the expectation of an indicator random variable is the probability of that event.

We say that a training process that returns $f : \mathcal{X} \rightarrow \mathcal{R}$ is unbiased with respect to a set S if $bias_S(f, y) = 0$. We further say that it is **totally unbiased** if we have the following:

$$\mathbb{P}[f(x_i) > f(x_j)] = \mathbb{P}[y_i > y_j] \quad \forall (i, j) \in S. \quad (4)$$

What this means is that, on average, the labeling function gets the ordering for examples indexed by i and j in the same order at the same frequency as true ordering frequency. Note that this is a stricter requirement than merely saying $bias_S(f, y) = 0$.

We define a root mean square style bias which involves sum of square terms rather than the sum terms in the equation (3):

$$\begin{aligned} rbias_S(f, y) &= \sqrt{\hat{\mathbb{E}}_S [(\mathbb{E}_{f,y} [\mathbb{I}[f(x_i) > f(x_j)] - \mathbb{I}[y_i > y_j]])^2]} \\ &= \sqrt{\hat{\mathbb{E}}_S [(\mathbb{P}[f(x_i) > f(x_j)] - \mathbb{P}[y_i > y_j])^2]}. \end{aligned} \quad (5)$$

Next, we define the variance over indicators on pairs of examples and simplify it as follows:

$$\begin{aligned} var_S(f) &= \hat{\mathbb{E}}_S [(\mathbb{E}_f [\mathbb{I}[f(x_i) > f(x_j)] - \mathbb{E}_f [\mathbb{I}[f(x_i) > f(x_j)]]])^2] \\ &= \hat{\mathbb{E}}_S [\mathbb{E}_f [\mathbb{I}[f(x_i) > f(x_j)] - \mathbb{P}[f(x_i) > f(x_j)]]^2] \\ &= \hat{\mathbb{E}}_S [\mathbb{E}_f [\mathbb{I}[f(x_i) > f(x_j)]^2 + \mathbb{P}[f(x_i) > f(x_j)]^2] \\ &\quad - 2\hat{\mathbb{E}}_S [\mathbb{E}_f [\mathbb{I}[f(x_i) > f(x_j)]\mathbb{P}[f(x_i) > f(x_j)]]] \\ &= \hat{\mathbb{E}}_S [\mathbb{E}_f [\mathbb{I}[f(x_i) > f(x_j)] + \mathbb{P}[f(x_i) > f(x_j)]^2] \\ &\quad - 2\hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)]]^2] \\ &= \hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)] - (\mathbb{P}[f(x_i) > f(x_j)])^2] \\ &= \hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)]\mathbb{P}[f(x_i) \leq f(x_j)]] \end{aligned} \quad (6)$$

On line two, we replaced the inner expectation over an indicator with its probability. On line three, we expanded the square term. On the fourth line, we note that the square of an indicator random variable is equal to the indicator random variable itself and also replaced another expectation over indicator with its probability.

The variance in equation (6) is independent of labels but depends on predictions. The variance term quantifies how often x_i is predicted higher compared to x_j times how often the ordering is reversed. If the predicted orders are the same over all models from a distribution, the variance is zero. On the other hand, the highest possible variance is 0.25 when x_i and x_j are predicted one way half the times and the other way the remaining half the times. A high variance thus indicates that a pair of examples could be ordered either way by a random draw of model. A family of models with small variance would give consistent rankings across training, whereas high variance models can differ in their ranking on pairs of examples.

We also analogously define the variance on indicators on true label pairs. It can be simplified using same recipe as above:

$$\begin{aligned} var_S(y) &= \hat{\mathbb{E}}_S [(\mathbb{E}_y [\mathbb{I}[y_i > y_j]] - \mathbb{E}_y [\mathbb{I}[y_i > y_j]])^2] \\ &= \hat{\mathbb{E}}_S [\mathbb{E}_y [\mathbb{I}[y_i > y_j]] - \mathbb{E}[\mathbb{I}[y_i > y_j]]]^2 \\ &= \hat{\mathbb{E}}_S [\mathbb{P}[y_i > y_j]\mathbb{P}[y_i \leq y_j]]. \end{aligned} \quad (7)$$

We now define ranking error that measures the fraction of examples where the orderings given by y_i and y_j and that given by f on x_i and x_j are in the opposite order. That is, the ranking error is the disagreement between f and y on pairs of examples in S :

$$\begin{aligned} errors_S(f, y) &= \hat{\mathbb{E}}_S [\mathbb{P}[(f(x_i) > f(x_j) \text{ and } y_i \leq y_j) \\ &\quad \text{or } (y_i > y_j \text{ and } f(x_i) \leq f(x_j))]]]. \end{aligned} \quad (8)$$

LEMMA 4.1.

$$rbias_S(f, y)^2 + var_S(f) + var_S(y) = errors_S(f, y). \quad (9)$$

PROOF. Consider the sum of the three terms above from equations (5), (6) and (7) to obtain,

$$\begin{aligned} &rbias_S(f, y)^2 + var_S(f) + var_S(y) \\ &= \hat{\mathbb{E}}_S [(\mathbb{P}[f(x_i) > f(x_j)] - \mathbb{P}[y_i > y_j])^2] \\ &\quad + \hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)]\mathbb{P}[f(x_i) \leq f(x_j)] + \mathbb{P}[y_i > y_j]\mathbb{P}[y_i \leq y_j]] \\ &= \hat{\mathbb{E}}_S [(\mathbb{P}[f(x_i) > f(x_j)]^2 + (\mathbb{P}[y_i > y_j])^2 \\ &\quad - 2\hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)]\mathbb{P}[y_i > y_j]] + \hat{\mathbb{E}}_S [\mathbb{P}[y_i > y_j]\mathbb{P}[y_i \leq y_j]] \\ &\quad + \hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)]\mathbb{P}[f(x_i) \leq f(x_j)]] \\ &= \hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)] + \mathbb{P}[y_i > y_j]] \\ &\quad - 2\hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)]\mathbb{P}[y_i > y_j]] \\ &= \hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j)](1 - \mathbb{P}[y_i > y_j]) \\ &\quad + \hat{\mathbb{E}}_S [\mathbb{P}[y_i > y_j](1 - \mathbb{P}[f(x_i) > f(x_j)])] \\ &= \hat{\mathbb{E}}_S [\mathbb{P}[f(x_i) > f(x_j) \text{ and } y_i \leq y_j] \\ &\quad + \hat{\mathbb{E}}_S [\mathbb{P}[y_i > y_j \text{ and } f(x_i) \leq f(x_j)]] \\ &= errors_S(f, y). \end{aligned} \quad (10)$$

On the third line, we did a quadratic expansion. On the next line we regrouped terms and eliminated probabilities that add to one. We

then regrouped terms and finally wrote the resulting expression as probabilities. $\square$

We presented our results so far by assuming stochastic y so far. However, in many practical scenarios, we do not have access to the distribution over labels. We merely observe an instance of the labels that are assumed to be ground truth labels. When we work with fixed labels, $vars(y)$ becomes zero, $vars(f)$ would remain as before since it is independent of the labels. In the case of deterministic labels, the bias term simplifies as follows:

$$rbias_S(f, y) = \sqrt{\hat{E}_S [(P[f(x_i) > f(x_j)] - I[y_i > y_j])^2]} \quad (11)$$

This definition of bias is a simple ℓ_2 distance between the indicator variable of label ordering and the probability with which a distribution over f gets it right. At an individual pair of example (i, j) level, it is also easy to see that the expression inside $\hat{E}_S$ is nothing but the error squared on that pair. However, over the entire set S , these quantities get aggregated to obtain the overall bias $rbias_S(f, y)$.

5 RELATIONSHIP TO AUC

We showed bias-variance decomposition for the ranking error in equation (9) when we had a distribution over f and y . Similarly we show a decomposition on the popular Area Under the Curve (AUC) metric over the distributions in f and y as well. While ranking error is a disagreement metric between f and y , the AUC is defined as the fraction of times labels and predictions agree. From the definitions of the AUC and our ranking error in equation (8) it is straightforward to see that $AUC_S(f, y) = 1 - errors_S(f, y)$. At this point, it seems that the AUC would negatively depend on the variance of f and noise in y since $errors_S(f, y)$ is equal to sum of root mean square bias (squared) plus the variance terms in equation (10). However, we show that the following equality holds for the AUC:

$$AUC_S(f, y) = \hat{E}_S [(P[f(x_i) > f(x_j)] - P[y_i \leq y_j])^2] \quad (12)$$

$$+ \hat{E}_S [P[y_i > y_j]P[y_i \leq y_j] + P[f(x_i) > f(x_j)]P[f(x_i) \leq f(x_j)]]$$

We show the above expression for AUC holds by showing that the above expression for AUC and the equation for $errors_S(f, y)$ from the first line of equation (10) add to one. We consider only one term involving pairs (i, j) and do not show the empirical mean over the set S in this section for brevity. The result still holds for the entire set S as the expression holds for each (i, j) :

$$\begin{aligned} & (P[f(x_i) > f(x_j)] - P[y_i > y_j])^2 + P[y_i > y_j]P[y_i \leq y_j] \\ & + P[f(x_i) > f(x_j)]P[f(x_i) \leq f(x_j)] \\ & + (P[f(x_i) > f(x_j)] - P[y_i \leq y_j])^2 + P[y_i > y_j]P[y_i \leq y_j] \\ & + P[f(x_i) > f(x_j)]P[f(x_i) \leq f(x_j)] \\ & = 2P[f(x_i) > f(x_j)]^2 + P[y_i > y_j]^2 + P[y_i \leq y_j]^2 \\ & \quad - 2P[f(x_i) > f(x_j)]P[y_i > y_j] - 2P[f(x_i) > f(x_j)]P[y_i \leq y_j] \\ & \quad + 2P[y_i > y_j]P[y_i \leq y_j] + 2P[f(x_i) > f(x_j)]P[f(x_i) \leq f(x_j)] \\ & = 2P[f(x_i) > f(x_j)] + P[y_i > y_j]^2 + P[y_i \leq y_j]^2 \\ & \quad - 2P[f(x_i) > f(x_j)] + 2P[y_i > y_j]P[y_i \leq y_j] \\ & = (P[y_i > y_j] + P[y_i \leq y_j])^2 = 1. \end{aligned}$$

We thus have the following expression for AUC from (12):

$$AUC_S(f, y) = align_S(f, y)^2 + vars(f) + vars(y).$$

where, we call the first term as an alignment and it is given by:

$$align_S(f, y)^2 = \hat{E}_S [(P[f(x_i) > f(x_j)] - P[y_i \leq y_j])^2].$$

As an example, when $P[y_i \leq y_j] = 1$, we would like to have $P[f(x_i) > f(x_j)] = 0$ so that the AUC is maximized. The definition of alignment is in contrast to the term $rbias_S(f, y)^2$ in the similar equation for $error(f, y)$ in equation (9). While $rbias_S(f, y)^2$ is a measure of distance between the ground truth and predictions, $align_S(f, y)^2$ is a measure of similarity between labels and predictions. Since this result is complementary to our earlier result on error, we do not explicitly study this equation in the experimental section. However, the above expression can be of independent interest in further understanding the AUC metric in many applications.

6 EXTENSIONS

In this section, we show some extensions of the bias-variance framework for ranking.

6.1 Utilities Beyond Ordering

It is straightforward to extend the bias-variance framework presented above to utilities beyond mere rank ordering. For example, in web-search ranking, consider two pairs of documents with $y_i = 4, y_j = 3$ and another pair with $y_i = 4, y_j = 0$. In both the cases $y_i > y_j$ but the cost of getting the ordering wrong is much higher for the second pair compared to the first. In our framework, we can change the original estimator $\hat{p}$ to the following to capture utilities beyond indicators: $p_S(y) := \hat{E}_S [I[y_i > y_j | f^*, x_i, x_j] U(y_i, y_j)]$. With this change and with a similar change in the definition of $p_S(f)$, all our equations hold good but in a modified form. The error becomes the expected loss in utility due to getting the ordering wrong on a pair of examples. The bias and variance terms would change to a weighted version of bias and variance. These extensions can be obtained by simple modifications in Section 4.

6.2 Top-k Candidate Selection

Sometimes, we care about selecting a few items from a candidate set for recommendation rather than ranking the entire candidate set. In such applications, we care about getting the top k examples right when ordered by scores from a model. In such cases, it would be more appropriate to set S in our analysis to be pairs where one of the items is in top k but the others are not. For example, when $k = 1$, we define $S = \{(i^*, j) | i^* = \arg \max_i y_i, y_j < y_{i^*} \forall j\}$. Similarly, depending on the application, S could be suitably defined for the task at hand.

6.3 Bias with Respect to Label Generation Model

If we have an overall bias for a model f , can we relate it to the bias of the model f^* (from Section 3) that generated the labels y ? For this, we start with the the bias term from equation (5).

$$\begin{aligned}
rbias_S(f, y)^2 &= \hat{E}_S [(P[f(x_i) > f(x_j)] - P[y_i > y_j])^2] \quad (13) \\
&= \hat{E}_S [(P[f(x_i) > f(x_j)] - P[f^*(x_i) > f^*(x_j)]) \\
&\quad + P[f^*(x_i) > f^*(x_j)] - P[y_i > y_j])^2] \\
&= \hat{E}_S [(P[f(x_i) > f(x_j)] - P[f^*(x_i) > f^*(x_j)])^2] \\
&\quad + \hat{E}_S [(P[f^*(x_i) > f^*(x_j)] - P[y_i > y_j])^2] \\
&\quad - 2\hat{E}_S [\delta_{ij}(P[f^*(x_i) > f^*(x_j)] - P[y_i > y_j])].
\end{aligned}$$

In the last expression, we have denoted $(P[f(x_i) > f(x_j)] - P[f^*(x_i) > f^*(x_j)])$ by δ_{ij} for brevity. The other two terms are identified as $rbias_S(f^*, y)^2$ and $rbias_S(f, f^*)^2$. Assuming that the label generation process is totally unbiased as given in equation (4), we get that the last term in equation (13) which is the multiplier of δ_{ij} is zero. Thus, with such an assumption, we have,

$$rbias_S(f, y)^2 = rbias_S(f, f^*)^2 + rbias_S(f^*, y)^2.$$

Unfortunately f^* is an unknown function, thus, $rbias_S(f^*, y)^2$ is the component of the bias that is not even captured by the ideal labeling function f^* . In reality, we only work with observed labels and $rbias_S(f^*, y)^2$ ends up being part of the bias we measure. However, we typically compare two models to look at their bias and $rbias_S(f^*, y)^2$ component is common for both such models and hence they cancel out in any comparison.

7 EXPERIMENTS

We now present our experimental results for the bias-variance framework for pairwise ranking. In our experiments, we report the error rate, its bias $rbias_S(f, y)^2$ and variance $var_S(f)$ components from equation (9). Since we work with actual realizations of labels rather than their distributions, $var_S(y)$ is zero in all our experiments. All our experiments were on publicly available MQ2008 from LETOR 4.0 dataset[24]. This dataset contains query, document pairs that are given graded relevance labels (0,1 or 2). There are more than 15 thousand query-document pairs that are represented by 46 features forming about 800 queries.

7.1 Bias-Variance Curves

In the first experiment, we present results showing several bias-variance curves. Typically simpler models have high bias but low variance in the traditional sense. However, we show a different behavior with respect to ranking. First this may seem counterintuitive, but we give explanations for this phenomenon.

For pointwise models, we minimized squared error on graded relevance labels with a linear model with ℓ_2 norm regularization (with a scalar weight parameter C) on the weight vector. For the pairwise model, we generated pairs of training examples within a query from the dataset. If the graded relevance label was higher for x_i compared to x_j , we said i was preferred over j for the pairwise training model. We then trained a pairwise logistic regression model with ℓ_2 norm regularization with the pairs of examples and the binary labels that were generated. For the listwise approach, we trained the ListNet [7] model that optimizes cross-entropy loss over the entire list. We took the graded relevance labels from the dataset and obtained the ground truth distribution over labels by

exponentiating graded relevance and normalizing at each query level. We then minimized the following loss as outlined in the ListNet paper:

$$-\frac{1}{|Q|} \sum_{q \in Q} \sum_{i=1}^{n_q} p(y_i^q) \log\left(\frac{\exp(w^\top x_i^q + b)}{\sum_{j=1}^{n_q} \exp(w^\top x_j^q + b)}\right) + C\|w\|^2 \quad (14)$$

All optimizations were implemented in tensorflow [1] using Adam optimizer [17] with default settings.

To study bias and variance, we need a distribution over models coming out of training. The LETOR datasets already come with five train/validation and test splits. For each training split, we obtained ten models by bootstrapping training examples. We sampled with replacement from the training set to achieve the required number of training examples for each run. The randomization in sampling and any randomization in the stochastic gradient descent (Adam in tensorflow) would generate a distribution over the trained models.¹ For the ten models from a training split, we computed error, bias and variance on the provided test split in the dataset. We repeated the training on each split and computed the metrics on the held out test set. The final reported metrics are by averaging from the results on all the test splits. With this procedure, we trained a total of 50 supervised learning models for each setting and model variant.

For the pointwise approach, we looked at the bias-variance decomposition for squared loss [14]. However, measuring squared error for pairwise or listwise models is meaningless. Therefore, we took the decomposition for cross-entropy loss [23, 29] but applied it on pairwise logistic loss. Specifically, the following decomposition from Yang et al. [29] shows the average loss on the left hand side and the bias and variance components on the right hand side using KL divergence:

$$\mathbb{E} \left[\sum_{l=0}^1 \pi_0(l) \log(\pi(l)) \right] = D_{KL}(\pi_0 | \bar{\pi}) + D_{KL}(\bar{\pi} | \pi),$$

where $\bar{\pi}(l) = \exp(\mathbb{E}(\log(\pi(l))))$ for $l = 0, 1$ is the exponentiated average log prediction over models. This decomposition was applied on pairs of examples and with logistic loss for the pairwise and the listwise approaches. Apart from these baselines, we also computed the ranking error and its decomposition as proposed in this paper. For the squared (logistic) loss baseline, squared (logistic) loss decomposition is based on the same proxy loss function, whereas the ranking decomposition is directly on the discrete ranking error.

For the pointwise bias-variance decomposition, we computed error, bias and variance for every query, URL pair. For the pairwise decomposition, we generated pairs of examples within each query from the dataset where the graded relevance labels were different. A document was better compared to another for the same query if the relevance label was higher. Since taking pairs of examples from a query can result in a quadratic number of pairs, we first averaged all metrics at the query level and then aggregated them to the dataset level. This way, queries having a large number of documents do not have an unfair representation in the overall metric. Even for

¹We expect stochasticity from the optimizer to be small or close to none when the objective is convex which is the case in many of our experiments.

Figure 1: Bias-variance and error curves for pointwise (top row), pairwise (middle row) and listwise (bottom row) approaches. All the right plots are with ranking decomposition from this paper. For squared loss, the left plot is decomposition of squared error, for pairwise and listwise, the left plots are decomposition of pairwise logistic loss. Note that bias/error have the y-axis on the left whereas variance has its y-axis on the right therefore bias and variance do not *visually* add up to error.

Figure 2: Histogram of bias values for ranking loss that resulted in the bias values for $C = 1$ and $C = 1000$ for the middle row right plot in Figure 1.

the pointwise metrics, we first averaged over a query and then averaged over the dataset to keep the metrics consistent across the methods. We also computed error bars by bootstrapping from the test examples, but the error bars were so small that they were not even visible in most of the plots, we are thus not showing error bars in our results.

The results for the squared loss are shown in the top row of Figure 1. The left plot shows error, bias and variance based on squared loss and the right plot shows for ranking error. As the regularization increases, we see that the squared error bias-variance decomposition behaving as expected. Higher regularization results in simpler models. We thus see variance being lower and bias being higher. However, for the ranking based bias-variance trade-off, initially as we increase regularization, the variance goes down. However, as the regularization increases further, the variance increases further. What this means is that if we used a high regularization model (simple model) for ranking, the rankings would oscillate a lot more than another model with lower variance.

The results for the pairwise models and the listwise models are shown in the second and the third rows of Figure 1 respectively. In these two cases, the traditional decomposition of the loss looks as expected. However, once again the pairwise ranking error and its bias-variance decomposition in the right plots look similar to what we observed for the pointwise loss with much higher variance at high regularization.

Figure 3: Change in error, bias and variance as we change the number of training examples at a fixed regularization level.

Figure 4: Toy example showing wider spread of the learned weight vector with increasing regularization for squared and ranking losses.

The results with ranking error bias-variance decomposition (the right plots) in Figure 1 are in contrast with the traditional views (the left plots) especially with respect to high ranking variance with high regularization. While this seems surprising, this can be explained as follows. As we increase regularization, the learned weight vector becomes smaller due to ℓ_2 norm regularization. At high regularization, the optimizer would try to push most of the weights close to zero. With squared loss optimization, the intercept term would be nearly constant across many trained models, but the weight vectors become close to zero based on the norm, but otherwise could be in a wider range for the discrete ranking task. With pairwise and listwise objective optimization also, predictions on examples become close to one another with larger regularization parameter. However, the ranking error being a discrete measure (0/1), as the predictions become closer, one model can give one ordering on a pair of examples, while another model could give quite the opposite. Recall from the definition of variance ($var_S(f)$ from equation 6) that it is high when models predict in different orderings on pairs of examples.

To further understand why we see the particular behavior of bias for ranking loss (right plot of second row) Figure 1, we looked at

the histograms of bias values aggregated to individual query level for two regularization values ($C = 1$ and $C = 1000$). As can be seen from the figure 1, regularization $C = 1000$ results in lower bias but higher variance. The histogram is shown in Figure 2. At $C = 1$, the ranking model has many instances with small bias, but also has a few instances with very high bias. At higher regularization, as the scores are pushed closer in the model, the model performs slightly worse at the left end of the histogram but is doing better on the right end. The plot shows overall more even distribution in the middle parts hence resulting in lower ranking bias.

Next we looked at the behavior of the proposed ranking loss decomposition as we varied the number of training examples at a fixed level of regularization. We present the results for only one of the approaches but the results looked similar across training approaches. We picked $C = 1e-3$ from the pairwise experiment. We decreased the number of training examples in fractions of original training and obtained error, bias and variance curves. The results are shown in Figure 3. As expected, the variance decreases as the number of examples in the training increases. Similarly, bias increases with more examples.

To illustrate the above phenomenon where more regularized model generates higher variance, we generated a two dimensional toy data with each feature drawn from the standard normal distribution. The labels were generated by multiplying the features with a randomly generated weight vector and by adding a small Gaussian noise with variance 0.1. The initial generated dataset was a pointwise dataset. Pairwise datasets were generated by comparing all 499.5k pairs of examples. We drew 100 such datasets with the model weight vector fixed. We learned pointwise and pairwise models with two features. We normalized each learned vector to unit norm and multiplied by a constant (ranking from such a transformation still remains the same) just for visually seeing the differences between the different learned models. The results can be seen in Figure 4. It can be seen that with small regularization, the learned vectors are closer to one another, but as the regularization becomes higher, we see a wider spread of models that results in higher variance. With higher spread of weight vectors, the rankings from different models would be much more different than with a smaller spread explaining the higher ranking variance.

7.2 Explaining model differences

In this final experiment, we trained two models - one with biased sampling and one with unbiased sampling. The goal is to show a practical application of the decomposition technique and use it to illuminate the differences in the two models. For the biased model, we took all examples with graded relevance labels 1 or 2 (but excluded labels 0). For the unbiased model, we took examples with labels 0, 1 or 2 uniformly at random but *matched* the number of training examples from the biased case. We then generated binary classification labels from graded relevance labels for both biased and unbiased models. The probability of a positive label was $1/(1+2*\text{grade})$ (i.e., 0 when grade=0, 2/3 when grade=1, 4/5 when grade=2). Once such training data/labels were generated, the rest of the experiment setup matched our experiment in the previous

Figure 5: Ranking error decomposition for the biased training (left), and for the unbiased training (right).

Figure 6: Histogram of bias (left plot) and variance (right plot) values aggregated at the query level for $C = 1$ in Figure 5.

section. We trained logistic regression models with both datasets 50 times for each setting as described before. The evaluation was done on the entire test dataset (following the same train/test folds mentioned in the previous section). The test labels were modified to binary using the same procedure that we followed for training in the unbiased case. We then generated pairs of examples within a query with the binary labels providing the true ordering.

The results for this experiment are shown in Figure 5. The left plot shows the bias-variance decomposition for the biased model whereas the right plot shows the results for uniform sampling (unbiased) case. It can be seen that the biased model exhibits higher bias as well as very high variance since it never saw graded relevance 0 that would always be labeled 0 when converted to binary. We show the histogram of the bias and the variance components aggregated at the query level for the two models in Figure 6. We see that on many queries, the unbiased model gets close to zero bias, while the biased model shows far fewer queries at the zero level but many more queries at higher bias values. The variance for the biased model is also quite high indicating difference in ordering on pairs of examples from successive model trainings. This experiment demonstrates how the framework can be practically used to compare models to obtain deeper insights.

8 CONCLUSIONS

In this paper, we developed a bias and variance decomposition technique for characterizing pairwise rankings of the eventual ranking task rather than that of the intermediate surrogate loss used for optimizing a model. We showed insights from real world experiments that would not have been obtained with traditional bias-variance decomposition approaches. In particular, we showed some seemingly counterintuitive results that show that although some models

may exhibit low variance from the point of view of the surrogate loss, when we look at the actual rankings generated, they can have high variance. Further, the decomposition was also extended to the well known AUC metric which gives a complementary view of the results in this paper.

The framework developed in this paper can provide additional insights into models that are deployed in various types of applications. Applying the framework to deeper architectures and tree based learning models such as gradient boosting decision trees remains as future work. There are also tantalizing implications of the perspective developed in this paper on explore/exploit systems commonly employed in search and recommender systems. For instance, even when the model variance (as per classical definition) is estimated to be small, the model might in fact be doing plenty of exploration in practice (due to high ranking variance). Expanding the work in this paper for a deeper treatment of the implications on explore/exploit approaches is an interesting future direction.

Further, this analysis opens up many possible extensions. For example, from the pairwise rankings, we can derive listwise utilities and directly measure the utilities at the list level with different choices of S . Beyond measures derived from pairs, there are several other measures used in ranking and Information Retrieval [20] such as NDCG, MRR, F_1 score etc. Whether it is possible to extend the framework to other such widely used metrics is another possible research direction.

ACKNOWLEDGMENTS

We thank Ajinkya More, Anuj Shah, Chris Steger, Claire Lackner, Ehtsham Elahi, Harald Steck, Justin Basilico and Vijay Bharadwaj for helpful discussions and feedback on this work.

REFERENCES

- [1] Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. <http://tensorflow.org/>. Software available from tensorflow.org.
- [2] Deepak Agarwal, Bee-Chung Chen, Rupesh Gupta, Joshua Hartman, Qi He, Anand Iyer, Sumanth Kolar, Yiming Ma, Pannagadatta Shivawamy, Ajit Singh, and Liang Zhang. 2014. Activity Ranking in LinkedIn Feed. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, New York, USA) (KDD '14). ACM, New York, NY, USA, 1603–1612. <https://doi.org/10.1145/2623330.2623362>
- [3] Ricardo Baeza-Yates. 2018. Bias on the Web. *Commun. ACM* 61, 6 (May 2018), 54–61. <https://doi.org/10.1145/3209581>
- [4] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. 2019. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences* 116 (07 2019), 201903070. <https://doi.org/10.1073/pnas.1903070116>
- [5] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank Using Gradient Descent. In *Proceedings of the 22nd International Conference on Machine Learning* (Bonn, Germany) (ICML '05). Association for Computing Machinery, New York, NY, USA, 89–96. <https://doi.org/10.1145/1102351.1102363>
- [6] Chris J.C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report MSR-TR-2010-82. Microsoft Research. <https://www.microsoft.com/en-us/research/publication/from-ranknet-to-lambdarank-to-lambdamart-an-overview/>
- [7] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning* (Corvallis, Oregon) (ICML '07). ACM, New York, NY, USA, 129–136. <https://doi.org/10.1145/1273496.1273513>
- [8] Gordon V. Cormack and Maura R. Grossman. 2019. Quantifying Bias and Variance of System Rankings. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) (SIGIR '19). Association for Computing Machinery, New York, NY, USA, 1089–1092. <https://doi.org/10.1145/3331184.3331356>
- [9] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems* (Boston, Massachusetts, USA) (RecSys '16). Association for Computing Machinery, New York, NY, USA, 191–198. <https://doi.org/10.1145/2959100.2959190>
- [10] Pedro Domingos. 2000. A Unified Bias-Variance Decomposition and its Applications. In *Proceedings of the 17th International Conference on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 231–238.
- [11] Pedro Domingos. 2000. A Unified Bias-Variance Decomposition for Zero-One and Squared Loss. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*. AAAI Press, 564–569.
- [12] Stuart Geman, Elie Bienenstock, and Rene Doursat. 1992. Neural Networks and the Bias/Variance Dilemma. *Neural Computation* 4, 1 (1992), 1–58. <https://doi.org/10.1162/neco.1992.4.1.1>
- [13] Carlos A. Gomez-Urbe and Neil Hunt. 2015. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Trans. Manage. Inf. Syst.* 6, 4, Article 13 (Dec. 2015), 19 pages. <https://doi.org/10.1145/2843948>
- [14] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer New York Inc., New York, NY, USA.
- [15] Thorsten Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Edmonton, Alberta, Canada) (KDD '02). Association for Computing Machinery, New York, NY, USA, 133–142. <https://doi.org/10.1145/775047.775067>
- [16] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (Cambridge, United Kingdom) (WSDM '17). Association for Computing Machinery, New York, NY, USA, 781–789. <https://doi.org/10.1145/3018661.3018699>
- [17] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]
- [18] Ron Kohavi and David Wolpert. 1996. Bias plus Variance Decomposition for Zero-One Loss Functions. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning* (Bari, Italy) (ICML '96). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 275–283.
- [19] Tie-Yan Liu. 2010. Learning to Rank for Information Retrieval. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Geneva, Switzerland) (SIGIR '10). Association for Computing Machinery, New York, NY, USA, 904. <https://doi.org/10.1145/1835449.1835676>
- [20] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, USA.
- [21] Marco Morik, Ashudeep Singh, Jessica Hong, and Thorsten Joachims. 2020. Controlling Fairness and Bias in Dynamic Learning-to-Rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) (SIGIR '20). Association for Computing Machinery, New York, NY, USA, 429–438. <https://doi.org/10.1145/3397271.3401100>
- [22] Brady Neal, Sarthak Mittal, Aristide Baratin, Vinayak Tania, Matthew Scicluna, Simon Lacoste-Julien, and Ioannis Mitliagkas. 2018. A Modern Take on the Bias-Variance Tradeoff in Neural Networks. arXiv:1810.08591 [cs.LG]
- [23] David Pfau. 2013. A generalized bias-variance decomposition for bregman divergences. http://davidpfau.com/assets/generalized_bvd_proof.pdf
- [24] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 Datasets. *CoRR* abs/1306.2597 (2013). <http://arxiv.org/abs/1306.2597>
- [25] Herbrich Ralf, Thore Graepel, and Klaus Obermayer. 1999. Support vector learning for ordinal regression. In *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, Vol. 1. 97–102 vol.1. <https://doi.org/10.1049/cp:19991091>
- [26] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback (UAI '09). AUAI Press, Arlington, Virginia, USA, 452–461.
- [27] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Pisa, Italy) (SIGIR '16). ACM, New York, NY, USA, 115–124. <https://doi.org/10.1145/2911451.2911537>
- [28] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position Bias Estimation for Unbiased Learning to Rank in Personal Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (Marina Del Rey, CA, USA) (WSDM '18). Association for Computing Machinery, New York, NY, USA, 610–618. <https://doi.org/10.1145/3159652.3159732>
- [29] Zitong Yang, Yaodong Yu, Chong You, Jacob Steinhardt, and Yi Ma. 2020. Rethinking Bias-Variance Trade-off for Generalization of Neural Networks. arXiv:2002.11328 [cs.LG]
- [30] Peng Zhang, Dawei Song, Jun Wang, and Yuehan Hou. 2013. Bias-Variance Decomposition of IR Evaluation. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Dublin, Ireland) (SIGIR '13). Association for Computing Machinery, New York, NY, USA, 1021–1024. <https://doi.org/10.1145/2484028.2484127>