A Cluster-Based Nearest Neighbor Matching Algorithm for Enhanced A/A Validation in Online Experimentation

Yan He
Yahoo Inc.
Sunnyvale, CA, USA
yhebfc@gmail.com

Lin Yu
Yahoo Inc.
Sunnyvale, CA, USA
lyu9@ncsu.edu

Miao Chen
Yahoo Inc.
Sunnyvale, CA, USA
glycob@gmail.com

William Choi
Yahoo Inc.
Sunnyvale, CA, USA
willchoi@yahooinc.com

Don Matheson
Yahoo Inc.
Sunnyvale, CA, USA
donm@yahooinc.com

ABSTRACT

Online controlled experiments are commonly used to measure how much value new features deployed to the products bring to the users. Although the experiment design is straightforward in theory, running large-scale online experiments can be quite complex. An essential step to run a rigorous experiment is to validate the balance between the buckets (a.k.a. the random samples) before it proceeds to the A/B phase. This step is called A/A validation and it serves to ensure that there is no pre-existing significant difference between the test and control buckets. In this paper, we propose a new matching algorithm to assign users to buckets and improve A/A balance. It has the capability to deal with massive user size and shows improved performance compared to existing methods.

CCS CONCEPTS

• **Mathematics of computing** → *Hypothesis testing and confidence interval computation.*

KEYWORDS

Experimentation; A/B testing; A/A Validation; Matching

ACM Reference Format:
Yan He, Lin Yu, Miao Chen, William Choi, and Don Matheson. 2022. A Cluster-Based Nearest Neighbor Matching Algorithm for Enhanced A/A Validation in Online Experimentation. In *Companion Proceedings of the Web Conference 2022 (WWW '22 Companion), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3487553.3524220

1 INTRODUCTION

Online controlled experiment, also called A/B testing, is one of the most powerful methods for data-driven decision making in many web-facing companies including Amazon, Facebook, Google, etc. [11], [12], [13], [21]. In business settings, in order to pick the best performing variant from a set of options, a randomized experiment is conducted to determine whether one variant has a better performance than the other(s) in terms of user behavior. Although the design of the randomized experiment is straightforward in theory, running large-scale online experiments can be quite complex in practice. One of the main challenges is the bucket A/A imbalance issue [1]. This problem is not unique to a particular company; most internet companies likely experience it–potentially without being aware. For example, a simple two-sample test has 5% chance to be imbalanced in theory using conventional 5% significance level. Imagine running 1000 experiments in the A/A stage, we would expect around 50 tests to be imbalanced due to chance. As a result, our online experimentation platform is expected to have imbalanced experiments opened every day.

In addition, we often observe the multiple comparison issue [4], [19], [22]. When we are interested in a set of metrics in an experiment, they have much higher probability than 5% to yield a significant difference in at least one of the metrics. Assume we look at 5 metrics, the probablity of getting at least 1 significant test is as high as $1 - (1 - 0.05)^5 \approx 22.6\%$. Such pre-existing metric differences could lead to incorrect conclusions and inappropriate product decisions.

In this paper, we propose a method called *Cluster-based Nearest Neighbor Matching* (CNNM) to create experiment buckets. We further illustrate that CNNM is able to achieve the desired level of A/A balance, and expand the A/A validation to multiple metrics.

The rest of the paper is organized as follows: Section 2 discusses the related work. In Section 3, we describe the CNNM algorithm in details. Section 4 provides the offline simulation results to illustrate the advantages. Section 5 gives an overview of the engineering implementation. In Section 6, we use real online experiments to show the effectiveness of the proposed methods. We conclude this paper in Section 7.

2 RELATED WORK

Solutions had been proposed in the prior research to solve the A/A imbalance problem for online experimentation. Kohavi et al. recommended carrying out an A/A period to validate bucket balance before starting the A/B phase of the experiments [14]. Only the balanced buckets are selected for the A/B phase. This method has

[1] Bucket: a random sample of users consisting of an experiment group.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
WWW '22 Companion, April 25–29, 2022, Virtual Event, Lyon, France
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9130-6/22/04…$15.00
https://doi.org/10.1145/3487553.3524220

been widely used by many companies for online experimentation [3], [17]. However, this method is not efficient in practice. It wastes testing space since a proportion of buckets fail the A/A validation. In addition, experimenters must wait a few days for A/A data in order to perform the validation, which is unfavorable to quick product development.

There also have been research studies in experiment design procedures to improve bucket balance. Blocking method [5] [7] divides users into blocks that are further divided into random buckets within each block, so that the buckets are balanced on block dimensions. Re-randomization method [16] repeats the randomization process until A/A balance is established. The Optimization procedure [8] finds the best vector of assignments to treatment in order to minimize a certain measures of imbalance. For online experimentation, a common challenge is how to scale these procedures to millions or billions of users.

Our proposed CNNM method solves the computation problem with user clustering. In this algorithm, we firstly group users into clusters. Instead of matching users in the aforementioned blocking method, we match clusters and achieve the desired level of A/A balance with reduced data size. In the next section, we will introduce the CNNM algorithm in details and prove the sample mean difference is still an unbiased estimator of the average treatment effect (ATE) for the created buckets.

3 METHODOLOGY

The CNNM algorithm is performed in the following two major steps.

3.1 Random Clustering of Users

The first step of our algorithm is to group users into clusters. At our company, hundreds of experiments are conducted everyday in a multi-layer experimentation platform [20] in which users are reshuffled in each layer to expand the testing capability. For a given layer, we assign each user a hash value randomly ranging in $[0, 999]$, with which users are randomly grouped into 1000 clusters. Then, we obtain the cluster mean for the metrics of interest by aggregating users with the same hash value. This way, we reduce the data size to 1000 clusters for matching, no matter how large the original use size is.

3.2 Cluster Matching

With the data of the 1000 clusters, we calculate their pair-wise distance and construct the buckets based on their nearest neighbors [9].

The distance matrix M is calculated first. Each element M_{ij} in the matrix represents the Mahalanobis distance between cluster i and j, and it is calculated based on the metrics of interest. Here we use 4 guardrail metrics for experiments at our company as an example: days visited, page views, network sessions, revenue. We compute the distance M_{ij} as follows:

(1) For each user, compute the values of the 4 guardrail metrics in the preceding seven days
(2) For each cluster i, obtain all the users with the hash value i. Calculate the cluster size and the average metrics
(3) Normalize the metrics calculated in step 2)
(4) Calculate the Mahalanobis distance as M_{ij} using the normalized metrics.

Then the buckets are constructed with the following procedure using the distance matrix. Assume we create three buckets with 5% size:

(1) Randomly select a cluster i from available hash values
(2) Locate two clusters j and k, which have the lowest distance from cluster i
(3) Randomly assign i, j, k into the three buckets
(4) Repeat step 1-3 till the buckets have 5% size.

In general, when creating K buckets instead of 3, we locate $K-1$ nearest clusters for the randomly selected one at each bucket assignment in step 2).

3.3 Asymptotic Property

Here we show the sample mean difference is still an unbiased estimator of the average treatment effect (ATE) for buckets created with the algorithm. Let Y_{ij} be the variable of interest for j^{th} user in the i^{th} cluster $j = 1, \ldots, N_i$ and $i = 0, 1, \ldots, 999$, where N_i is the number of users in i^{th} cluster. Then Y_{ij}^T or Y_{ij}^C is the potential outcome if j^{th} user in i^{th} hash value had received the test or control experience. The treatment effect for user j is $\delta_j = Y_{ij}^T - Y_{ij}^C$.

However, in reality it is impossible to observe the pair (Y_{ij}^T, Y_{ij}^C) for user j. Instead, based on Rubin Causal Model [18], the average treatment effect (ATE) is estimated by the sample average in each bucket,

$$\begin{aligned}\hat{\delta} = \bar{Y}^T - \bar{Y}^C &= \frac{1}{|S_T|}\sum_{i\in S_T}\frac{1}{N_i^T}\sum_{j=1}^{N_i^T} Y_{ij}^T - \frac{1}{|S_C|}\sum_{i\in S_C}\frac{1}{N_i^C}\sum_{j=1}^{N_i^C} Y_{ij}^C \\ &= \frac{1}{|S_T|}\sum_{i\in S_T}\bar{Y}_{i\cdot}^T - \frac{1}{|S_C|}\sum_{i\in S_C}\bar{Y}_{i\cdot}^C\end{aligned}$$

where S^T and S^C are the set of clusters for test and control bucket, respectively. Since each cluster is selected by simple random sampling without replacement from the available clusters and both buckets are sampled from the same population, the procedure of allocating the users into buckets is the same as performing one-stage cluster sampling. Then, the estimated treatment effect $\hat{\delta}$ is an unbiased estimator of the ATE over the population $\delta = \mathbb{E}(Y^T - Y^C) = \mathbb{E}Y^T - \mathbb{E}Y^C$ [6], [10], [15].

4 OFFLINE SIMULATION

In this section, we evaluate the CNNM algorithm via simulations and compare its performance with the following existing methods Yahoo used in the past:

(1) **Random selection**: randomly selects clusters and allocates them into the experiment buckets
(2) **Ready-to-use A/A buckets** [2]: cuts lower and upper 5% tail clusters for each metric, and randomly allocates the remaining clusters to each bucket

To generate simulated data, we start with actual user-level data collected from our product and then randomly assign a hash value in $[0, 999]$ to each user. Users with the same hash value are grouped

into one cluster resulting in 1000 clusters. We use this dataset to run the simulation study.

4.1 Imbalance Rate Comparison for Various Bucket Sizes

As we discussed in Section 1, a challenge of running multiple experiments is having high imbalance rate in the A/A stage, thus leading to incorrect test conclusions. In this subsection, we compare the imbalance rate of the 3 methods with simulations and illustrate that the CNNM algorithm generates much lower imbalance rate compared to the other two methods.

In the simulation, we generate a bucket pair by assigning clusters into each bucket, and test if their page views difference is statistically significant at 5% significance level. We simulated the bucket pairs 1000 times for each method and calculated the imbalance rate for comparison. Then we performed the simulation for various bucket sizes and presented the results in Figure 1.

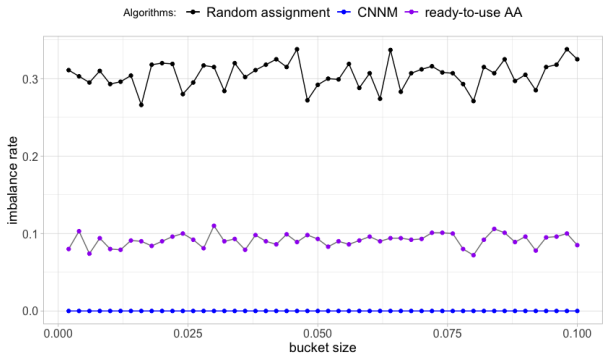

Figure 1: Imbalance rate comparison for various bucket sizes

It is shown that the CNNM algorithm has much better performance than the two baseline algorithms. It decreases the imbalance rate to almost 0 consistently across all bucket sizes, which means the new method could guarantee the A/A balance for the bucket pairs with high confidence.

Random selection method has the worst performance, with more than 100 bucket pairs out of 1000 failing across all bucket sizes. It is clear that the A/A imbalance issue is severe with simple random bucket assignment. Ready-to-use A/A effectively reduces the bucket imbalance rate to 5%, which is still less effective than CNNM.

4.2 Bucket Imbalance Rate Comparison for Multiple Metrics

In practice, we commonly look at multiple metrics in order to evaluate online experiment performances. Due to multiple comparison issue, the chance of A/A imbalance increases quickly with more metrics to compare if we use random bucket user assignment. But the CNNM algorithm is able to assure the balance for multiple metrics.

With a similar setting as Section 4.1, we expanded the number of metrics to compare starting from one to seven metrics (days visited, page views, time spent, total revenue, display revenue, video revenue, stream ads revenue). In Figure 2, we show the bucket imbalance rates of the three compared algorithms versus various number of metrics. As described earlier, the random selection method suffers from the multiple comparison issue. Its imbalance rate increases when additional metrics are added for comparison. The ready-to-use A/A method is more robust than random selection, but its imbalance rate is still as high as 10% with more than 4 metrics or more. As a contrast, the proposed CNNM algorithm stays robust with the smallest imbalance rate and no performance deterioration when the number of metrics increases.

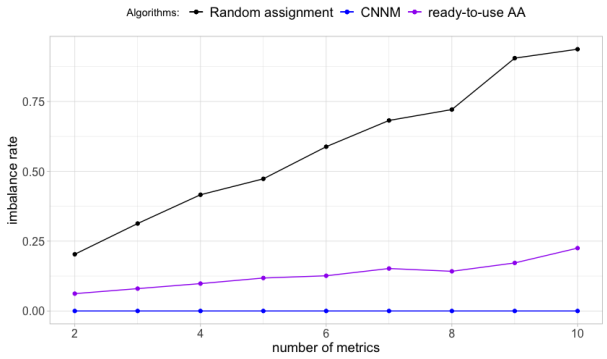

Figure 2: Bucket imbalance rate comparison for different number of metrics comparison

5 ENGINEERING IMPLEMENTATION

In this section we outline how the CNNM algorithm is implemented in our data platform in details.

5.1 Implementation Process Outline

Figure 3 illustrates the offline data pipeline of the multi-layer experimentation platform.

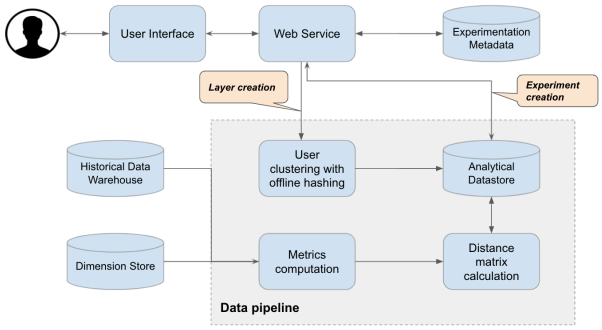

Figure 3: Offline data pipeline for implementing the *Nearest Neighbor Matching algorithm*

(1) **User clustering**: when an experimentation layer is created in the platform, a hash value in $[0, 999]$ is assigned to each user using an offline hash function with the unique layer seed

(2) **Distance matrix calculation**: on a daily basis, we compute the values of the metrics of interest by aggregating the preceding 7-day data for each user and obtain the cluster level metrics mean. Then we calculate the Mahalanobis distance matrix M between the 1000 clusters
(3) **Experiment creation**: when a request is received to create a new experiment, the clusters are assigned to buckets following the procedure in Section 3.2

5.2 Speed Up Bucket Creation Process

We propose the following procedure for speedy cluster matching in order to create new buckets:

(1) Pre-calculate the distance matrix for each layer daily and store them in the disk
(2) When an experiment creation request is received, randomly select 100 clusters and read their 100×100 distance matrix into RAM for matching
(3) In the iterative matching process to construct buckets, when the available clusters in the RAM are below some threshold (say 50), read another 100 random clusters and their distance matrix from the disk to continue the matching process
(4) Repeat Step 2)-3) till we have desired bucket size

In this procedure, a sub sample of the generated clusters is used in order to complete the matching with minimum time. It is known that in user-level matching some outliers are not able to find a match in a sub sample for a specified threshold in distance. We mitigate the challenge using clusters because the aggregated metrics at the cluster level are centered to the population mean with less severe outlier problems. The aforementioned bucket assignment process runs much faster in a sub sample by reducing the distance matrix size by 99% from 1000×1000 to 100×100. As a result, the platform can respond to experiment creation request in a timely manner.

6 ONLINE VALIDATION

In this section, we discuss the setup and results of online validation conducted on a few internet products, and compare the performance between CNNM algorithm and the baseline methods.

6.1 Experimental Design

To illustrate the performance between the new approach and the baselines with the real dataset, we follow the steps in Section 5 to create a few online experiments for three different products, say product A, B and C[2]. Specifically, we use each method to create three experiments for each of the products and keep all the parameters the same. Unless otherwise stated, these are the defaults:

- Each experiment contains five buckets. In order to measure the performance on different bucket size, we used 1% buckets for product A and 5% for product B and C
- Five metrics to be validated in each bucket pair: days visited, page views, sessions, revenue and the bucket size

6.2 Experimental Results

We looked into a 2-week period of data to compare the imbalance rate between the CNNM algorithm and the baseline methods. In particular, we pull the five default metrics to conduct the A/A validation in a pairwise fashion for all the five buckets. Therefore, with ten bucket pairs and five metrics, a total of 50 hypothesis tests were carried out for each experiment. The imbalance rate out of 50 tests were calculated to evaluate the validation results.

Table 1: The imbalance rate comparison between CNNM algorithm and the baselines

5 metrics, 5 buckets		Imbalance Rate		
Product	Bucket Size%	CNNM	Random Assignment	Ready-to-use A/A
A	1%	0.57%	13.43%	4.57%
B	5%	6.57%	11.71%	18.00%
C	5%	1.43%	8.57%	3.43%

The results in Table 1 show that our approach has much lower imbalance rate than the other two methods across all the products. We repeated this procedure for different time period and the results were consistent. It is worthwhile to note that the online results were not as perfect as the ones from simulation. There are two possible reasons to explain the gaps:

(1) **Inaccurate data logging**: in our data pipelines, a user falling into an experiment bucket will be stamped with the corresponding test ID, which is referred as the bucket stamping process. In practice two issues could arise in this process: (1) missing test ID and (2) incorrect test ID [1]. They are rare but when existing could lead to a discrepancy between the computed matrix and the online measurement.
(2) **Seasonal effect**: as described in Section 3, the CNNM algorithm allocates the clusters into experiment buckets based on the data observed in the preceding seven days. When there is a strong seasonality, the balance established from the history cannot guarantee the same for the future, even if the freshest data is used in the methodology. In future studies, expanding the historical period used for validation as well as understanding a specific product's longitudinal data pattern should further improve the algorithm's performance.

7 CONCLUSIONS

Online experimentation plays an essential role in the product development cycles in many IT companies. In this paper we present a novel framework for the bucket assignment process implemented in a large scale multi-layered experimentation platform. It eliminates the need of conducting manual A/A validation before the A/B phase and guarantees the A/A balance with improved confidence. Besides, the algorithm can be applied to an online experimentation platform dealing with multiple metrics and massive amount of users. We hope this paper can benefit online experimentation practitioners and motivate additional research in this domain.

[2]Hide the product names due to the confidentiality

ACKNOWLEDGMENTS

The authors would like to thank Mahendrasinh Ramsinh Jadav, Sai Sree Kamineni, Chandrashekhar Shaw for their help in implementing the new approach and setting up layers and experiments for testing, and Aishwarya Iyer for her UI work on the platform.

REFERENCES

[1] Nirupama Appiktala, Miao Chen, Michael Natkovich, and Joshua Walters. 2017. Demystifying dark matter for online experimentation. In *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 1620–1626.

[2] Russell Chen, Miao Chen, Mahendrasinh Ramsinh Jadav, Joonsuk Bae, and Don Matheson. 2017. Faster online experimentation by eliminating traditional a/a validation. In *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 1635–1641.

[3] Thomas Crook, Brian Frasca, Ron Kohavi, and Roger Longbotham. 2009. Seven pitfalls to avoid when running controlled experiments on the web. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1105–1114.

[4] Charles W Dunnett. 1955. A multiple comparison procedure for comparing several treatments with a control. *J. Amer. Statist. Assoc.* 50, 272 (1955), 1096–1121.

[5] Robert Greevy, Bo Lu, Jeffrey H Silber, and Paul Rosenbaum. 2004. Optimal multivariate matching before randomization. *Biostatistics* 5, 2 (2004), 263–275.

[6] Ralph H Henderson and Thalanayar Sundaresan. 1982. Cluster sampling to assess immunization coverage: a review of experience with a simplified sampling method. *Bulletin of the World Health Organization* 60, 2 (1982), 253.

[7] Michael J Higgins, Fredrik Sävje, and Jasjeet S Sekhon. 2016. Improving massive experiments with threshold blocking. *Proceedings of the National Academy of Sciences* 113, 27 (2016), 7369–7376.

[8] Nathan Kallus. 2018. Optimal a priori balance in the design of controlled experiments. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 80, 1 (2018), 85–112.

[9] James M Keller, Michael R Gray, and James A Givens. 1985. A fuzzy k-nearest neighbor algorithm. *IEEE transactions on systems, man, and cybernetics* 4 (1985), 580–585.

[10] Sally M Kerry and J Martin Bland. 1998. The intracluster correlation coefficient in cluster randomisation. *Bmj* 316, 7142 (1998), 1455–1460.

[11] Ronny Kohavi, Thomas Crook, Roger Longbotham, Brian Frasca, Randy Henne, Juan Lavista Ferres, and Tamir Melamed. 2009. Online experimentation at Microsoft. *Data Mining Case Studies* 11, 2009 (2009), 39.

[12] Ron Kohavi, Alex Deng, Brian Frasca, Toby Walker, Ya Xu, and Nils Pohlmann. 2013. Online controlled experiments at large scale. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1168–1176.

[13] Ron Kohavi and Roger Longbotham. 2017. Online Controlled Experiments and A/B Testing. *Encyclopedia of machine learning and data mining* 7, 8 (2017), 922–929.

[14] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M Henne. 2009. Controlled experiments on the web: survey and practical guide. *Data mining and knowledge discovery* 18, 1 (2009), 140–181.

[15] Sharon L Lohr. 2019. *Sampling: design and analysis*. Chapman and Hall/CRC.

[16] Kari Lock Morgan and Donald B Rubin. 2012. Rerandomization to improve covariate balance in experiments. *The Annals of Statistics* 40, 2 (2012), 1263–1282.

[17] Eric T Peterson. 2004. *Web analytics demystified: a marketer's guide to understanding how your web site affects your business*. Bookrenter.

[18] Paul R Rosenbaum and Donald B Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70, 1 (1983), 41–55.

[19] Dave J Saville. 1990. Multiple comparison procedures: the practical solution. *The American Statistician* 44, 2 (1990), 174–180.

[20] Diane Tang, Ashish Agarwal, Deirdre O'Brien, and Mike Meyer. 2010. Overlapping experiment infrastructure: More, better, faster experimentation. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 17–26.

[21] Ya Xu and Nanyu Chen. 2016. Evaluating mobile apps with A/B and quasi A/B tests. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 313–322.

[22] Ya Xu, Nanyu Chen, Addrian Fernandez, Omar Sinno, and Anmol Bhasin. 2015. From infrastructure to culture: A/B testing challenges in large scale social networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2227–2236.