INVESTIGATION AND ANALYSIS OF STATISTICAL ATTENTION MECHANISMS IN CLICK-THROUGH-RATE PREDICTION: THE IMPACT OF LAYER NORMALIZATION AND INTERACTION COMPONENT INTEGRATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

EGE BERK BÜYÜKBAŞ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

SEPTEMBER 2024

Approval of the thesis:

INVESTIGATION AND ANALYSIS OF STATISTICAL ATTENTION MECHANISMS IN CLICK-THROUGH-RATE PREDICTION: THE IMPACT OF LAYER NORMALIZATION AND INTERACTION COMPONENT INTEGRATION

submitted by **EGE BERK BÜYÜKBAŞ** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Naci Emre Altun
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Halit S. Oğuztüzün
Head of Department, **Computer Engineering** _____

Prof. Dr. Pınar Karagöz
Supervisor, **Computer Engineering, METU** _____

Prof. Dr. Cem İyigün
Co-supervisor, **Industrial Engineering, METU** _____

Examining Committee Members:

Prof. Dr. İsmail Hakkı Toroslu
Computer Engineering, METU _____

Prof. Dr. Pınar Karagöz
Computer Engineering, METU _____

Assoc. Prof. Dr. Burkay Genç
Computer Engineering, Hacettepe University _____

Date:05.09.2024

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Ege Berk Büyükbaş

Signature :

ABSTRACT

INVESTIGATION AND ANALYSIS OF STATISTICAL ATTENTION MECHANISMS IN CLICK-THROUGH-RATE PREDICTION: THE IMPACT OF LAYER NORMALIZATION AND INTERACTION COMPONENT INTEGRATION

Büyükbaş, Ege Berk
M.S., Department of Computer Engineering
Supervisor: Prof. Dr. Pınar Karagöz
Co-Supervisor: Prof. Dr. Cem İyigün

September 2024, 76 pages

The accurate prediction of Click-Through Rate (CTR) is a key metric for enhancing user experience and optimizing revenue in online shopping and e-commerce businesses. This study explores the suitability of various statistical attention mechanisms—mean attention, max attention, mean-max, mean-std attention, and bitwise attention—under different hyper-parameter candidate sets within the most common and conventional CTR prediction algorithms. By conducting extensive experiments across the most commonly used open-source datasets for CTR prediction, this empirical study examines whether these attention mechanisms can effectively boost the informational utility of each field's low-dimensional feature embedding, potentially leading to improved prediction accuracy.

Our findings show that each attention mechanism behaves uniquely across different algorithms and datasets. The application of these attention mechanisms to traditional CTR prediction models may demonstrate significant improvements in prediction per-

formance by implicitly focusing on relevant features and their interactions. This research aims to contribute to the field of CTR prediction by providing a comprehensive analysis of how different attention mechanisms can enhance the predictive ability of well-known conventional CTR prediction algorithms and offer insights for the future development of more sophisticated and accurate CTR prediction systems.

Keywords: Click-Through Rate (CTR), Statistical Attention Mechanisms, Interaction Component, Deep Learning, Feature Embedding

ÖZ

TIKLAMA ORANI TAHMİNİNDE İSTATİSTİKSEL DİKKAT MEKANİZMALARININ İNCELENMESİ VE ANALİZİ: KATMAN NORMALLEŞTİRMENİN VE ETKİLEŞİM BİLEŞENİ ENTEGRASYONUNUN ETKİSİ

Büyükbaş, Ege Berk
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Pınar Karagöz
Ortak Tez Yöneticisi: Prof. Dr. Cem İyigün

Eylül 2024 , 76 sayfa

Tıklama Oranı (CTR) tahmininin doğru yapılması, çevrimiçi alışveriş ve e-ticaret işletmelerinde kullanıcı deneyimini artırmak ve gelirleri optimize etmek için önemli bir ölçüttür. Bu çalışma, en yaygın ve geleneksel CTR tahmin algoritmalarında farklı hiper-parametre aday setleri altında çeşitli istatistiksel dikkat mekanizmalarının (attention mechanisms)—ortalama dikkat (mean attention), maksimum dikkat (max attention), ortalama-maksimum (mean-max attention), ortalama-standart sapma dikkat (mean-std attention) ve bitwise dikkat (bitwise attention)—uygunluğunu araştırmaktadır. CTR tahmini için en yaygın kullanılan açık kaynaklı veri kümeleri üzerinde kapsamlı deneyler yaparak, bu dikkat mekanizmalarının her alanın düşük boyutlu özellik gömme işleminin bilgi faydasını etkili bir şekilde artırıp artırmadığını ve bunun da tahmin doğruluğunu iyileştirmeye yol açıp açmayacağını inceleyen ampirik bir çalışmadır.

Bulgularımız, her bir dikkat mekanizmasının farklı algoritmalar ve veri kümeleri üze-

rinde benzersiz davrandığını göstermektedir. Bu dikkat mekanizmalarının geleneksel CTR tahmin modellerine uygulanması, ilgili özelliklere ve bunların etkileşimlerine dolaylı olarak odaklanarak tahmin performansında önemli iyileştirmeler gösterebilir. Bu araştırma, çeşitli dikkat mekanizmalarının tanınmış geleneksel CTR tahmin algoritmalarının tahmin yeteneğini nasıl artırabileceğini kapsamlı bir şekilde analiz ederek CTR tahmini alanına katkıda bulunmayı ve daha sofistike ve doğru CTR tahmin sistemlerinin gelecekteki geliştirilmesine yönelik içgörüler sunmayı amaçlamaktadır.

Anahtar Kelimeler: Tıklama Oranı, İstatistiksel Dikkat Mekanizmaları, Etkileşim Bileşeni, Derin Öğrenme, Özellik Gömme

To my family.

ACKNOWLEDGMENTS

This work is funded by Scientific and Technological Research Council of Turkey (TUBİTAK) under BIDEB 2210-A funding.

I would like to express my gratitude to my supervisors Prof. Dr. Pınar Karagöz and Prof. Dr. Cem İyigün for their guidance, encouragement and criticisms.

I would like to thank my colleagues at Trendyol for always pushing me to be better.

I would like to thank to my fiancee Cansu Arslan for bringing joy to my life and always supporting me.

Lastly, I would like to thank my family. I could not have achieved this degree without their support and the love they gave me.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS	
DNN	Deep Neural Network
MLP	Multi Layer Perceptron
CTR	Click-Through Rate
LR	Logistic Regression
FM	Factorization Machines
DCN	Deep & Cross Network
CIN	Compressed Interaction Network
SENET	Squeeze and Excitation Network
NFM	Neural Factorization Machines
AFM	Attentional Factorization Machines
DIN	Deep Interest Network
DSIN	Deep Session Interest Network
FFM	Field-aware Factorization Machines
MMBAttn	Mean-Max and Bitwise Attention Network
RNN	Recurrent Neural Networks

CHAPTER 1

INTRODUCTION

The accurate prediction of Click-Through Rate (CTR) is one of the most important objectives in the fast-paced environment of online shopping and e-commerce. CTR serves as a critical metric for enhancing user experience and optimizing revenue, making it a key objective for businesses aiming to maximize their online engagement and profitability. Although there have been significant advances in CTR prediction algorithms, there remains substantial potential for improvement and analysis, particularly through the application of sophisticated attention mechanisms.

A minor uplift in CTR prediction can lead to significant improvements in the revenue for the industry [1, 2, 3, 4]. Accurate CTR prediction not only helps to increase the personalized user experience but also increases revenue in the e-commerce advertisements [5]. As a result, online shopping and e-commerce businesses heavily invest in implementing advanced models and algorithms to achieve more accurate CTR predictions, as even small gains can lead to significant financial impacts. In addition to financial concerns, in the literature, from the simplest models (e.g. Logistic Regression) to the current state-of-the-art (SOTA) deep learning models (e.g. FinalMLP), the AUC values in some datasets (e.g., Avazu) increased from 75.16 to 76.66, indicating that the percentage uplift was only about 2% [6].

This thesis aims to explore the suitability and effectiveness of various statistical attention mechanisms on the most common and robust CTR prediction algorithms. These attention mechanisms have shown promise in other domains, such as computer vision and natural language processing, yet their application in CTR prediction remains under-explored. Although there are several models that implement attention mechanisms in their approach to CTR prediction, most studies with attention mechanisms

focus on achieving an uplift rather than understanding the underlying reasons why they work. By systematically analyzing these mechanisms under different hyperparameter candidate sets, this thesis seeks to uncover and investigate the potential to improve the informational utility of each field's low-dimensional feature embeddings and to understand why these attention mechanisms work or do not work in different contexts.

To this end, a sufficient number of extensive experiments were conducted using a diverse set of four widely-used open-source online datasets for CTR prediction. This empirical experiment design provides a robust framework for evaluating how each attention mechanism interacts with various algorithms and datasets. To maximize the benefits of attention mechanisms in CTR prediction, we observed that each attention mechanism application strategy exhibits unique behaviors in different contexts.

The findings of this study show that while not all of the considered attention mechanisms significantly enhance traditional CTR prediction models, some do in specific contexts. By updating the feature embeddings and focusing on the most relevant features and their interactions, these mechanisms can contribute to improved prediction performance. This research not only highlights the potential of attention mechanisms in improving CTR prediction accuracy but also aims to provide valuable insights for future developments in this field.

1.1 Problem Definition

The CTR prediction task involves extracting relevant feature interactions, with features represented using low-dimensional feature embeddings. While interactions can be manually specified, this approach is challenging to implement across all domains and requires substantial domain knowledge. Therefore, it is crucial to have models that can automatically capture relevant feature interactions. Several methods exist to extract these feature interactions and to enhance their performance. Attention mechanisms are one such method that can be used for both extracting feature interactions and improving the performance of the extracted feature interactions.

There are a few attention mechanisms in the literature that update feature embeddings

to capture relevant feature interactions. However, most studies report only performance uplift without determining which attention mechanism is superior to others in different contexts. The aim of this thesis is to examine how the attention mechanisms known in the literature (Mean Attention, Max Attention, and their joint versions) and the ones we proposed (Mean-Standard Attention) behave with different architectures, models, and datasets, and to investigate whether generalizations about their performance can be made.

We aim to determine whether the attention mechanisms we experimented with provide improvements over the most conventional models and how layer normalization affects models with attention mechanisms through experimentation. Additionally, we investigate the performance of attention mechanisms that do not feed the interaction layer and explore whether a superior attention mechanism architecture can be generalized for all datasets and models. This investigation necessitates extensive experimentation for effective generalization.

1.2 Contributions and Novelties

The contributions of this thesis can be summarized as follows:

- A comprehensive examination of the performance of well-known attention mechanisms, as well as additional ones introduced in this study, across various architecture, model, and dataset selections.
- A detailed analysis of the performance of the attention mechanisms we experimented with, focusing on their ability to provide improvements over the most conventional models.
- An investigation into the impact of attention mechanisms with layer normalization on conventional models.
- An analysis of the performance of attention mechanisms that are not integrated into the interaction layer.
- An exploration of the potential for generalizing an optimal attention mechanism architecture for all datasets and models.

- A summary of the potential drawbacks and advantages of attention mechanisms across different types of datasets and models.

1.3 The Outline of the Thesis

In this thesis, the chapters are organized as follows:

Chapter 1 provides an introduction, including the problem definition and contributions of this study. It also contains a comprehensive literature review of CTR prediction models and known attention mechanisms in this field, highlighting the research gaps in the literature regarding attention mechanisms.

Chapter 3 presents the required theoretical background. It details feature embedding, the most common CTR prediction algorithms, and statistical attention mechanisms.

Chapter 4 describes the proposed attention mechanisms and different architecture sets that we experimented with. The attention mechanisms can use different summary statistic methods, may include a layer normalization step, or may have outputs that do not feed into the interaction layer. The details of these variations are explained in this chapter.

Chapter 5 outlines the experimental setups and presents the results, accompanied by a detailed discussion section to support or refute our hypotheses about the performance of the proposed methods.

Finally, Chapter 6 concludes the thesis with a summary of the findings. It also discusses future work and potential improvements that could be made to the proposed methods.

CHAPTER 2

LITERATURE REVIEW

2.1 Click-Through Rate Prediction

This section reviews the key methods and models in the historical evolution of CTR prediction, with a particular focus on feature interaction methods. This evolution can be categorized into three main subsections: traditional linear models, shallow interaction models, and deep learning-based models. The growing complexity of user behavior and data sizes and the corresponding need for more sophisticated models to learn complex feature interactions led to new developments and approaches in the history of CTR prediction field. The timeline of the CTR models and their corresponding categories can be seen in Table 2.1.

2.1.1 Early Linear Models

The earliest approaches heavily relied on tradition linear models, particularly Logistic Regression (LR) [7, 8]. Because of its simplicity, ease of deployment in the industry and interpretability, LR became one of the most adopted model in this field. The traditional linear model can easily handle large-scale data with clear understanding of which features are the most important for the task. However, LR does not able to learn nonlinear interactions which are highly important for CTR prediction, limiting the performance of the predictions. As the volume of the data and the complexity of user behaviors increased, the limitations of LR become more restrictive.

2.1.2 Low Order Interaction Models

To address the shortcomings of linear models, Factorization Machines (FM) were introduced as a way to model pairwise feature interactions efficiently [9]. FMs allow to model feature interactions by using latent vectors to represent each feature. Instead of expliticitly adding a feature interaction term for every possible pair, FMs can effectively capture the second-order interactions without needing to explicitly add each pair. FMs introduce a more efficient way to model feature interactions by using latent vectors to represent each feature. This has been proved to handle sparse data which is a common case in recommender systems where field interactions are highly sparse. Another advantage of factorizing the interaction terms into a product of latent factors is reducing the number of parameters that need to be learned, so FMs reduce the number of parameters to be learnt when it is compared to manually adding pairwise interactions in logistic regression. The other advantage which is worth to highlight is generalization capability of FMs. In LR with pairwise feature interactions, if some feature combinations never appear in the training data, the model will not able to learn these interactions, but FMs generalize better by leveraging the latent vectors to infer interactions for unseen feature pairs.

Briefly, FM's ability to capture pairwise interactions without manual feature engineering marked a significant improvement over LR. However, the scope of FM was limited to second-order interactions, which constrained its ability to fully model complex, high-order relationships between features. As a result, while FM outperformed traditional linear models, it still fell short of the growing demands for higher-order interaction modeling in CTR prediction tasks.

2.1.3 Transition to High Order Interactions

To address the limitations of models like FM in capturing only second-order interactions, the development of more complex models became necessary. The vanilla Deep Neural Networks (DNN) is good at learning the high-order feature interactions, but they have a problem with generalization.

The Wide & Deep [10] is one of the first model that attempts to utilize both low

and high level feature interactions. It combines a linear (Wide) component and a neural network (Deep) component, trained jointly. The two components-Wide and Deep- are responsible for memorization and generalization respectively. The Wide component is feature engineered cross interactions of features and needs a manual pairwise interaction extraction with a linear model, but it is effective for memorization of relevant items. On the other hand, the Deep part does not require any feature engineering and it is better to generalize feature interaction which rarely occur in the dataset but it is more prone to over-generalization if feature interactions are too sparse.

While the low-order feature interactions should be extracted by manually in the Wide part, it is challenging to implement this for all tasks. To address this, Factorization Machines (FM) [9] were introduced as a solution for efficiently modeling second-order interactions and are particularly effective in sparse data scenarios, as detailed in subsection 2.1.2. However, despite their success in capturing second-order interactions, FM alone is insufficient for the growing need to model complex high-order relationships.

Building on the groundbreaking performance of the Wide & Deep model, the DeepFM [2] model was proposed. DeepFM has a similar structure with Wide & Deep where there are two components for low and high-order feature interactions. The most important difference is that DeepFM leverages the FM component to extract second-order feature interactions, instead of relying on manually feature-engineered Wide component. DeepFM is considered as one of the state-of-the-art models in CTR prediction due to it is robust and effective for all open source datasets in online. [11]

One of the biggest disadvantages of DeepFM and FM-based architectures is the limitations for its representative power for high-order feature interactions due its shallow structure. There have been several studies about extending to higher orders such as High Order Factorization Machines (HOFM) [12], but it results with huge computational cost due to the enormous number of parameters. The Deep & Cross Network (DCN) [3] model was proposed to efficiently learn bounded high-order feature interactions. The structure of the proposed model is similar to DeepFM, with only a few small differences. It has two components where Deep part is exactly same as

DeepFM's Deep component, and Cross Network is responsible for low-order feature interactions. Although Cross Network and FM are responsible for low-order feature interactions, the Cross Network is able to learn any degree of feature interactions theoretically thanks to cross layers. The highest polynomial degree increases at each cross layer degree which does not require any manual feature engineering. The DCN model is also a state-of-the-art approach in CTR prediction due to its simple yet effective architecture.

Another conventional model is the xDeepFM [13] model which contains improvements on DeepFM by introducing the Compressed Interaction Network (CIN) as a replacement of FM component. The CIN layer can capture feature interactions at vector-wise level but in a very efficient way by compressing the interactions with convolutions.

Another model that should be mentioned and that greatly affects our work is FiBiNet [14] model. FiBiNet model does not have a joint two streams as deep and interaction components. The feature interaction extraction is applied before the deep part in the model. The feature interaction part contains two Bilinear Interaction layers which are fed by original feature embeddings and updated feature embeddings. The update mechanism is based on SENet (Squeeze and Excitation Network) [15] layer, and the SENet layer initially used in computer vision area with remarkable results. The aim of SENet layer is updating the feature embeddings to increase the effects of relevant features and interactions in the model. It is actually an attention mechanism in that sense, and in our study, we experimented different type of SENet-Like architectures as our attention mechanisms.

In our experiments, we chose DNN, DeepFM and DCN as our base models which are the most common and conventional models, but there are also other worth to mention models in the field of CTR prediction. The AutoInt [4] model is another notable advancement in the field of CTR prediction. AutoInt uses self-attention mechanisms with residual connections that allow model to learn low-order and high-order feature interactions without any feature engineering. Another one is Neural Factorization Machines (NFM) [16] which surpass some of the limitations of the FM by leveraging the neural networks at the top of bilinear interaction component. This aggregation

allows NFM to catch more complex relationships between features better than FM.

2.2 Attention Networks in CTR Prediction

The attention mechanism is a powerful tool to dynamically extract relevant features and interactions. The power of the attention mechanisms is a well-established concept in natural language processing and computer vision field [17, 18, 19]. There are different type of attention mechanisms, which are used in CTR prediction domain, with different objectives.

The earliest adoption of the attention mechanism in CTR prediction domain is the Attentional Factorization Machines (AFM) [20], which uses the attention mechanism to update the weights of the FM's pairwise interaction outputs. The pairwise interaction outputs are fed into a attention network that has a MLP to calculate the weights for each pair, then multiplied with pairwise interaction output. Attention mechanism are also used for capture the sequential features in CTR prediction task.

The attention mechanism is also extensively used for capturing sequential features in user behavior, which is crucial for personalized recommendation. The self-attention mechanism is used to retrieve relevant sequential features in user history at Deep Interest Network (DIN) [21] and Deep Session Interest Network (DSIN) [22]. DIN implements attention mechanism to dynamically select the most relevant items from the user's historical actions with respect to target item. This leads model to update the weights according to different historical interactions and their relevance to the user's current interest. DSIN, on the other hand, extends this idea in terms of session-based sequential self-attention mechanism. One of the biggest advantage of using self-attention over traditional recurrent neural networks (RNN), which process sequences sequentially, is self-attention mechanism can efficiently handle long sequences of user interactions thanks to parallel processing of the sequences.

The self-attention mechanism not only being used for sequential features but also in the AutoInt [4], it is implemented to learn feature interactions by leveraging multi-head self attention network with residual connections as we described above. The aim of using residual connections is preserving the original features alongside the

attention-based feature interactions. Another model that feeds updated vectors with attention mechanism is DAFM (Deep Attention Factorization Machine) model, but with one big difference which is the updated vectors are not fed into the FM component, which is the low order interaction component in the architecture, but only the Deep part of the model [23].

Other than self attention mechanism, the Squeeze and Excitation Network (SENet) [15] based attention mechanisms are highly used for updating the feature embeddings. The performance of the SENet was already a known fact in computer vision, but FiBiNet [14] is the first model that uses SENet in its architecture. SENet is used to update the feature embeddings which will be fed into Bilinear Interaction layer, and it learns the feature importance and related interactions with SENet layer. SENet layer has three parts which are squeeze, excitation and reweight. In the squeeze component, the summary statistic of each field embedding to represent global information about each field embedding. The summary statistic is mean pooling in FiBiNet and mean and max pooling with groups in FiBiNet++ [24] which is low-parameter and high-performer version of FiBiNet. The excitation part is where the weight is are calculated by using all global information coming from squeeze part for the instance by applying a MLP which reduces and repairs the dimension of the input vector. The final part is reweight and it produces the updated feature embeddings by multiplying the weights of the excitation part and original feature embeddings. Then, these new feature embeddings are fed into Bilinear Interaction to create CTR prediction. FiBiNet++ also leverages skip-connections by elementwise addition and layer normalization after the reweight step.

There are other CTR prediction models that incorporate the SENet mechanism in various forms. The FAT-DeepFFM [25] model uses DeepFM-like architecture with a SENet-like mechanism. Instead of the FM component in DeepFM, it employs Field-aware factorization machines (FFM) [26], which increase the parameter size as each pair in FFM has its own weight. The attention mechanism is proposed as CENet (Compose Excitation Network), which shares components with SENet but uses $1x1$ convolution as the pooling operation to create summary statistics. FAT-DeepFFM updates the feature embeddings before the DeepFFM part in CENet layer.

Another model is ECANFM [27], which combines ECANET [28], based on the ECA method in the computer vision field, with NFM [16]. ECANET is also another SENet-like attention mechanism, and in its squeeze part, it also uses mean pooling. However, the excitation parts differ, as ECANET uses one-dimensional convolution instead of an MLP structure in its excitation part.

Finally, there is an attention module named MMBAttn [29] that focuses solely on an attention mechanism based on the SENet architecture, with vector-wise and bit-wise attentions. It implements mean and max pooling in parallel in the squeeze part for attentions, summing them to create feature embedding updates. Additionally, The bit-wise attention, which is a special type of SENet-like attention, emphasizes relationships between individual bits within the features. They experimented with mean-attention, max-attention, mean-max attention, bit-wise attention, and a joint method on a DNN model, and tested the joint method on a few SOTA models to evaluate the proposed method's usefulness.

Table 2.1: Evolution of CTR Prediction Models and their Key Contributions. SE-based means SENet-wise Attention Mechanism is applied.

Category	Model Name	Year	Key Contributions
Early Linear Models	Logistic Regression (LR)	2007	Simple, easy to deploy, interpretable model, but limited to linear interactions, struggles with non-linear interactions.
Low Order Interaction Models	Factorization Machines (FM)	2010	Efficiently captures pairwise feature interactions, handles sparse data, reduces parameter space.
High Order Interaction Models	Wide & Deep	2016	Two components for memorization and generalization by utilizing linear models and deep models respectively, captures both low and high-order interactions. Requires manual feature engineering.
High Order Interaction Models	Higher-Order FM (HOFM)	2016	Extends FM to higher-order interactions with huge computational costs.
High Order Interaction Models	DeepFM	2017	Automates feature engineering by leveraging FM component to replace linear part, captures both low and high-order interactions.
High Order Interaction Models	Deep & Cross Network (DCN)	2017	Learns bounded high-order interactions efficiently, avoids manual feature engineering, simple architecture.
Attention-based Models	Attentional FM (AFM)	2017	Uses attention mechanism to weight FM's pairwise interactions.
High Order Interaction Models	xDeepFM	2018	Introduces Compressed Interaction Network (CIN) as replacement of FM to capture explicit vector-wise interactions efficiently.
Attention-based Models	Deep Interest Network (DIN)	2018	Uses attention to dynamically select relevant user historical interactions for the target item.
Attention-based Models	Deep Session Interest Network (DSIN)	2019	Session-based attention model, handles long session efficiently with self-attention.
Attention-based Models	AutoInt	2019	Leverages multi-head self-attention to learn both low- and high-order feature interactions without feature engineering. Residual connections to preserve the original features.
SE-based Attention Models	FiBiNet	2019	Uses SENet for dynamic feature importance, applies bilinear interaction layers for CTR prediction.
SE-based Attention Models	FAT-DeepFFM	2019	Replaces FM with Field-Aware FM (FFM), introduces CENet for feature importance with SENet-like mechanism.
SE-based Attention Models	ECANFM	2021	Combines NFM and ECANET, an efficient attention model using 1D convolutions for feature importance.
Attention Models	DAFM	2022	DeepFM architecture with attention mechanism, the updated vectors are not fed into the FM component
SE-based Attention Models	FiBiNet++	2023	Uses mean and max pooling with groups, low parameter, high-performer. Skip-connections to preserve information. Layer normalization for a better representation of updated weights.
SE-based Attention Models	MMBAttn	2023	Introduces Max-Mean and Bit-wise attention for CTR prediction, combines mean-max and bit-wise attentions.

CHAPTER 3

PRELIMINARIES

This chapter describes the essential components of conventional CTR prediction models and our proposed squeeze-and-excitation-based attention mechanisms.

3.1 Feature Embedding

In domains such as computer vision and natural language processing, data is often spatially correlated, allowing raw features to be directly used as dense features. However, recommender systems typically deal with sparse, categorical raw data that can be transformed into high-dimensional features using one-hot encoding. Embedding layers enable these high-dimensional features to be converted into low-dimensional, dense representations. For example, a user who is male and likes sports can be represented as high-dimensional sparse features using one-hot encoding, as shown below: `[(1,0), (0,0,1,0,...)]`, where the first field represents gender and the second field represents the type of activity the user likes. An embedding layer can be implemented to compress the information contained in these sparse, high-dimensional features. Another advantage of using feature embeddings is that they produce same-length embeddings, even when the original feature lengths of different fields vary. To generalize, an input feature value vector$X = \{x_{\text{field}_1}, \ldots, x_{\text{field}_M}\}$ which stores one-hot vectors for categorical fields and raw values for continuous fields, can be transformed into a field embedding vector $E = \{e_1, \ldots, e_M\}$ where M is the number of feature fields, and $e_i \in \mathbb{R}^D$, where D is the embedding dimension and i represents the i-th field.

Embedding layers are applied not only to categorical features but also to numerical

features in our study. An example transformation for an instance containing three input features—two categorical and one numerical—can be seen in Figure 3.1.

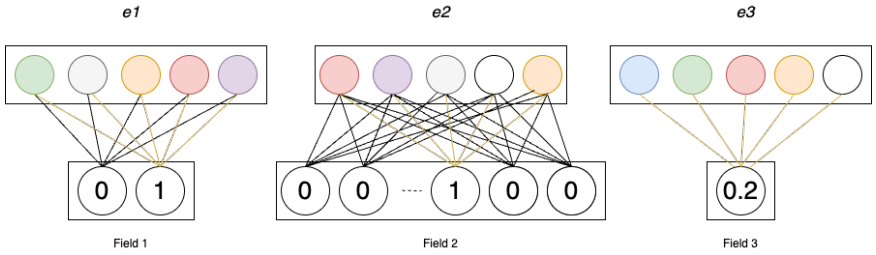

Figure 3.1: An example transformation from field vectors to feature embeddings is shown, where the embedding dimension is set to 5 in this example.

3.2 Base Models

3.2.1 DNN

A Deep Neural Network (DNN), also known as a Multi-Layer Perceptron (MLP), is widely used not only in CTR prediction models but across nearly every domain in the deep learning field. In this study, all three base models incorporate a deep network as a component in their implementations. A DNN consists of several fully connected feed-forward layers. The advantage of deep networks lies in their ability to effectively capture high-order feature interactions, as all features interact with each other during the feed-forward process. The overall architecture of a DNN model is illustrated in Figure 3.2.

As shown in Figure 3.2, an instance with sparse features, transformed into its low-dimensional feature embeddings, is fed into the feed-forward layers. The output of the embedding layer is concatenated into a single long vector for the input, as given in Equation 3.1:

$$a_0 = \text{concatenation}([e_1, e_2, \ldots, e_m]) \tag{3.1}$$

a_0 is fed into the model, and the below forward process is executed for the first layer:

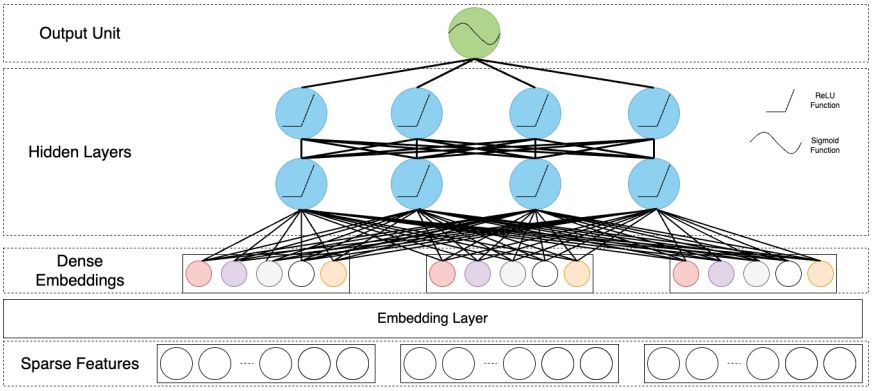

Figure 3.2: Example architecture of DNN model

$$a_1 = \sigma\left(W_0 a_0 + b_0\right) \tag{3.2}$$

where:

- $a_0 \in \mathbb{R}^{d_0}$ is the input feature vector.
- $a_1 \in \mathbb{R}^{d_0}$ is the output of the first hiden layer.
- $W_0 \in \mathbb{R}^{d_0 \times d_1}$ is the weighting matrix.
- M is the number of feature fields.
- D is the embedding dimension.
- $b_0 \in \mathbb{R}^{d_1}$ is the bias term.
- d_1 is the output dimension of first hidden layer.

and d_0 is equal to $M \times D$, d_1 is the output dimension of first hidden layer. Finally, σ is the activation function which allow DNN model to capture nonlinear relationships between features, and we set this as ReLU and the function equation is given in Equation 3.3:

$$f(x) = \max\left(0, x\right) \tag{3.3}$$

The equation 3.2 can be applied to for all layers in the deep network. The generalized form of the equation is given in Equation 3.4:

$$a_{l+1} = \sigma\left(W_l a_l + b_l\right) \tag{3.4}$$

where l is the depth of the layer. After the hidden layers, a dense feature vector is achieved to be fed into final fully connected layer to get the prediction for the instance with the below Equation 3.5:

$$\hat{y}_{DNN} = \sigma\left(W_L a_L + b_L\right) \tag{3.5}$$

where $W_L \in \mathbb{R}^{L\times 1}$, L is the number of hidden layers, and σ is the activation function used as Sigmoid because this function produces outputs which relies between $(0, 1)$. The Sigmoid function is given in Equation 3.6

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.6}$$

Finally, the $\hat{y}_{DNN}$ will be a scalar value which relies between $(0, 1)$ that can be used as CTR value for the given instance.

3.2.2 DeepFM

DeepFM is the second base model used in our study and consists of two components: the Deep component and the FM component. The Deep component is essentially a DNN model, with an architectural design identical to the DNN model described in Section 3.2.1. The primary contribution of DeepFM to the CTR prediction field is its second component, the Factorization Machines (FM). As discussed in Section 3.2.1, while DNNs are highly effective at learning high-order feature interactions, they may lose some important low-order feature interactions due to the complex interactions within each layer. The overall architecture of the FM component is illustrated in Figure 3.3.

The FM model can theoretically capture both first-order and second-order feature interactions. The mathematical formulation for the FM component is given in Equation 3.7

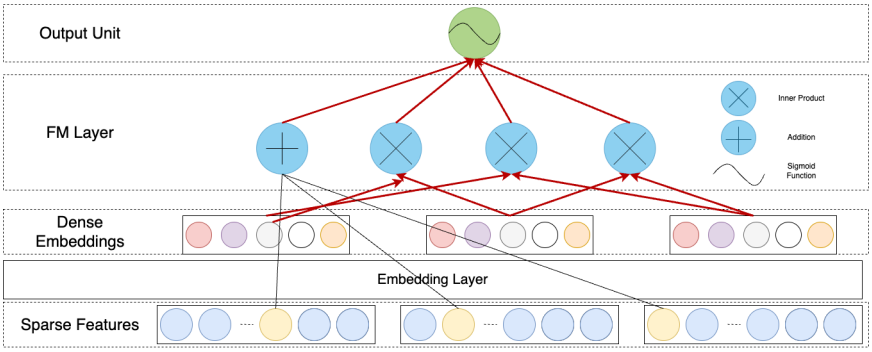

Figure 3.3: Overall architecture of FM model

$$\hat{y}_{FM} = W_0 + \sum_{i=1}^{M} W_i x_i + \sum_{i=1}^{M} \sum_{j=i+1}^{M} e_i e_j \tag{3.7}$$

where:

- $\hat{y}_{FM}$ is the predicted output of FM.
- W_0 is the global bias.
- W_i are the weights of the individual features.
- e_i and e_j are the feature embeddings for the interactions.
- x_i and x_j are the feature values.

The equation can be separated into two parts: the addition part and the inner product part. The addition part, $W_0 + \sum_{i=1}^{M} W_i x_i$ is responsible for learning first-order feature interactions. In this part, each individual feature value is multiplied by its corresponding weight, and the summation of these values, along with the global bias, represents the first-order feature effects for the instance. The inner product part, $\sum_{i=1}^{M} \sum_{j=i+1}^{M} e_i e_j$, captures second-order feature interactions. The summation of the inner products of each pair of feature embeddings represents the second-order feature interaction effects for the instance.

The Deep and FM components each produce a one-unit output, and the training is a joint operation, ensuring that all the parameters of both components are updated together. The final prediction for the CTR value for an instance is calculated as shown in Equation 3.8

$$\hat{y}_{DeepFM} = \sigma\left(\hat{y}_{FM} + \hat{y}_{DNN}\right) \tag{3.8}$$

where σ is the Sigmoid function to be sure that the predicted CTR value $\hat{y}_{DeepFM} \in (0, 1)$. An important note is although the DNN model has a Sigmoid function at the end of our formulation for $\hat{y}_{DNN}$ calculation, the Deep component of DeepFM does not have this activation function at the end of its architecture.

3.2.3 DCN

The Deep & Cross Network (DCN) is the last conventional architecture we selected as one of our base models. DCN is quite similar to DeepFM in terms of the parallel training of high-order and low-order feature interaction extractions. It consists of two parallel components: the Deep and Cross components, much like DeepFM. The Deep component is identical to that in DeepFM, except for the output size. Unlike in DeepFM, the output length of the Deep component is not 1 but rather matches the length of the last hidden layer unit in the Deep component. The overall architecture of DCN is illustrated in Figure 3.4.

The FM and Cross Network components are similar in their underlying approach to low-order feature interaction extraction. The key difference between the two is that FM can only capture second-order feature interactions, whereas the Cross Network can learn interactions of unlimited degrees. The Cross Network contains several cross layers, which are the essential parts of feature interaction extraction. The cross layer equation can be expressed as shown in Equation 3.9

$$a_{l+1} = a_0 {a_l}^{\top} W_l + b_l + a_l \tag{3.9}$$

where:

- $a_0 \in \mathbb{R}^{MD}$ is the original input feature embedding vector.
- $a_l \in \mathbb{R}^{MD}$ is the output of feature embedding vector at layer l.
- $W_l \in \mathbb{R}^{D}$ is the weight vector at layer l.

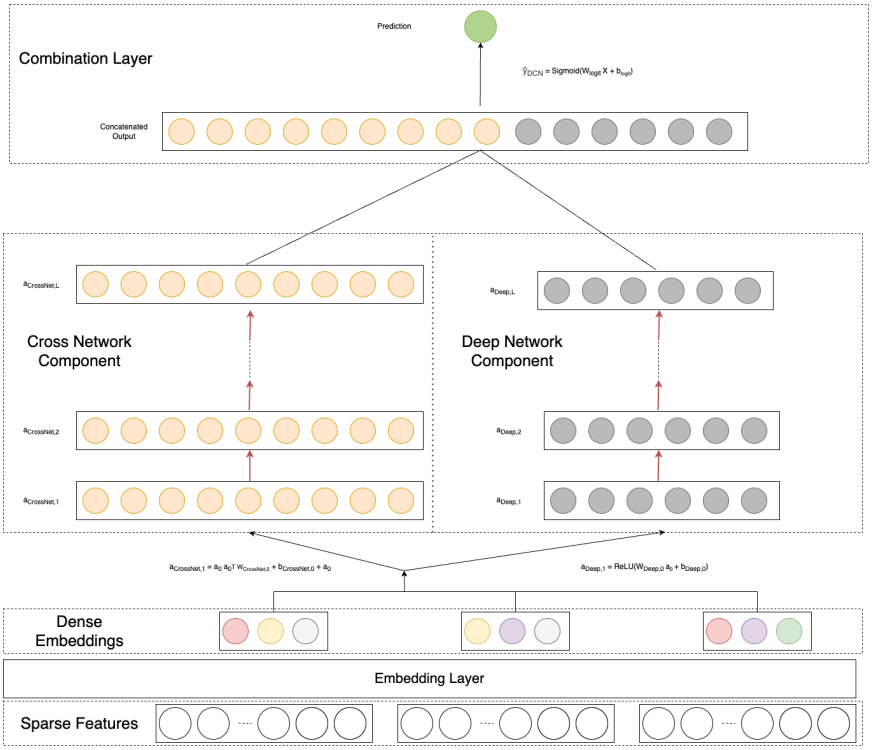

Figure 3.4: Overall architecture of DCN model

- $b_l \in \mathbb{R}^D$ is the bias term at layer l.

The degree of the feature interaction can be determined by the rank of the cross layer. Therefore, to capture the second-order feature interactions, $L = \max(l)$ should be set as 1 because it consists of two feature vector interaction. A generalized rule is the highest ranked feature interaction for a given L is $L + 1$. Each cross layer also adds back to its input feature vector to not lose important information during the crossing process. The visualization of layer $l + 1$ formulation for a cross layer can be seen in Figure 3.5.

In DCN, the training of the Deep and low order feature interaction components (Cross Network) is parallel and joint, similar to DeepFM. However, the outputs of the two components are not scalar values but vectors of the component outputs. Therefore, DCN includes a combination layer at the top of its structure,where the outputs from the two components are concatenated into a single vector and then fed into a standard logits layer. The prediction formula can be expressed as given in Equation 3.10

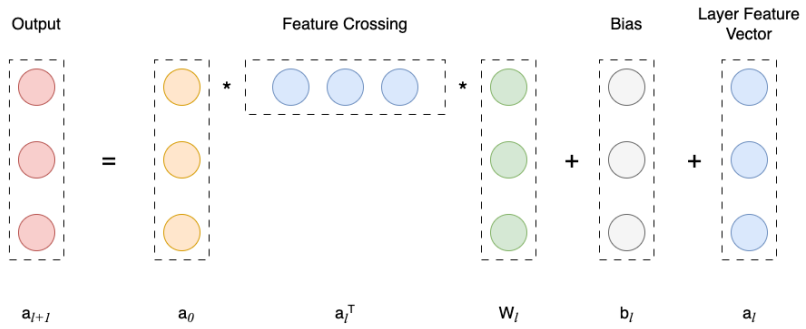

Figure 3.5: Visualization of a cross layer formulation for layer $l + 1$

$$\hat{y}_{\text{DCN}} = \sigma\left(W_{logit}\text{concatenate}([a_{\text{CrossNetwork},(L_{\text{CrossNetwork}}+1)}, a_{\text{Deep},(L_{\text{Deep}}+1)}]) + b_{logit}\right) \tag{3.10}$$

where:

- $a_{\text{CrossNetwork},(L_{\text{CrossNetwork}}+1)} \in \mathbb{R}^{MD}$ is the output of last cross layer in Cross Network Component.
- $a_{\text{Deep},(L_{\text{Deep}}+1)} \in \mathbb{R}^{H}$ is the output of last hidden layer in the Deep Component.
- $W_{logit} \in \mathbb{R}^{(MD+H)}$ is the weight vector at prediction logit layer.
- $b_{logit} \in \mathbb{R}^{1}$ is the bias term at prediction logit layer.
- $\hat{y}_{\text{DCN}}$ is the predicted output of DCN.

and σ is Sigmoid as in other base models. $L_{\text{CrossNetwork}}$ is the number of cross layers in DCN and L_{Deep} is the number of hidden layers in DCN's Deep Component.

3.3 Comparison of Base Models

All three base models were chosen because they are the most robust and widely used models in the field of CTR prediction [11]. There are both similarities and differences among these models. The primary similarity is that all three models include a deep component. However, the DNN is purely a deep component without any interaction

extraction component and can be described as a one stream model. On the other hand, DCN and DeepFM contain interaction extraction components—Cross Network and FM, respectively—and can be described as two stream models. Both DNN and DeepFM directly produce a single unit prediction, whereas DCN has a combination output layer to produce a single unit prediction. The generalized architectures for each model can be seen in the Figure 3.6, illustrating the differences and similarities.

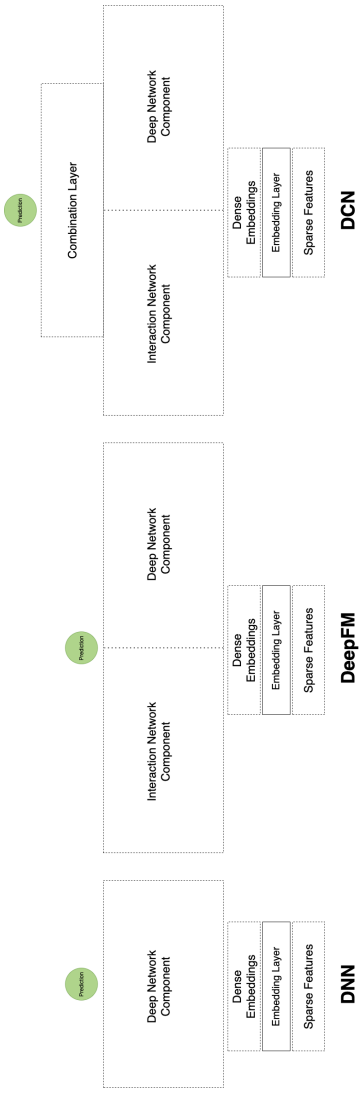

Figure 3.6: Generalized architectures of base models

3.4 Squeeze and Excitation Networks

Our attention mechanism is based on the Squeeze and Excitation Networks (SENet) module. SENet has proven its success in the computer vision domain by capturing the relationships between the channels of convolutional features. The first use of the SENet layer in the CTR prediction task was implemented in the FiBiNet model. The FiBiNet model differs slightly from our base models, as it includes two identical Bilinear Interaction layers with two different inputs: the first input is feature embeddings, and the second input is feature embeddings updated with SENet attention. The two output vectors of the Bilinear Interaction layers are concatenated and then used to calculate the prediction value.

In the study, researchers found that the SENet layer enhances the performance of the bilinear interaction layer by highlighting important feature interaction weights. The SENet layer consists of three components: Squeeze, Excitation, and Reweight. In the Squeeze part, a summary statistic of each field embedding is calculated to represent the field, and various summary statistic methods, such as max or mean pooling, can be used. FiBiNet uses mean pooling as the summary statistic, while FiBiNet++ (a low-parameter, better-performing version of FiBiNet) uses both mean and max pooling within each group that is split for each field embedding.

The Excitation component learns the weight of each field embedding based on the summary statistics produced during the Squeeze step. Finally, the Reweight step is performed by multiplying the excitation vector with the original field embedding to produce an updated vector.

Following the groundbreaking performance of FiBiNet, several novel models have incorporated the SENet mechanism. For example, FAT-DeepFFM uses a 1x1 convolution layer as the pooling operation in the Squeeze part and names the model CENet (Compose Excitation Network). Another model, ECANFM, uses a 1x1 convolution layer in its Excitation part instead of Squeeze, with mean pooling as the summary statistic for the fields. Finally, the MMBATTN (Max-Mean and Bitwise Attention) model implements three SENet-like attention mechanisms in a joint manner. The summary statistics are mean and max for the vector-wise attention mechanism, and

there is also a bitwise attention mechanism in the joint architecture of the attention mechanism.

The steps of the SENet layer will be elaborated on in the upcoming subsections and overall architecture of SENet layer can be seen in the Figure 3.7.

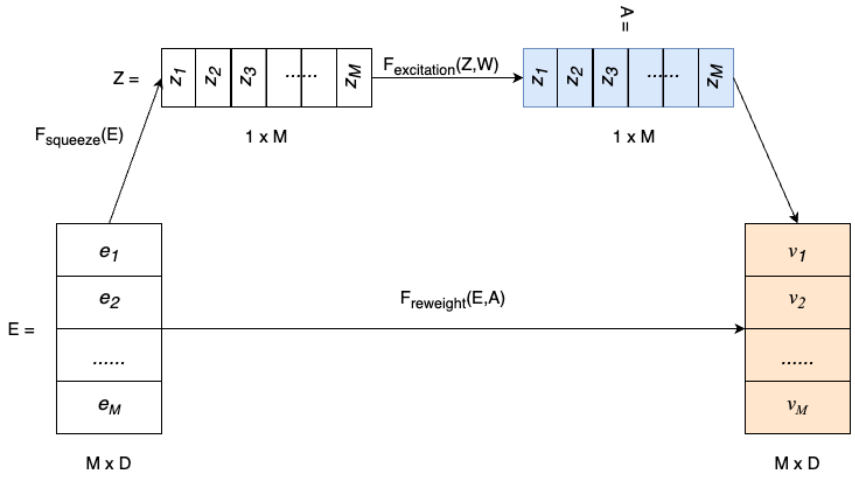

Figure 3.7: Overall architecture of SENet layer

3.4.1 Squeeze

In the Squeeze step, the 'summary statistics' of each field embedding are calculated using any pooling mechanism. The options are all functions that can summarize a vector distribution. The most common are mean pooling and max pooling, but in our study, we also implemented Mean-Max joint pooling and Mean-Standard Deviation (Std) joint pooling, as these can better capture the distribution of the field embedding. The original field embedding vector is $E = \{e_1, \ldots, e_M\}$ where M is the number of feature fields, and each field embedding has D dimension vector which is the embedding dimension. The squeeze layer outputs a statistic vector $Z = \{z_1, \ldots, z_M\}$. z_i is a scalar value which stores the global information for the i-th field and where $i \in [1, ..., M]$. The proposed pooling method calculations will be elaborated on in Section 4.

Additionally, if we use bit-wise attention, we do not have a squeeze step, as all bits in the field embedding are represented individually. However, we concatenate the field

embeddings into one vector to employ bit-wise excitation.

3.4.2 Excitation

The Excitation step is used to calculate the weight of the field embedding using the summary statistic vector Z. This step consists of two fully connected nonlinear layers: the first layer is a dimensionality-reduction layer, and the second layer restores the dimension to the input dimension. The output of the Excitation step, which is the weight of the field embedding, can be formulated as follows in Equation 3.11

$$A = F_{excitation}(\mathbf{Z}) = \sigma_2(W_2\sigma_1(W_1 Z)) \tag{3.11}$$

where:

- $\mathbf{Z} \in \mathbb{R}^{1\times M}$ is the summary statistic vector.
- $W_1 \in \mathbb{R}^{M\times \frac{M}{r}}$ is the weight matrix at the dimension reduction layer.
- $W_2 \in \mathbb{R}^{\frac{M}{r}\times M}$ is the weight matrix to restore the dimension layer.
- σ_1 and σ_2 are the activation functions for each layer, respectively.
- $\mathbf{A} \in \mathbb{R}^{1\times M}$ is the field weight vector.

If the attention mechanism is bit-wise instead of vector-wise, the equation can be formulated as given in Equation 3.12

$$A = F_{excitation}(\mathbf{Z}) = \sigma_2(W_2\sigma_1(W_1 Z)) \tag{3.12}$$

where:

- $\mathbf{Z} \in \mathbb{R}^{1\times (MD)}$ is the concatenated field embedding vector.
- $W_1 \in \mathbb{R}^{MD\times \frac{MD}{r}}$ is the weight matrix at the dimension reduction layer.
- $W_2 \in \mathbb{R}^{\frac{MD}{r}\times MD}$ is the weight matrix to restore the dimension layer.

- σ_1 and σ_2 are the activation functions for each layer, respectively.
- $\mathbf{A} \in \mathbb{R}^{1 \times MD}$ is the field weight vector.

3.4.3 Re-weight

The Reweight step, also referred to as rescale in the original paper, is the last step in the SENet attention mechanism. The objective is to update the field embeddings according to the weights calculated in the Excitation step. The original field embedding E and the field weight vector A are field-wise (vector-wise) muliplied to create the updated embeddings $V = \{v_1, \ldots, v_M\}$. Reweight step can be formulated as given in Equation 3.13.

$$V = F_{reweight}(\mathbf{E}, \mathbf{A}) = [e_1 \cdot a_1, \ldots, e_M \cdot a_M] = [v_1, \ldots, v_M] \tag{3.13}$$

where:

- $E = \{e_1, \ldots, e_M\}$ is the original field embedding vector.
- $A = \{a_1, \ldots, a_M\}$ is the field weight vector which is the output of excitation layer.
- $V = \{v_1, \ldots, v_M\}$ is the updated field embedding vector.

If the attention mechanism is bit-wise attention, then the formulation has slight changes as given in Equation 3.14.

$$V = F_{reweight}(\mathbf{E}, \mathbf{A}) = [e_1 \cdot a_1, \ldots, e_M D \cdot a_M D] = [v_1, \ldots, v_M D] \tag{3.14}$$

where:

- $E = \{e_1, \ldots, e_M D\}$ is the original concatenated field embedding vector.
- $A = \{a_1, \ldots, a_M D\}$ is the field weight vector which is the output of excitation layer.

- $V = \{v_1, \ldots, v_M D\}$ is the updated concatenated field embedding vector.

An important note for bit-wise attention is that, since bit-wise attention uses the concatenated embeddings vector and outputs a concatenated vector, the output vector is transformed back into its original shape after the reweight step so that each field has its own embedding again. That is, $V \in \mathbb{R}^{MD}$ is transformed into $V \in \mathbb{R}^{M \times D}$.

CHAPTER 4

STUDIED ANALYSIS APPROACH

In this chapter, we introduce the attention mechanisms that we experimented on considered models as elaborated on Section 3, as well as the hyperparameter candidates, which include the reduction ratio for the squeeze part, the application of layer normalization, and feeding the updated embeddings to the interaction layer.

4.1 Studied Attention Mechanisms

We have studied several well-known statistical attention mechanisms and an additional one that, to our knowledge, no study has ever tried. FiBiNet [14] is the first model to use the SENet [15] module in the CTR prediction task. In FiBiNet, the squeeze component establishes the summary statistic vector using mean pooling instead of max pooling, which is the pooling function of the original SENet module for computer vision tasks. They stated that mean pooling performs better than max pooling, but the performance results are not provided in the paper. However, in FiBiNet++ [24], they improved the model's performance by implementing an enhanced SENet layer, SENet+. Instead of retrieving a summary statistic for each field embedding, they split each field embedding into the same number of groups, calculate the summary statistic for each group using max and mean pooling, and concatenate the resulting summary vectors of each group as the output for the squeeze part. The aim of implementing both mean and max pooling is to provide more useful information about field embeddings. Additionally, MMBAttn [29] also implemented mean and max pooling in their application, but instead of concatenating them, they summed the two summary statistic vectors as the aggregation method. Although the performance

of mean pooling, max pooling, and mean-max pooling with summation on the DNN model for the Criteo and Avazu datasets is shared, the performance of concatenation on the DNN model for the Criteo and Avazu datasets, as well as the same summary statistics method on different models and datasets, is not shared. As stated, mean-max pooling with summation is the best-performing attention mechanism among solely mean and max attention mechanisms on the DNN model; hence, MMBAttn includes a component that uses mean-max attention. The other component of MMBAttn is bit-wise attention, which is based on bit-wise relations instead of vector-wise relations as in summary statistics. Each bit is used in the attention mechanism instead of a representative vector. Finally, we propose experimenting with the mean-standard deviation attention mechanism, as the aim is to provide meaningful information about the distribution of the field embedding. The final attention mechanisms that are experimented with are listed below:

- Mean Attention Mechanism: Uses only mean pooling in the squeeze part.
- Max Attention Mechanism: Uses only max pooling in the squeeze part.
- Mean-Max Concatenation Attention Mechanism: Applies mean and max pooling with concatenation as the aggregation method in the squeeze part.
- Mean-Max Summation Attention Mechanism: Applies mean and max pooling with summation as the aggregation method in the squeeze part.
- Mean-Std Concatenation Attention Mechanism: Uses mean and standard deviation pooling with concatenation as the aggregation method in the squeeze part.
- Mean-Std Summation Attention Mechanism: Uses mean and standard deviation pooling with summation as the aggregation method in the squeeze part.
- Bitwise Attention Mechanism: No squeeze part, as there is no summary statistic for the vectors.

We elaborate on the dimensions of the vectors and the operations in the steps followed by each attention mechanism in the subsections.

4.1.1 Mean Attention

The Mean Attention Mechanism uses mean pooling as the summary statistic vector in the squeeze part, and the remaining steps (excitation and reweighting) are the same as in a regular SENet layer. In the excitation part, the reduction ratio is set to 1, 3, and 5 in the experiments. The calculation for the squeeze part in the Mean Attention Mechanism can be formulated as given in Equation 4.1.

$$z_i = F_{squeeze}(\mathbf{e_i}) = \frac{1}{D}\sum_{j=1}^{D} e_{ij} \tag{4.1}$$

where:

- $\mathbf{e_i} = [e_1, e_2, \ldots, e_D]$ is the field embedding vector for i-th field.
- D is the embedding dimension.
- z_i is the mean of the i-th field embedding vector.

4.1.2 Max Attention

The Max Attention Mechanism uses max pooling instead of mean pooling, as in the Mean Attention Mechanism, while the other parts remain the same as in the Mean Attention. The z_i scalar value, which is the i-th field summary statistic under max pooling, can be formulated as given in Equation 4.2.

$$z_i = F_{squeeze}(\mathbf{e_i}) = \max_{j=1,\ldots,D}\{e_{ij}\} \tag{4.2}$$

where:

- $\mathbf{e_i} = [e_{i1}, e_{i2}, \ldots, e_{iD}]$ is the field embedding vector for the i-th field.
- D is the embedding dimension.
- z_i is the maximum value of the i-th field embedding vector.

4.1.3 Mean-Max Concatenation Attention

The Mean-Max Concatenation Attention is a joint summary statistic vector constructed to provide more information about the field embedding vectors. The squeeze part contains two separate squeeze functions, which are the same as in the Mean Attention and Max Attention mechanisms. However, Mean-Max Concatenation differs in the excitation part, as the two summary statistic vectors for mean and max functions are concatenated and used as input for the excitation part. This idea is similar to FiBiNet++, where the excitation part takes a vector of different size and produces an output of the size of the field embedding vector. The excitation step can be formulated as given in Equation 4.3.

$$A = F_{excitation}(\mathbf{Z_{concat}}) = W_2\sigma(W_1 Z_{concat}) \tag{4.3}$$

where:

- $\mathbf{Z_{concat}} \in \mathbb{R}^{1\times 2M}$ is the concatenated summary statistic vector which is $concat(Z_{mean}, Z_{max})$.
- $W_1 \in \mathbb{R}^{2M\times \frac{2M}{r}}$ is the weight matrix at the dimension reduction layer.
- $W_2 \in \mathbb{R}^{\frac{2M}{r}\times M}$ is the weight matrix to restore the dimension layer.
- σ is the activation function which is ReLU.
- $\mathbf{A} \in \mathbb{R}^{1\times M}$ is the field weight vector.
- $\mathbf{r} \in [1, 3, 5]$ is the reduction ratio

4.1.4 Mean-Max Summation Attention

Another aggregation method for mean pooling and max pooling operations is summing the summary statistic vectors instead of concatenating them. Although FiBiNet++ concatenates these two vectors, some models in the CTR prediction field sum these two vectors. Summation can lead to a loss of information, but it may also increase the generalization of the summary vectors. The mathematical difference between the two aggregation methods (concatenation and summation) is that there will

be two excitation layers in summation, whereas there is only one layer in concatenation. The excitation steps can be formulated as given in Equation 4.4.

$$\begin{aligned} A_{\text{mean}} &= F_{\text{excitationmean}}(\mathbf{Z}\text{mean}) = W_{2_{\text{mean}}}\sigma(W_{1_{\text{mean}}}\mathbf{Z}_{\text{mean}}) \\ A_{\text{max}} &= F_{\text{excitationmax}}(\mathbf{Z}\text{max}) = W_{2_{\text{max}}}\sigma(W_{1_{\text{max}}}\mathbf{Z}_{\text{max}}) \\ A &= A_{\text{mean}} + A_{\text{max}} \end{aligned} \tag{4.4}$$

where:

- $\mathbf{Z_{mean}} \in \mathbb{R}^{1\times M}$ is the summary statistic vector for mean pooling.
- $\mathbf{Z_{max}} \in \mathbb{R}^{1\times M}$ is the summary statistic vector for max pooling.
- $W_{1_{mean}} \in \mathbb{R}^{M\times\frac{M}{r}}$ is the weight matrix at the dimension reduction layer for mean pooling.
- $W_{1_{max}} \in \mathbb{R}^{M\times\frac{M}{r}}$ is the weight matrix at the dimension reduction layer for max pooling.
- $W_{2_{mean}} \in \mathbb{R}^{\frac{M}{r}\times M}$ is the weight matrix to restore the dimension layer for mean pooling.
- $W_{2_{max}} \in \mathbb{R}^{\frac{M}{r}\times M}$ is the weight matrix to restore the dimension layer for max pooling.
- σ is the activation functions which are ReLU.
- $\mathbf{A_{mean}} \in \mathbb{R}^{1\times M}$ is the field weight vector for mean pooling.
- $\mathbf{A_{max}} \in \mathbb{R}^{1\times M}$ is the field weight vector for max pooling.
- $\mathbf{A} \in \mathbb{R}^{1\times M}$ is the field weight vector.
- $\mathbf{r} \in [1, 3, 5]$ is the reduction ratio.

4.1.5 Mean-Std Concatenation Attention

Another method to compress information from the feature vectors is using standard deviation pooling. However, without an auxiliary pooling operation, standard deviation alone will not be sufficient for information storage. Therefore, mean pooling and

standard deviation pooling can be used together to create summary statistic vectors, as some computer vision and voice recognition models have done [30, 31]. However, to our knowledge, there are no models that have used standard deviation for the CTR prediction task. The only difference between Mean-Std Concatenation and Mean-Max Concatenation attention is that the squeeze layer uses standard deviation pooling instead of max pooling. The remaining parts are the same as in Mean-Max Concatenation attention in terms of parameter sizes and mathematical operations. The z_i scalar value, which is the i-th field summary statistic under standard deviation pooling, can be formulated as given in Equation 4.5.

$$z_i = F_{squeeze}(\mathbf{e_i}) = \sqrt{\frac{1}{D}\sum_{j=1}^{D}(e_{ij} - \mu)^2} \tag{4.5}$$

where:

- $\mathbf{e_i} = [e_{i1}, e_{i2}, \ldots, e_{iD}]$ is the field embedding vector for the i-th field.
- D is the embedding dimension.
- μ is the mean of the field embedding vector $\mathbf{e_i}$.
- z_i is the standard deviation of the i-th field embedding vector.

4.1.6 Mean-Std Summation Attention

Similar to Mean-Max Summation, we also conducted experiments with Mean-Std Summation attention. The only difference from Mean-Max Summation is that, instead of the Max pooling function, we use the Standard Deviation pooling function. Moreover, the formulations are exactly the same as in Mean-Max Summation, except for the Max pooling function.

4.1.7 Bitwise Attention

Bitwise Attention is the final proposed attention mechanism in our study. The aim of Bitwise Attention is to use all the bits in the excitation step to calculate the weight

of each bit, instead of using vector-wise weighting. Bitwise Attention does not have a squeeze part, as there is no pooling mechanism. Additionally, the field embedding vector is flattened and undergoes an MLP to reduce the dimension and then restore it to the original size in the excitation part. Another difference is in the reweighting part, where the attention value, which has the same dimension as the total dimension of the feature embedding, is elementwise multiplied with the field embedding to obtain the updated feature embeddings. In the final step, the concatenated updated vector is reshaped into field vectors that have the same dimension as the input at the beginning. The excitation part of the Bitwise Attention can be formulated as given in Equation 4.6.

$$A = F_{excitation}(\mathbf{E_{concat}}) = W_2\sigma(W_1 E_{concat}) \tag{4.6}$$

where:

- $\mathbf{E_{concat}} \in \mathbb{R}^{1\times MD}$ is the concatenated field embedding vector.
- $W_1 \in \mathbb{R}^{MD\times \frac{MD}{r}}$ is the weight matrix at the dimension reduction layer.
- $W_2 \in \mathbb{R}^{\frac{MD}{r}\times MD}$ is the weight matrix to restore the dimension layer.
- σ is the activation function which is ReLU.
- $\mathbf{A} \in \mathbb{R}^{1\times MD}$ is the field weight vector.
- $\mathbf{r} \in [1, 3, 5]$ is the reduction ratio

The reweighting part of the bitwise attention can be formulated as below:

$$\mathbf{V} = \mathbf{A} \cdot \mathbf{E}_{\text{concat}} \tag{4.7}$$

where:

- $\mathbf{V} \in \mathbb{R}^{1\times MD}$ is the updated concatenated field embedding vector

Then, V is reshaped into field vectors which has the dimension as $\mathbb{R}^{M\times D}$.

4.2 Studied Architectural Designs

In this section, the design choices of the attention mechanisms are detailed. In our experiments, we analyzed the performance of applying the layer normalization algorithm after the attention mechanism and examined the effect of feeding only the Deep part with the attention mechanism instead of both the Deep and Interaction parts.

4.2.1 Layer Normalization

Layer Normalization (Layer Norm) is a crucial normalization method for stabilizing training in the machine learning domain [32]. It is also a technique to reduce overfitting, as it normalizes the input across features during each training phase. Several studies in the literature have shown that utilizing layer normalization after the attention mechanism leads to better performance in CTR prediction tasks [24, 33]. In this thesis, the aim is to investigate the effects of layer normalization on attention mechanisms as an additional operation after the attention mechanism. Therefore, layer normalization is applied to the updated field vectors before feeding these vectors to the base models. Given a vector $\mathbf{V} = [v_1, v_2, \ldots, v_d]$, Layer Normalization formulation is given in Equations 4.8 through Equations 4.11.

1. Compute the mean μ of the vector:

$$\mu = \frac{1}{d}\sum_{i=1}^{d} v_i \tag{4.8}$$

2. Compute the variance σ^2 of the vector:

$$\sigma^2 = \frac{1}{d}\sum_{i=1}^{d}(v_i - \mu)^2 \tag{4.9}$$

3. Normalize the vector:

$$\hat{v}_i = \frac{v_i - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{4.10}$$

where ϵ is a small constant added for numerical stability and it is 1e-05 as in PyTorch default setting.

4. Apply the learned scale γ and shift β:

$$y_i = \gamma \hat{v}_i + \beta \tag{4.11}$$

The final output vector $\mathbf{Y} = [y_1, y_2, \ldots, y_d]$ is the result of the Layer Normalization.

4.2.2 No Feeding on Interaction Component

As it is discussed in Section 3.2, DeepFM and DCN have two components: the Deep component and the Low-Order Interaction component. For DeepFM, the low-order interaction component is the FM component, and for the DCN model, it is the Cross Network. The objective of low-order feature interaction components is to capture a bounded degree of feature interactions using field embedding vectors. Therefore, the effects of updating the field embedding vectors with attention mechanisms on low-order feature interaction components should be considered. There are a few studies that feed the updated feature vector with the attention mechanism only to the Deep component, but not to the interaction component [23]. However, to our knowledge, there are no studies that compare the performance with or without feeding the interaction component. Therefore, we conduct our experiments with the options of feeding or not feeding the interaction component for DeepFM and DCN, but not for DNN, as it only has the Deep component.

4.2.3 Experimental Choices for Architectural Desing

In this subsection, we explain the experimental choices for each base model. As discussed above, the updated field embedding vectors with attention mechanisms can either be fed or not fed into the interaction component for DeepFM and DCN, resulting in two options for the experiments on 'No Feeding on Interaction Component.' Additionally, layer normalization can be applied or not applied after the attention mechanisms for all base models, which also leads to two more options. Therefore, there are four experimental design options for DeepFM and DCN, and two for DNN. The overall structures for these four options can be seen in Figure 4.1

As explained above, DCN and DeepFM models have all 4 experimentation options,

but DNN only has option 3 and option 4 as it does not have an interaction component.

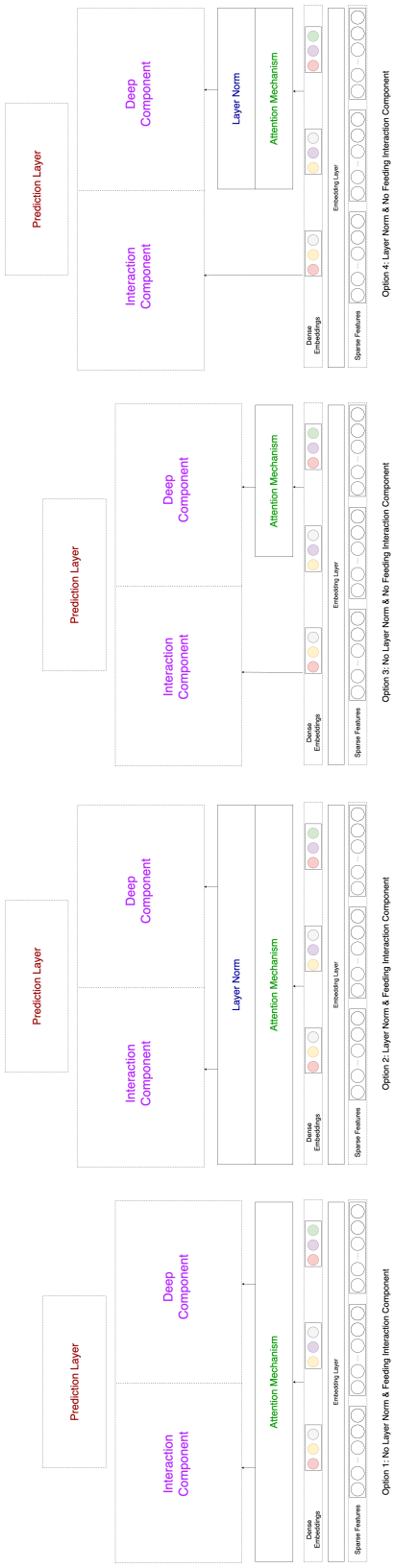

Figure 4.1: Overall structures of experimentation settings

CHAPTER 5

EXPERIMENTAL RESULTS

This section elaborates on the experimental results of different attention mechanism under different architectural setups for four different datasets on the three most common models for CTR prediction. By these experiments, we aimed to answer the following research questions:

- `(RQ1)`: Do the attention mechanisms we experimented provide improvements over the mostly known conventional models?
- `(RQ2)`: Can we make general inferences about the performance of attention mechanisms across different datasets and models?
- `(RQ3)`: Does layer normalization improve the performance of models with attention mechanisms?
- `(RQ4)`: Does the attention mechanism perform better without feeding into the interaction layer?

5.1 Experiment Setup

5.1.1 Datasets

Three of the most widely used datasets have been employed and the splits for training, validation and test sets. For the Criteo, MovieLens, and Frappe datasets, the data are randomly split into 7:1:2 ratios for the training, validation, and test sets. On the other hand, the Avazu dataset is split into a 7:2:1 ratio for the training, validation, and test sets, as in the BARS open-benchmark datasets for CTR prediction [11]. Moreover,

Table 5.1: Summary Statistics of Datasets

Dataset	# Instances	# Fields	# Categorical Fields
Criteo	45M	39	26
Avazu	40M	22	22
MovieLens	2M	3	3
Frappe	289K	10	10

no preprocessing steps are applied, as the goal of these models is to learn the hidden features and interactions without any feature engineering. These four datasets have different number of categorical and numerical variables as the feature fields. Moreover, these datasets were selected because they represent different characteristics.

Criteo [34] is a display ad dataset that contains ad impressions as instances and binary labels as ad clicked or not. This dataset is the largest dataset in terms of number of examples and 39 feature fields which 26 of them are categorical fields and rest of them are numerical. **Avazu** [35] is a mobile advertisement click dataset and it contains 22 categorical fields that store user and advertisement information. **Movielens** [36] dataset is a well-known recommender systems dataset that contains 3 categorical fields which are userID, movieID and tagID and label is whether corresponding user has tagged the movie. **Frappe** [37] dataset stores logs of app usages for the users who have different attributes such as country, weather, daytime with other user specific information such as userID, appID. In the dataset, there are 10 categorical features. The label represents if the app user made a session under the given context features. The summary statistics of the datasets are listed in Table 5.1

5.1.2 Evaluation Metric

We used **AUC (Area Under the ROC Curve)** as our metric for model and architectural design evaluation. AUC is the most common metric for CTR prediction model

evaluation in the literature. AUC can be measured by the area under the ROC curve. To establish a ROC curve, TPR (True Positive Rate) and FPR (False Positive Rate) should be calculated for various thresholds over the prediction probabilities. This metric measures the probability that a randomly chosen positive example has a higher rank than randomly chosen negative example. It is highly important to note that a 0.001-point uplift for AUC metric is accepted as a significant improvement in this field [1, 2, 3, 4].

Logloss is used for evaluation and optimization within the training process. Logloss, which is also known as binary cross entropy, is mainly used for binary classification tasks. One of the most important differences between Logloss and AUC is that AUC does not depend on the positive label ratio of the dataset. Logloss formula can be formulated in Equation 5.1

$$\text{Logloss} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right] \tag{5.1}$$

In the equation, y_i and $\hat{y}_i$ are the values for the true and predicted labels for the corresponding instance i and N is the total instances in the training set. The objective in our binary classification model is to decrease this number as much as possible while training.

5.1.3 Architecture Comparison

In this study, we evaluated various attention mechanisms, including mean, max, mean-max summation, mean-max concatenation, mean-std summation, mean-std concatenation, and bitwise attentions. These mechanisms were employed to three different models which are vanilla MLP (DNN), DeepFM and DCN. Each model was tested with different architectural hyperparameter settings, incorporating layer normalization and the inclusion or exclusion of an interaction layer. Only the DNN model does not have an option for feeding interaction layer because the model does not possess an interaction layer.

5.1.4 Implementation Details

For all three baseline model, we used the reported best hyperparameter settings on BARS open-benchmark datasets for CTR prediction and all the attention mechanism experiments for each model and dataset pairs are built on these hyperparameter settings. We executed our experiments by using FuxiCTR [38] library which is an open source CTR prediction library. To ensure a fair comparison, we reported our experimental results instead of using the reported AUC results for the baseline models. In these reported best settings, all the hyperparameters for each model are kept same, except for embedding regularizers and dropout rates. Embedding dimension for each field is 10, MLP contains 3 hidden layers with shape of [400, 400, 400] and batch size is 4096. The number of cross network layer is set as 3. We followed the same optimization steps with BARS, the initial learning rate is set as 0.001 and an adaptive learning rate strategy is used, where the learning rate is divided by a factor of 10 each time the validation loss increases after an epoch. The model is trained for 100 epochs at maximum, with early stopping logic by using a patience parameter of 2 and the optimizer is chosen as Adam [39]. Adam optimizer is a Stochastic Gradient Descent optimizer type algorithm that is based on adaptive estimation of first and second order moments. The Adam optimizer updates the parameters using the given in Equations 5.2 through Equations 5.6.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{5.2}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{5.3}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{5.4}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{5.5}$$

$$\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{5.6}$$

where:

- m_t is the first moment estimate.
- v_t is the second moment estimate.

- $\hat{m}_t$ is the bias-corrected first moment estimate.
- $\hat{v}_t$ is the bias-corrected second moment estimate.
- θ_t are the parameters at time step t.
- α is the learning rate.
- β_1 and β_2 are the exponential decay rates for the moment estimates.
- g_t is the gradient at time step t.
- ϵ is a small constant for numerical stability.

The reduction ratio for attention mechanisms are experimented with 1, 3, 5 ratios and the best model according to the AUC score is considered for the comparisons. Each attention mechanism has 2 or 4 architectural design choice option as attention mechanism with layer normalization or not, and attention mechanism solely affects the DNN part of the model. All of these experiments conducted on a RTX 4000 GPU with CUDA version 11.7.

5.2 Results and Discussions

In this section, the experiment results and analyses will be shared and discussed.

5.2.1 Performances of Attention Mechanisms On Conventional Models (RQ1)

In the literature, the success of different attention mechanisms in CTR prediction task has been studied several times [20, 21, 4, 23, 14, 25, 26, 27]. However, in these studies, the success of the proposed architecture—which may contain components other than the attention mechanism—was generally examined in comparison to the baseline results, rather than focusing on the attention mechanism itself. Additionally, as our knowledge, no study has ever been conducted before on the behavior of attention mechanisms across different conventional models and the most commonly used datasets. To analyze the performances of attention mechanism, the experiment results of Option 1 settings in Figure 4.1 are being used in the tests. The tests are conducted

by comparing AUC value of the experiments and the base model AUC values. Therefore, the only difference between the two pairs is attention mechanism layer. The experimental datasets for the base and experiment values are given in Appendix A.1. In the experiments, the gradient flow problem was observed for Mean-Standard Summation Attention and Mean-Standard Concatenation Attention mechanisms on the DCN base model and Avazu dataset. These two cases are excluded in the analysis as they are accepted as outliers. Moreover, this problem is not observed when the layer normalization is activated and more details is provided in Section 5.2.3. To begin the initial examination of overall behavior, Table 5.2 shows the attention mechanisms that performed the best in each dataset and base model, as well as the percentage uplift and change observed in comparison to the base AUC value and experiment AUC value.

Table 5.2: Best performer attention mechanisms for each dataset and model pair. LN refers to Layer Normalization and ICF refers to Interaction Component Feeding.

DataSet	Base Model	LN	ICF	Attention Mechanism	Experimented Attention Mechanism AUC	Base Model AUC	Change	Percentage Uplift
Avazu	DCN	Yes	Yes	Max Attention	0.7654	0.7623	0.0031	0.4067%
Avazu	DNN	Yes	NA[1]	Mean Max Concatenation Attention	0.7648	0.7635	0.0013	0.1703%
Avazu	DeepFM	Yes	Yes	Mean Max Summation Attention	0.7644	0.7642	0.0002	0.0262%
Criteo	DCN	-	-	Bitwise Attention	0.8141	0.8137	0.0004	0.0492%
Criteo	DNN	-	NA[1]	Bitwise Attention	0.8140	0.8136	0.0004	0.0492%
Criteo	DeepFM	-	-	Bitwise Attention	0.8142	0.8137	0.0005	0.0614%
Frappe	DCN	Yes	-	Bitwise Attention	0.9846	0.9843	0.0003	0.0305%
Frappe	DNN	-	NA[1]	Mean Attention	0.9846	0.9844	0.0002	0.0203%
Frappe	DeepFM	Yes	Yes	Mean Max Summation Attention	0.9849	0.9844	0.0005	0.0508%
Movielens	DCN	-	-	Bitwise Attention	0.9695	0.9686	0.0009	0.0929%
Movielens	DNN	Yes	NA[1]	Mean Max Concatenation Attention	0.9686	0.9676	0.0010	0.1033%
Movielens	DeepFM	Yes	Yes	Max Attention	0.9686	0.9683	0.0003	0.0310%

For the Avazu dataset, the best-performing attention mechanisms were Max Attention for DCN and Mean-Max Attention mechanisms for DNN and DeepFM. Notably, all best-performing attention mechanisms were obtained when layer normalization was applied, with the updated weights being fed into the interaction component. Among all the best-performing experiments for each dataset-model pairs, the highest percentage uplift was achieved using the Max Attention mechanism on the Avazu dataset with the DCN base model with 0.4067% uplift. For the Criteo dataset, all the best-performing experiments utilized the Bitwise Attention mechanism, where neither layer normalization nor interaction component feeding was applied. The Criteo

[1] Interaction Component feeding is not applicable for DNN models

dataset is the only dataset that contains mixed field types, and numerical features might perform better with bitwise attention mechanism then vectorwise attention mechanisms.

For the Frappe dataset, the best-performing attention mechanisms varied across the models. In the DCN model, Bitwise Attention was the top performer, while Mean Attention excelled in the DNN model. In the DeepFM model, the best results were obtained using the Mean Max Summation Attention mechanism. Interestingly, for the Frappe dataset, the best-performing experiments involved a mix of using and not using layer normalization and interaction component feeding, depending on the model. Among the Frappe results, the highest percentage uplift was achieved with the DeepFM model using the Mean Max Summation Attention mechanism, leading to a 0.0508% uplift.

For the Movielens dataset, Bitwise Attention resulted as the best performer for the DCN model as in the Frappe dataset, while the DNN model performed best with the Mean Max Concatenation Attention mechanism. For the DeepFM model, Max Attention was the top performer. Similar to the Frappe dataset, the Movielens dataset achieved success with Bitwise Attention in the DCN model, even without layer normalization or interaction component feeding. The DNN model, on the other hand, showed the highest percentage uplift among all Movielens experiments, with a 0.1033% improvement using the Mean Max Concatenation Attention mechanism.

To measure the general success of attention mechanisms, two tests were performed on all datasets and base models. To compare the AUC values, we used two tests which are Wilcoxon signed-rank test and the Sign Test to measure whether there were significant changes. Even though these two tests are similar, as they are non-parametric tests, they have different objectives. The objective of the Wilcoxon signed-rank test is to determine if the distribution of the differences is symmetric about zero, while the objective of the Sign Test is to test whether there is a significant difference in the number of positive and negative differences in paired data.

The number of positive impacts, total pairs, median percentage changes, and corresponding p-values of the tests can be seen in the Table 5.3. Bitwise Attention, Max Attention and Mean Attention performed better than the base model more than half of

Table 5.3: Comparison of Different Attention Mechanisms Based on Median Percentage Change, Wilcoxon p-value, and Sign Test p-value

Attention Mechanism	# of Positive Impact	Total Experiments	Sign Test p-value	Percentage Change in Medians	Wilcoxon Test p-value
Bitwise Attention	7	12	0.7744	0.0505%	0.1677
Max Attention	7	12	0.7744	-0.1067%	0.4142
Mean Attention	9	12	0.1460	-0.1123%	0.4238
Mean Max Summation Attention	2	12	0.0386*	-0.5052%	0.0076*
Mean Max Concatenation Attention	4	12	0.3877	-0.9937%	0.0342*
Mean Std Summation Attention	1	11	0.0117*	-1.5916%	0.0109*
Mean Std Concatenation Attention	4	11	0.5488	-4.8987%	0.0420*

the times, but we cannot conclude as significant them according to Sign Test, as their p-values are greater than 0.05. Moreover, the most positive effect was achieved by obtaining higher test AUCs than base AUC values in 9 out of every 12 experiments in the Mean Attention mechanism. On the other hand, the median percentage is decreasing, implying that there should be a few base models and datasets that are affected negatively due to the Mean Attention mechanism. The joint attention mechanisms performed worse than the base model itself in most of the experiments. Additionally, joint mechanisms with summation aggregation significantly performed worse than base model according to Sign test. The median percentage change is decreasing for all joint mechanisms, and these performance decreases are significant according to Wilcoxon signed-rank test. Moreover, the median percentage change was most negatively affected by Mean Standard Attention mechanisms.

This analysis is also examined more specifically on a dataset basis, and the number of samples for each attention mechanism and dataset pair decreases to three. The overall performances of attention mechanisms on each dataset can be seen in Table 5.4.

According to test results, none of the test results are significant, which may be due to the small sample size, which is just three. However, the overall tendency has some patterns. In the Criteo dataset, which contains the most number of fields, all attention mechanisms performed better than the base model, except for the joint mechanisms with summation aggregation method. Additionally, the median values of all attention mechanism are increasing, and Bitwise Attention and Mean Attention mechanisms have the highest uplifts for the median percentage changes. The second dataset is Avazu, which has the second-highest number of fields. The Bitwise Attention and

Table 5.4: Comparison of Different Attention Mechanisms Based on Median Percentage Change, Wilcoxon p-value, and Sign Test p-value on across datasets

Attention Mechanism	Dataset	# of Positive Impact	Total Experiments	Sign Test p-value	Percentage Change in Medians	Wilcoxon Test p-value
Bitwise Attention	Avazu	2	3	1.000	-0.026%	0.750
Max Attention	Avazu	1	3	1.000	-0.026%	1.000
Mean Attention	Avazu	2	3	1.000	0.013%	0.750
Mean Max Summation Attention	Avazu	0	3	0.250	-0.131%	0.250
Mean Max Concatenation Attention	Avazu	1	3	1.000	-0.065%	0.750
Mean Std Summation Attention	Avazu	0	2	0.500	-0.223%	0.500
Mean Std Concatenation Attention	Avazu	1	2	1.000	-0.118%	1.000
Bitwise Attention	Criteo	3	3	0.250	0.049%	0.250
Max Attention	Criteo	3	3	0.250	0.037%	0.250
Mean Attention	Criteo	3	3	0.250	0.049%	0.250
Mean Max Summation Attention	Criteo	2	3	1.000	0.037%	0.157
Mean Max Concatenation Attention	Criteo	3	3	0.250	0.012%	0.250
Mean Std Summation Attention	Criteo	1	3	1.000	0.000%	0.317
Mean Std Concatenation Attention	Criteo	3	3	0.250	0.025%	0.250
Bitwise Attention	Movielens	2	3	1.000	-0.031%	0.500
Max Attention	Movielens	1	3	1.000	0.010%	0.500
Mean Attention	Movielens	0	3	0.250	-0.031%	0.250
Mean Max Summation Attention	Movielens	0	3	0.250	-0.878%	0.250
Mean Max Concatenation Attention	Movielens	0	3	0.250	-0.867%	0.250
Mean Std Summation Attention	Movielens	0	3	0.250	-1.652%	0.250
Mean Std Concatenation Attention	Movielens	0	3	0.250	-1.239%	0.250
Bitwise Attention	Frappe	0	3	0.250	-0.020%	0.180
Max Attention	Frappe	1	3	1.000	0.000%	0.317
Mean Attention	Frappe	3	3	0.250	0.010%	0.250
Mean Max Summation Attention	Frappe	0	3	0.250	-0.904%	0.250
Mean Max Concatenation Attention	Frappe	0	3	0.250	-0.731%	0.250
Mean Std Summation Attention	Frappe	0	3	0.250	-0.498%	0.250
Mean Std Concatenation Attention	Frappe	0	3	0.250	-0.904%	0.250

Mean Attention mechanisms performed better than the base model half of the times. The only uplift in the median values is occurred with the Mean Attention mechanism. On the other hand, the joint mechanisms with summation aggregation methods (Mean-Max Summation Attention and Mean-Standard Summation Attention) did not achieve better test AUCs than the base models. The third dataset is Frappe, which has 10 fields and the fewest instances among the datasets. Models with the Mean Attention mechanism performed better than the base models in all three experiments. However, only the model with the Max Attention mechanism performed better than the base models in terms of AUC value, while the remaining attention mechanisms achieved worse results than the base model AUC values. Finally, the last dataset is MovieLens dataset, which contains 3 fields, the lowest number of fields among the datasets. Moreover, only the Bitwise Attention mechanism performed better than the base model more than half of the times, and the attention mechanisms which are vector-wise mechanisms established poor performances. The reason behind this may

be the low number of fields lead to poor performances with vector-wise attention mechanisms.

5.2.2 Comparison of Attention Mechanisms (RQ2)

In this section, we aim to determine whether there are any significant performance differences among the attention mechanisms. To achieve this, non-parametric statistical methods are applied to ensure that the comparisons remain robust and applicable across diverse data scenarios. In these analyses, Friedman Chi-Square and posthoc Conover Friedman tests are applied to compare the performances of Attention mechanisms with each other. The Friedman Chi-Square test is a non-parametric test that examines whether there is a significant difference between groups. The null hypothesis is that there are no differences in the distributions of the groups, meaning it tests whether the groups are identical or not. Additionally, the posthoc Conover test is also a non-parametric test, and it is used to perform pairwise comparison after the Friedman Chi-Square which indicated a significant difference between the groups. The aim of the posthoc Conover test outputs groups that are significantly differ with other groups. Firstly, the overall performances of attention mechanisms are analyzed to capture a general view. The test result of the Friedman Chi-Square test indicates significantly different groups, which are the AUC values of different attention mechanisms on different datasets, with p-value as 9.810e-7. Therefore, the posthoc Conover test can be applied to differentiate the performances of the attention mechanisms. To conduct the posthoc Conover test, the mean rank of AUC values for each attention mechanism is used. The output of the posthoc Conover test is pairwise p-values which test if there is significant difference between the pairs, and the overall sign plot which represents the p-values can be seen in Figure 5.1.

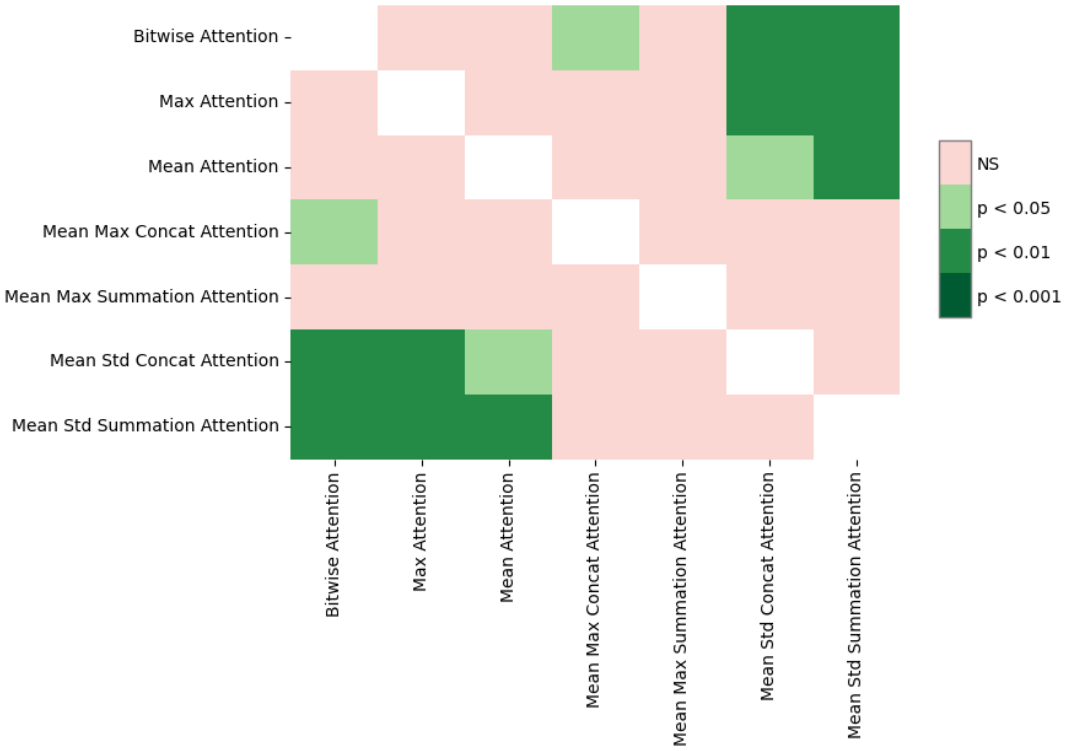

Figure 5.1: Overall Conover posthoc test sign plot of attention mechanisms

The pink cells represent insignificant differences, while the green cells indicate significantly different pairs. The tone of the color implies the significance level: the darkest shade represents the most significant differences, while the lightest shade represents the least significant differences. In addition to sign plot, the critical difference diagram, which can be seen in Figure 5.2, indicates differentiated groups. The dark line denotes that there is no significant difference between the connected attention mechanisms. The average rank of attention mechanisms is ordered from right to left, from highest to lowest. Therefore, we can conclude that the Bitwise Attention mechanism is the best performer on the overall. However, there is no significant difference between Bitwise Attention and Max Attention, Mean Attention, and Mean-Max Summation Attention mechanisms, as the CD line passes through all these attention mechanisms. On the other hand, Bitwise Attention significantly outperforms Mean-Std Concatenation Attention, Mean-Std Summation Attention, and Mean-Max Concatenation Attention. Additionally, Max Attention also significantly differs from the Mean-Std Attention mechanisms.

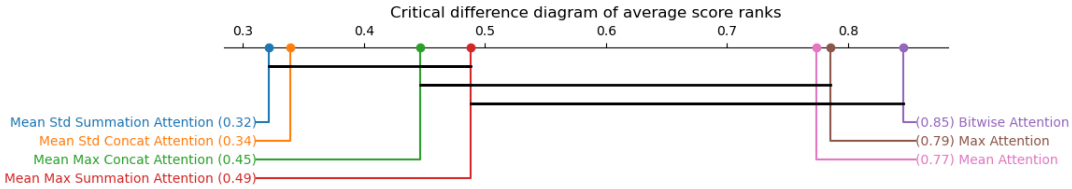

Figure 5.2: Overall Conover posthoc test Critical Difference diagram of attention mechanisms

To make a deep dive analysis, the same tests are conducted for each dataset separately. The first dataset is Avazu, and this dataset contains 22 categorical fields. We are not able to conclude that there is a statiscally significant difference between the groups according to Friedman Chi-Square test as the p-value is 0.117. Although the test did not indicate a statistical difference between the groups, we conducted the posthoc Conover to examine the effects of the groups on each other. The Conover test sign plot for Avazu dataset can be seen in Figure 5.3, and the CD diagram can be seen in Figure 5.4.

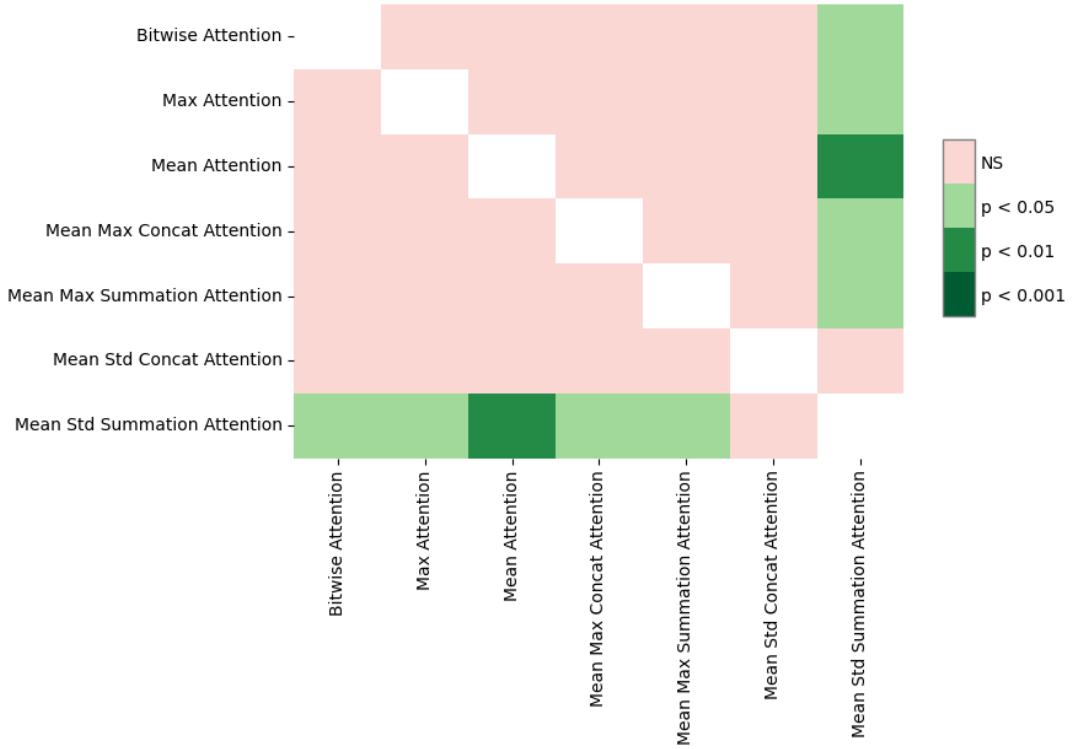

Figure 5.3: Conover posthoc test sign plot of attention mechanisms for Avazu Dataset

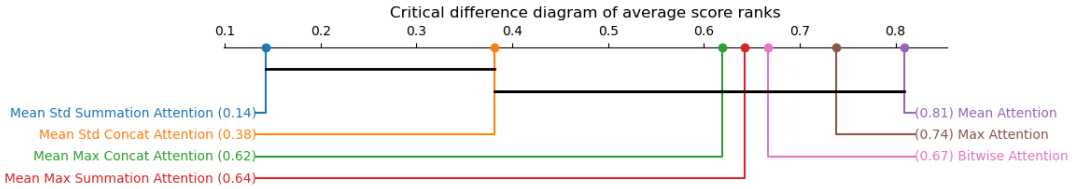

Figure 5.4: Conover posthoc test Critical Difference diagram of attention mechanisms for Avazu Dataset

Although the Friedman Chi-Square test does not indicate a significant difference between the groups, the posthoc Conover test shows a significant difference between Mean-Standard Summation and some other attention mechanisms under the pairwise comparison. Moreover, although in the overall comparisons the highest average ranked attention mechanism was Bitwise Attention, the highest average ranked attention mechanism is Mean Attention, followed by Max Attention for the Avazu dataset. However, there is indifference between Mean Attention, Max Attention, Bitwise Attention, Mean-Max Attention, Mean-Max Concatenation Attention and Mean-Standard Concatenation Attention.

The second dataset examined is the Criteo dataset, which has the most features, and its fields are of mixed types. According to the Friedman Chi-Square test, there is no statistically significant difference among the attention mechanisms, as the p-value is 0.208. However, it is worth examining the posthoc Conover test results to gain a basic understanding of the attention mechanisms on this dataset. The Conover test sign plot for Criteo dataset can be seen in Figure 5.5, and the CD diagram can be seen in Figure 5.6.

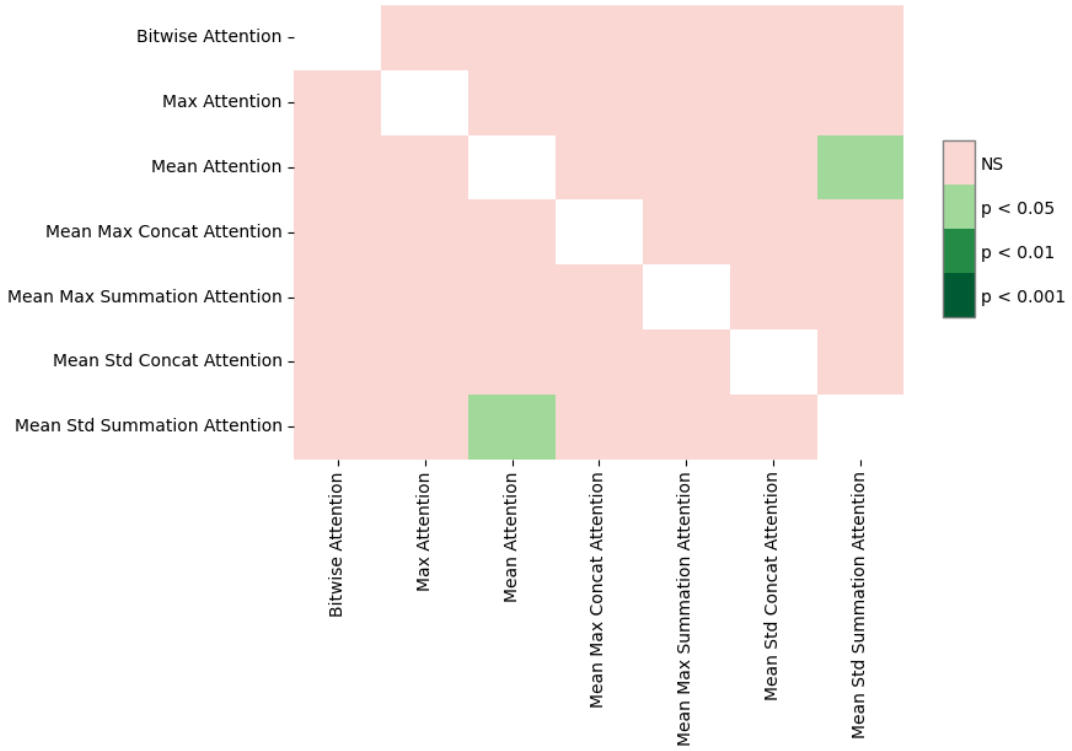

Figure 5.5: Conover posthoc test sign plot of attention mechanisms for Criteo Dataset

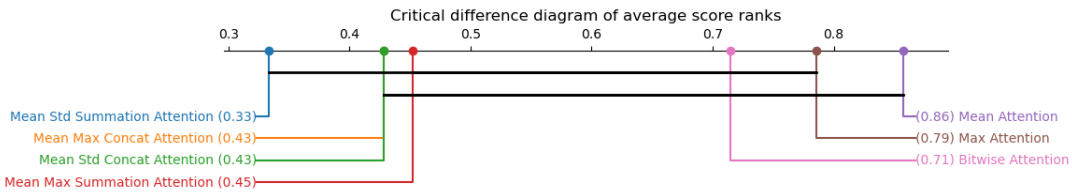

Figure 5.6: Conover posthoc test Critical Difference diagram of attention mechanisms for Criteo Dataset

As seen in Figure 5.6, the behavior of the attention mechanisms is similar to that of the Avazu dataset. As with the Avazu dataset, the highest average-ranked attention mechanism is Mean Attention, followed by Max Attention. However, there is indifference among Mean Attention, Max Attention, Bitwise Attention, Mean-Max Summation Attention, Mean-Max Concatenation Attention, and Mean-Standard Concatenation Attention. The only difference is that Mean-Standard Summation is almost indifferent to every attention mechanism except for Mean Attention.

The third dataset is Frappe, which contains 10 categorical fields with the lowest number of records. This is the first dataset that can be concluded that there is a significant difference between the attention mechanisms with p-value as 0.0402 according to the Friedman Chi-Square test. The differences among the attention mechanisms are more concrete in this dataset, and the Conover test sign plot for the Frappe dataset can be seen in Figure 5.7, and the CD diagram can be seen in Figure 5.8.

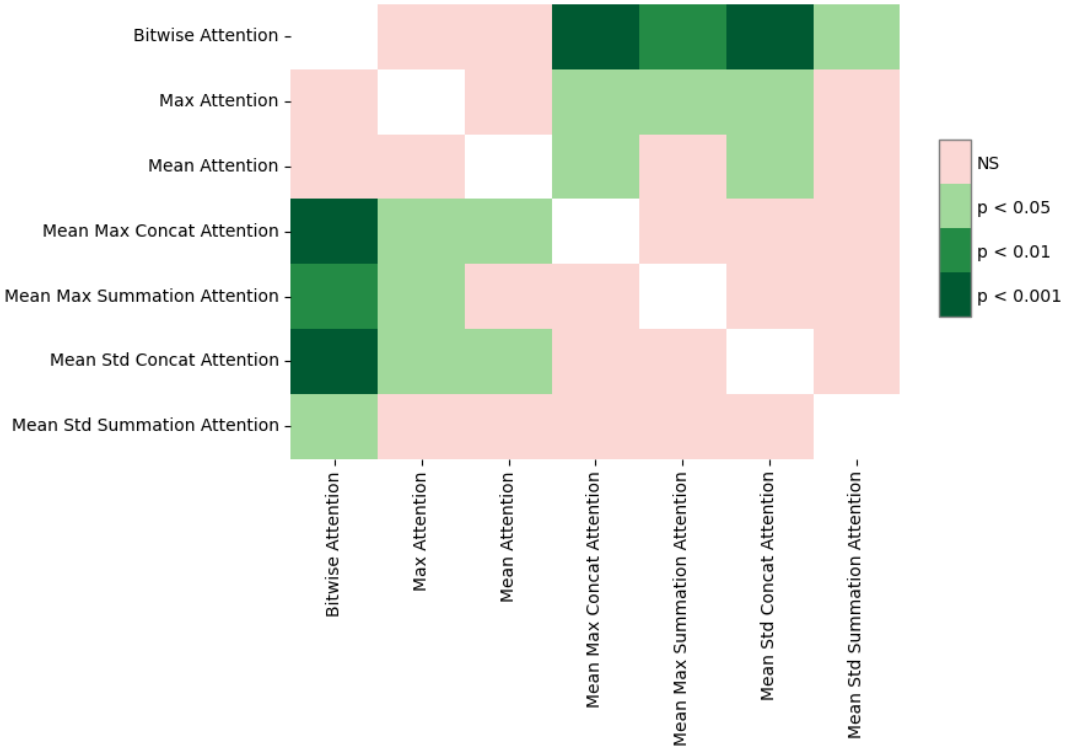

Figure 5.7: Conover posthoc test sign plot of attention mechanisms for Frappe Dataset

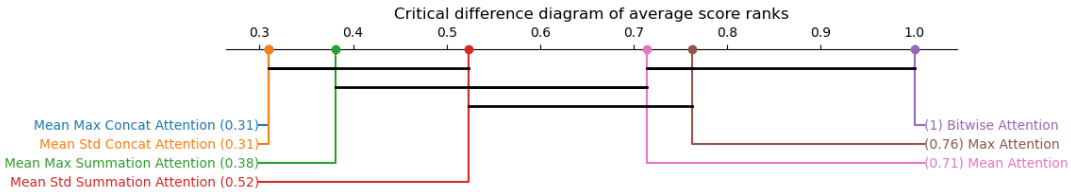

Figure 5.8: Conover posthoc test Critical Difference diagram of attention mechanisms for Frappe Dataset

According to the CD diagram of the posthoc Conover test, the highest average-ranked

attention mechanism is Bitwise Attention, with an average rank of 1, meaning it is always placed in the first rank. However, we cannot conclude this difference is significant among Bitwise Attention, Max Attention and Mean Attention, as the CD line is linked all three attention mechanism according to the CD diagram. Additionally, Bitwise Attention is significantly performing better than joint attention mechanisms, namely Mean-Max Concat Attention, Mean-Standard Concat Attention, Mean-Max Summation Attention, and Mean-Standard Summation Attention. This is illustrated by the absence of a connecting CD line between Bitwise Attention and these joint attention mechanisms, indicating a clear and significant performance distinction.

The last dataset is MoveiLens, which has only 3 categorical fields. The p-value is 0.0137 according to the Friedman Chi-Square test and there is significant differences among the groups of AUC values for different attention mechanisms. Moreover, the posthoc Conover test sign plot for MovieLens dataset can be seen in Figure 5.9, and the CD diagram can be seen in Figure 5.10.

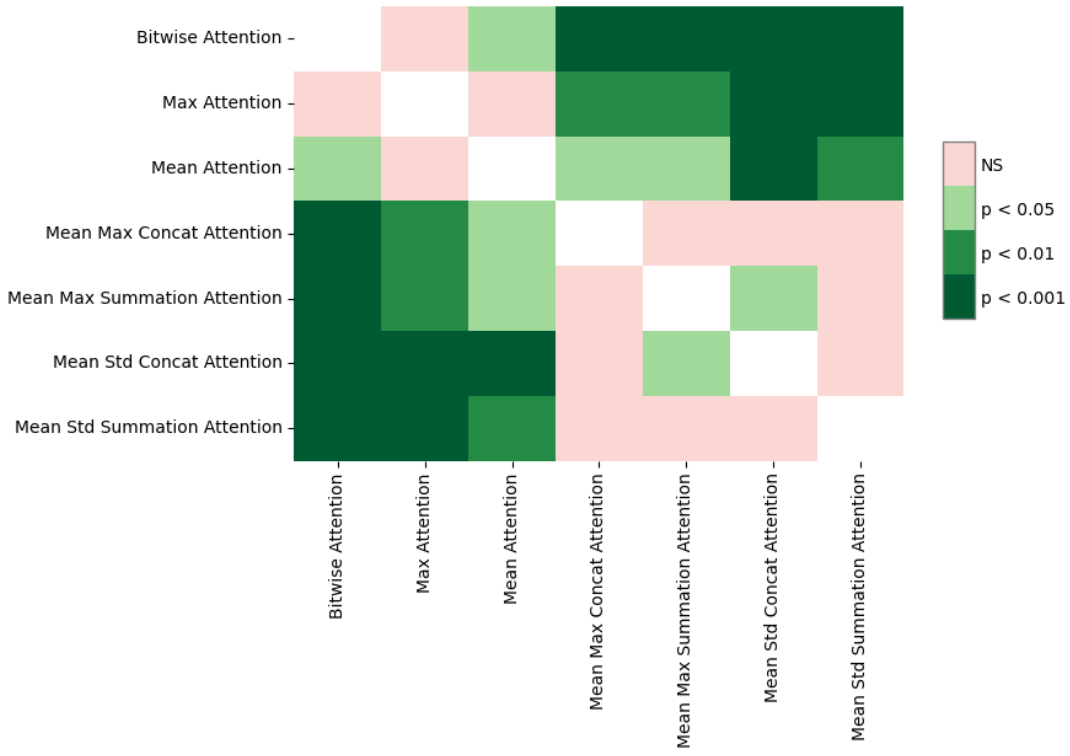

Figure 5.9: Conover posthoc test sign plot of attention mechanisms for MovieLens Dataset

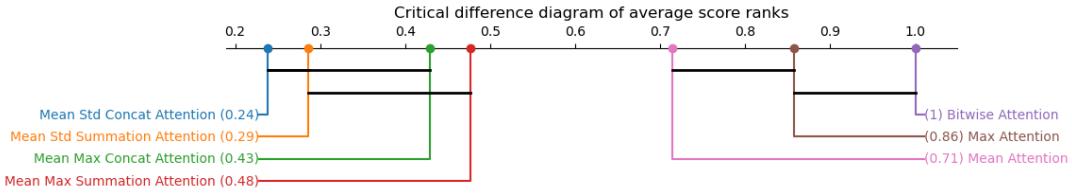

Figure 5.10: Conover posthoc test Critical Difference diagram of attention mechanisms for MovieLens Dataset

Based on the critical difference (CD) diagram of the Conover posthoc test on the MovieLens dataset, Bitwise Attention is again observed to have the highest average-ranked attention mechanism as in the Frappe dataset. Moreover, Bitwise Attention is significantly different from all other attention mechanisms except Max Attention. The difference between joint and solely attention mechanisms (e.g., Bitwise Attention, Max Attention, Mean Attention) is the most significant in MovieLens dataset. One possible reason for this could be the low number of fields, so vector-wise attention mechanisms may not provide any additional benefit to the categorical fields.

5.2.3 Impact of Layer Normalization on Attention-Based Models (RQ3)

To analyze the effect of layer normalization, we first compare the performance of attention mechanisms with and without layer normalization. All hyperparameters, except for layer normalization (whether it is active or not), are kept the same for the test pairs, including the base model, interaction feeding, and dataset. The Test Set AUC values of the pairs with and without layer normalization are used to analyze the performances. We conducted our tests first on all datasets and then on dataset breakdowns. The same two tests that are implemented in Section 5.2.1 are conducted to analyze the impact of layer normalization.

For the comparisons of all datasets, there are 140 pairs with and without layer normalization which 78 of the pairs showed better performance with layer normalization. However, we could not conclude that the difference in the number of positive pairs are significantly more than negative ones according to the Sign Test (p-value = 0.20). On the other hand, there is a 1.02% increase in median of AUC values when layer

normalization is active. Moreover, according to the Wilcoxon signed-rank test, the median difference between the pairs are significant as we can reject the null hypothesis at a confidence level of 5% (p-value = 0.00003).

When we examine the effect of the layer normalization on a per-dataset basis, we observed different behaviors across all datasets. There are 35 pairs in each dataset, and Table 5.5 shows the number of experiments where layer normalization has a positive effect when it is active, compared to when it is not, and the Sign Test p-values.

Table 5.5: Impact of Layer Norm across different datasets

Dataset	# of Layer Norm Positive Impact	Total Experiments	p-value
Avazu	32	35	4.1770e-07*
Criteo	3	35	4.1770e-07*
MovieLens	21	35	0.3105
Frappe	22	35	0.1755

Layer normalization performed best on the Avazu dataset, which contains only 22 categorical features, as 32 pairs of the 35 all pairs had a better performance under layer normalization. In addition to that, the difference is significant according to the Sign Test with p-value as 4.1770e-07. Conversely, in the Criteo dataset, which is a mixed dataset with 26 categorical and 13 numerical fields, layer normalization had a positive impact in only 3 out of 35 pairs, and the Sign Test outputs that the negative difference is also significant (p-value = 4.1770e-07). In the Frappe and MovieLens datasets, we observe similar results with 22 and 21 positive impacts out of 35 pairs, respectively, and the positive number differences to negative numbers is insignificant according to the Sign Test.

Table 5.6: Layer Norm effects for Median Changes on across Datasets

Dataset	Percentage Change in Medians	p-value
Avazu	0.131%	6.222e-06*
Criteo	-0.061%	1.967e-06*
MovieLens	0.238%	0.0015*
Frappe	0.183%	0.0025*

As shown in Table 5.6, the effect of layer normalization is significantly positive in the Avazu, MovieLens and Frappe datasets according to Wilcoxon signed-rank test, and the highest uplift achieved in MovieLens dataset with 0.238% median change. On the other hand, in the Criteo dataset, the median of AUC values is decreasing with 0.061% when layer normalization is active, and this change in the median is significant at a confidence level of 5%. As we stated layer normalization only performed worse on the Criteo dataset, and the reason might be occurred due to intrinsic variability in numerical features that layer normalization might over-smooth.

We conducted our tests for the impact analysis of layer normalization across the datasets and base models. Table 5.7 shows that the effects of datasets on models reflect similar trends. All models (DNN, DCN, DeepFM) show consistent results on the Avazu dataset, and except DNN tests, the uplifts are significant for both tests which are Sign Test and Wilcoxon signed-rank test. For the Criteo dataset, all test results are significant and we can conclude that layer normalization performs poorly independent from the base model. For the small datasets which are Frappe and MovieLens, the behaviors are similar. Although there are more positive impacts than negatives, we could not able define as significant difference according to Sign test. However, some of the median percentage changes in the distributions are significant (DCN-Frappe, DeepFM-MovieLens, DCN-MovieLens).

Table 5.7: Impact of Layer Norm across different Datasets and Base Models

Dataset	Base Model	# of Layer Norm Positive Impact	Total Experiments	Sign Test p-value	Percentage Change in Medians	Wilcoxon Test p-value
Avazu	DNN	5	7	0.4531	0.0393%	0.1563
Avazu	DeepFM	14	14	0.0001*	0.1245%	0.0001*
Avazu	DCN	13	14	0.0018*	0.1901%	0.0004*
Criteo	DNN	0	7	0.0156*	-0.1475%	0.0156*
Criteo	DeepFM	2	14	0.0129*	-0.0430%	0.0036*
Criteo	DCN	1	14	0.0018*	-0.0614%	0.0041*
Frappe	DNN	5	7	0.4531	0.4390%	0.1563
Frappe	DeepFM	8	14	0.7905	0.0712%	0.1520
Frappe	DCN	9	14	0.4240	0.1884%	0.0295*
Movielens	DNN	3	7	1.0	0.0829%	0.1441
Movielens	DeepFM	10	14	0.1796	0.9454%	0.0295*
Movielens	DCN	8	14	0.7905	0.2695%	0.1040

In the last stage of our analysis on the effects of layer normalization, we conducted our tests based on attention mechanisms and observed that the effect of layer normalization on joint attention mechanisms (Mean-Max Attention and Mean-Std Attention) is significantly positive for both tests. These joint mechanisms perform poorly as we discussed, but their performances are better and might benefit more from layer normalization due to their inherent weaknesses. There may be several reason behind that. Firstly, one of the layer normalization advantage is stabilizing the training by normalizing the inputs to each layer. Mean-Max and Mean-Std models were performing poorly and this smoothing feature of layer normalization might reduce the variance in the output of the models. Secondly, we observed gradient flow problems in some of the experiments with Mean-Std where the model could not achieve learning, and these results occurred when the layer normalization was not utilized. Layer normalization might help to solve these issues by ensuring that the gradients maintain in a stable range.

On the other hand, in Bitwise, Max and Mean Attention, the majority stated that better results were obtained without layer normalization in terms of number of negative pairs and decrease in the median change, but we can conclude that only mean attention is significant as a result of the Sign Test. The overall statistics for effects of layer normalization across attention mechanisms can be seen in Table 5.8

Table 5.8: Impact of Layer Norm across Attention Mechanisms. LNPI refers to Layer Normalization Positive Impact

Attention Type	# of LNPI	Total Experiments	Sign Test p-value	Percentage Change in Medians	Wilcoxon Test p-value
Bitwise Attention	8	20	0.5034	-0.0561%	0.5061
Max Attention	6	20	0.1153	-0.0112%	0.0533
Mean Attention	4	20	0.0118*	-0.0169%	0.0133*
Mean Max Concatenation Attention	14	20	0.1153	0.9016%	0.0019*
Mean Max Summation Attention	14	20	0.1153	0.4688%	0.0025*
Mean Std Concat Attention	15	20	0.0414*	1.0333%	0.0012*
Mean Std Summation Attention	17	20	0.0026*	0.7191%	0.0003*

5.2.4 Impact of Not Feeding Attention Mechanisms Outputs on Interaction Component (RQ4)

Another hyperparameter in our test set is whether or not to activate the feeding of updated feature embeddings with the attention mechanism to low-order feature interaction components.

To analyze the impacts of feeding updated feature embeddings to the interaction and deep components together, the following steps are the same as those described in the layer normalization analysis steps in subsection 5.2.3. The overall performance with and without the feeding mechanisms is examined under the circumstances where all hyperparameters, except for feeding the interaction component (base model, interaction feeding, and dataset), are kept the same for the test pairs. One difference from the layer normalization analysis is that the experiments are only conducted with DeepFM and DCN bases, as the DNN model does not have a low-order interaction component. We have used same statistical tests to monitor the evaluate the feeding interaction component effects. The evaluation metrics are the AUC values of with and without feeding the interaction component.

For the comparisons across all datasets, there are 112 pairs with and without feeding the interaction component, compared to 140 pairs for layer normalization. The number of pairs decreased because the DNN model cannot be used for this test by nature. Moreover, only 49 of the pairs showed an uplift in AUC values when the feeding mechanism was activated, indicating that in most experiments, feeding the interaction

component with updated feature vectors led to performance decreases. However, we cannot state that the difference in the number of negative pairs is significantly greater than the positive pairs according to the Sign Test, with a p-value of 0.2191. Additionally, the median AUC values for the experiments where the feeding mechanism was activated decreased by 0.64% compared to those where the feeding mechanism was deactivated. However, this median value decrease is not significant according to the Wilcoxon signed-rank test, with a p-value of 0.71.

For the dataset-based analysis, the impacts of the feeding interaction mechanism are similar across all datasets. Each dataset contains 28 pairs, and Table 5.9 demonstrates the number of experiments with positive impacts, the total number of experiments, and the p-value for the Sign Test. All datasets, except for MovieLens, are negatively affected by feeding the interaction component in terms of the number of positive impacts. The only positive impact occurred in the MovieLens dataset, with 16 out of the 28 pairs resulting in better AUC values. However, we cannot conclude that these effects are significant, as their p-values for the sign test are greater than 0.05.

Table 5.9: Impact of Not Feeding Attention Mechanisms Outputs on Interaction Component across different datasets. ICFPI refers to Interaction Component Feeding Positive Impact

Dataset	# of ICFPI	Total Experiments	p-value
Avazu	13	28	0.8506
Criteo	9	28	0.0872
MovieLens	16	28	0.5716
Frappe	11	28	0.3449

As shown in Table 5.10, feeding the interaction component with updated feature vectors leads to smaller changes in median differences compared to changes in layer normalization (Table 5.6). The interaction component feeding helps to increase performance in the Avazu and MovieLens datasets, with median changes of 0.0065%

and 0.0414%, respectively. However, these changes are insignificant according to the Wilcoxon signed-rank test. Moreover, the median AUC values decrease for the Criteo and Frappe datasets, but these decreases are also insignificant.

Table 5.10: Not Feeding Attention Mechanisms Outputs on Interaction Component effects for Median Changes on across Datasets

Dataset	Percentage Change in Medians	p-value
Avazu	0.0065%	0.3739
Criteo	-0.0184%	0.2743
MovieLens	0.0414%	0.5301
Frappe	-0.0356%	0.5388

When we dive into dataset and base model analysis, the general tendency does not change, and all changes are insignificant except for the DeepFM base model in the Criteo dataset. The most positive impact is observed for the DCN base model in the MovieLens dataset, with 16 out of the 28 pairs resulting in better AUC values. Feeding the interaction component works best for MovieLens, as we discussed. However, the only significantly impacted dataset and base model combination is the Criteo dataset with the DeepFM base model. In this case, only 2 out of the 14 pairs are positively affected when feeding the interaction component is active, with a significant p-value of 0.0129 according to the sign test. The median percentage decrease is 0.0184%, which is significant according to the Wilcoxon signed-rank test.

Table 5.11: Impact of Not Feeding Updated Features with Attention Mechanisms on Interaction Component across different Datasets and Base Models. ICFPI refers to Interaction Component Feeding Positive Impact

Dataset	Base Model	# of ICFPI	Total Experiments	Sign Test p-value	Percentage Change in Medians	Wilcoxon Test p-value
Avazu	DeepFM	8	14	0.7905	0.0065%	0.5759
Avazu	DCN	5	14	0.4240	-0.0197%	0.1726
Criteo	DeepFM	2	14	0.0129*	-0.0184%	0.0481*
Criteo	DCN	7	14	1.0000	0.0061%	0.3711
Frappe	DeepFM	5	14	0.4240	-0.0712%	0.5301
Frappe	DCN	6	14	0.7905	-0.0356%	0.8751
MovieLens	DeepFM	7	14	1.0000	0.0259%	0.5830
MovieLens	DCN	9	14	0.4240	-0.0465%	0.6001

Finally, the tests are conducted based on attention mechanisms, and the results indicate that the Bitwise Attention mechanism has a minor impact, with only 5 out of 16 experiments showing a positive effect and a median percentage change of 0.0168%, and these results are statistically insignificant according to the sign test and Wilcoxon signed-rank. Max Attention and Mean Attention are the only attention mechanisms that exhibited positive effects, with 11 out of 16 experiments and 10 out of 16 experiments resulting in better AUC values, respectively. However, the significance threshold was not reached for the number of positive pairs, but the median percentage change, which is 0.0730% for Max Attention, can be concluded as significant according to Wilcoxon signed-rank test. Although the median change, which is 0.0899% for Mean Attention, is higher than Max Attention, we have to fail to reject the null hypothesis for Wilcoxon signed-rank test. All joint attention mechanisms are negatively affected in terms of number of positive pairs, but none of them are significant according to the sign test. In addition to that, the median percentage change for Mean-Standard Summation Attention is -0.0283% and it is significant according to Wilcoxon signed-rank test. The only uplift in median percentage change among the joint mechanisms is occurred in Mean Max Summation Attention with 0.1130%, but we cannot conlude this uplift as significant. The overall statistics for effects of feeding updated features across attention mechanisms can be seen in Table 5.12

Table 5.12: Impact of Not Feeding Updated Features with Attention Mechanisms on Interaction Component across Attention Mechanisms. ICFPI refers to Interaction Component Feeding Positive Impact

Attention Type	# of ICFPI	Total Experiments	Sign Test p-value	Perc. Change in Medians	Wilcoxon Test p-value
Bitwise Attention	5	16	0.2101	0.0168%	0.2941
Max Attention	11	16	0.2101	0.0730%	0.0352*
Mean Attention	10	16	0.4545	0.0899%	0.1398
Mean Max Concatenation Attention	7	16	0.8036	-0.0567%	0.7820
Mean Max Summation Attention	6	16	0.4545	0.1130%	0.7545
Mean Std Concatenation Attention	5	16	0.2101	-1.544%	0.5095
Mean Std Summation Attention	5	16	0.2101	-0.0283%	0.0950*

5.2.5 Key Findings

The prominent findings of the conducted analyses can be listed as follows:

- **General Performances of Attention Mechanisms:**
  - The solely attention mechanisms (Bitwise Attention, Max Attention, and Mean Attention) generally performed better than the base models, though joint mechanisms achieved the highest scores in a few cases.
  - In the dataset-specific analyses, most attention mechanisms outperformed the base models in the Criteo dataset, which features a diverse set of fields. However, when the number of fields is decreasing the performance of attention mechanisms are also starting to decrease.
  - Bitwise Attention is the least affected mechanism by the reduction in the number of fields, maintaining better performance across datasets with fewer fields.
  - Although Mean-Std Attention is used in other domains, this attention mechanism performed poorly in most of the experiments. The reason behind this may be that Standard Deviation pooling captures the variance in the feature embeddings, which can be an important indicator in other domains, such as voice recognition, but not for the CTR prediction task.
- **Comparison of Attention Mechanisms:**

- In the comparative analysis, Bitwise Attention, Mean Attention, and Max Attention generally outperform joint mechanisms. However, no significant performance differences were observed between attention mechanisms for the most complex datasets (Criteo and Avazu).
- Mean Attention ranks highest on average for the Criteo and Avazu datasets, which have a larger number of fields.
- For datasets with fewer fields (Movielens and Frappe), joint mechanisms perform poorly compared to other attention mechanisms, with Bitwise Attention emerging as the best performer.
- It is observed that vector-wise attention mechanisms can face trouble leveraging the advantages of the attention mechanism if there are a limited number of fields.
- The average ranking of the attention mechanisms in the radar chart in Figure 5.11 and the data matrix, which contains the top 3 performing attention mechanisms for each dataset in Figure 5.12 illustrates the overall performance of different attention mechanisms across datasets with varying characteristics and visually represents our conclusions about the varying effectiveness of attention mechanisms depending on dataset complexity and the number of fields.

- **Impact of Layer Normalization:**
  - The effects of layer normalization on attention mechanisms vary depending on the dataset and mechanism. Layer normalization generally improves performance in all datasets except Criteo, which contains mixed categorical and numerical fields. The reason behind may be because of the gradient flows that are observed a few times, and implementing layer normalization may help to stabilize the gradients, enhance learning and overall model performance.
  - Joint attention mechanisms performed better when the layer normalization is implemented. The reason behind may be because of the gradient flows that are observed a few times, and implementing layer normalization may help to stabilize the gradients, enhance learning and overall model performance.

- **Impact of Feeding Low-Order Feature Interaction Component:**

  - The impact of feeding interaction components with updated weights by attention mechanisms is mostly neutral. However, Max Attention and Mean Attention show slight performance improvements when the interaction layer is fed with updated weights instead of raw field vectors.

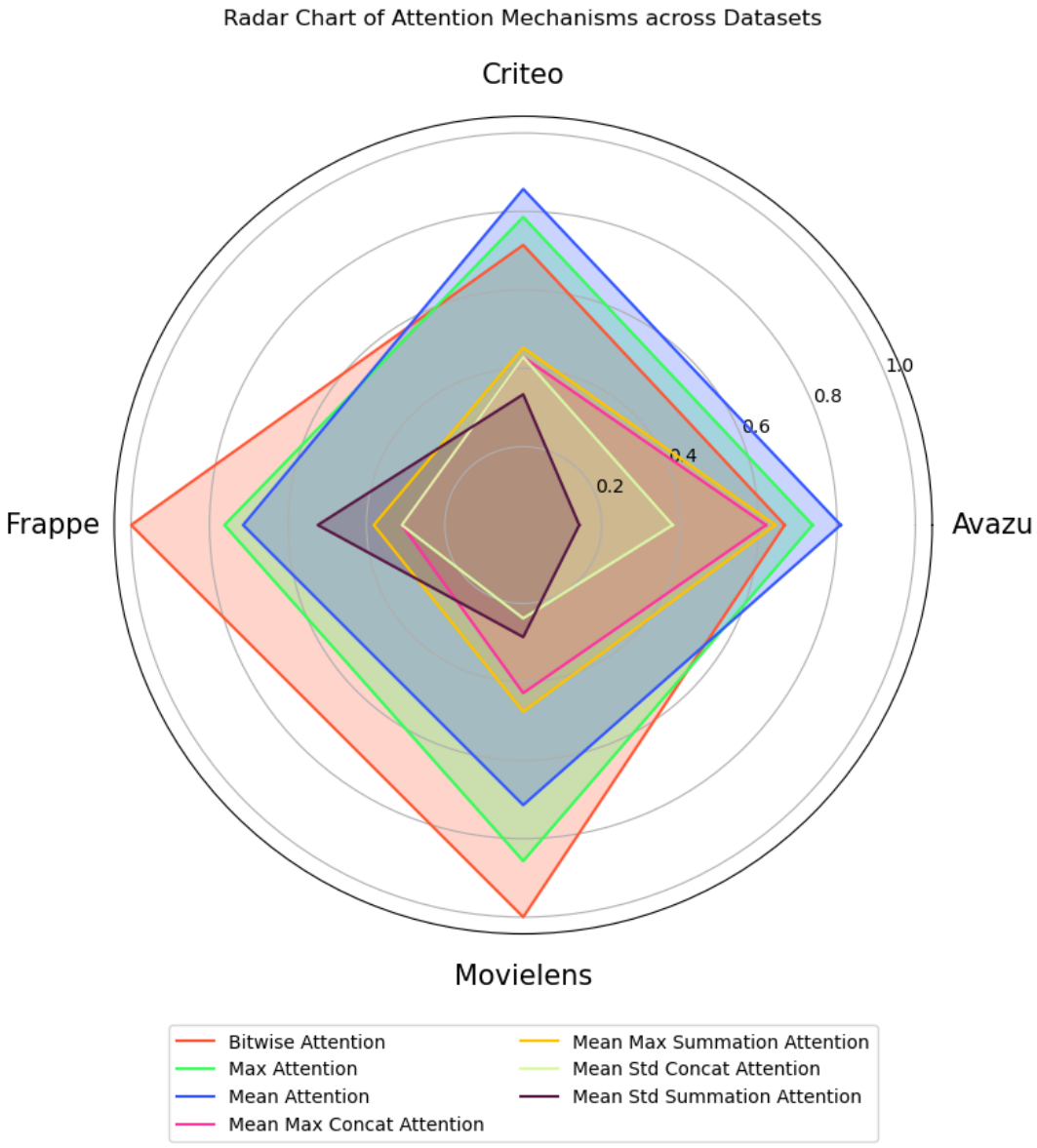

Figure 5.11: Radar Chart of Attention Mechanisms across the Datasets

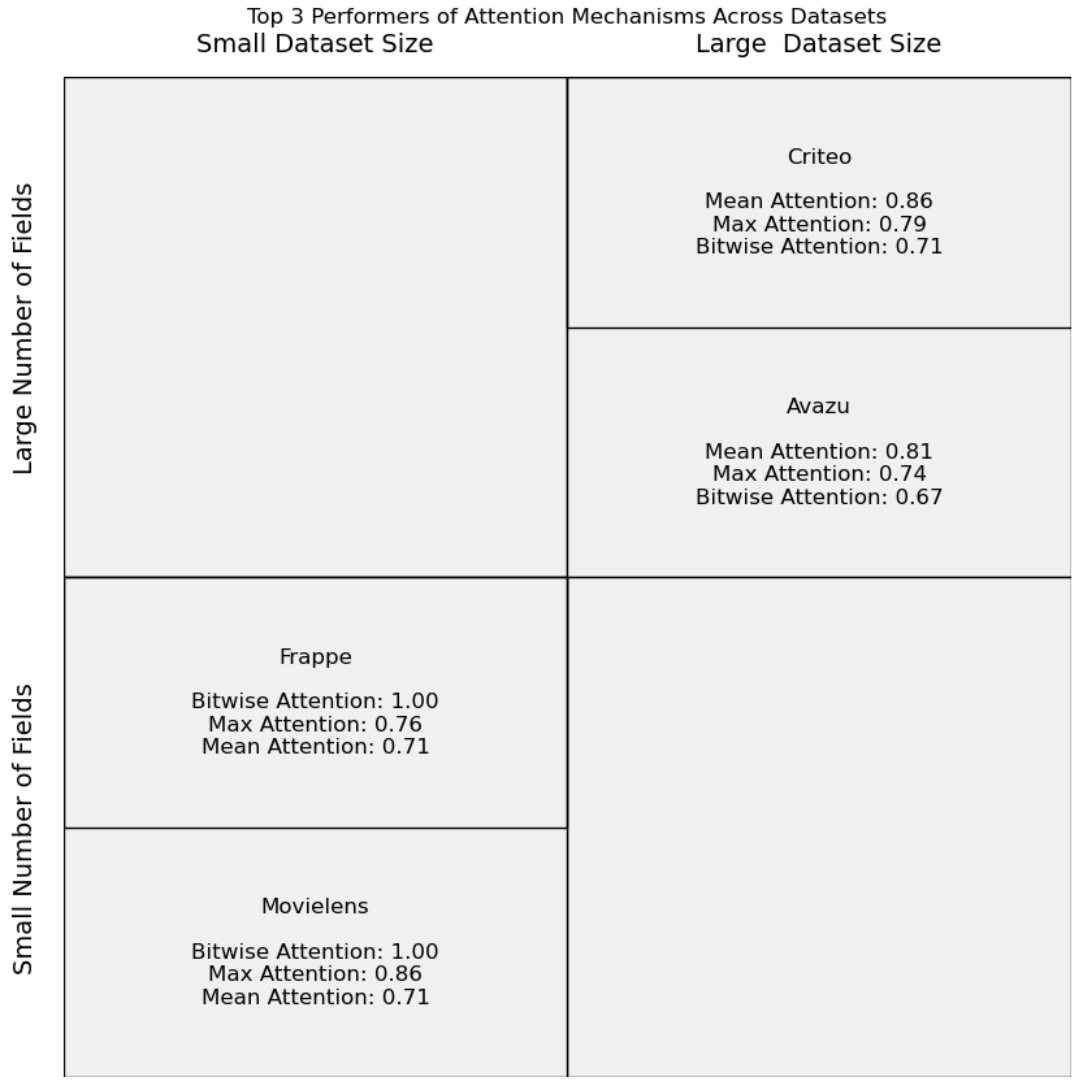

Figure 5.12: Data Matrix of Attention Mechanisms across the Datasets

CHAPTER 6

CONCLUSION

To conclude, in this thesis, we aimed to examine the effects of attention mechanisms in CTR prediction models on highly used datasets and the models. Our analysis reveals plenty of key insights into the performance of attention mechanisms on the conventional models under the different type of hyper-parameter settings. The attention mechanisms do not have to perform well in every condition, and the design of choice should be considered before the experiment phase. For instance, the joint attention mechanisms occasionally achieved the highest score, but the solely attention mechanisms like Bitwise Attention, Max Attention and Mean Attention mechanisms are more robust and performed well in most of the cases. Therefore, it can be suggested that if the dataset that will be used to predict CTR values has few number of categorical variables, it is more likely to perform poorly with joint mechanism while Bitwise Attention mechanism can perform better than without attention mechanism.

Additionally, it is observed that layer normalization generally performed well. One difference was in the Criteo dataset, which contains mixed type of fields, where the layer normalization may lead to over-smoothing in the numerical features.

Finally, the feeding updated weights from attention mechanisms into the interaction layer showed neutral to slightly positive effects, especially for Max Attention and Mean Attention.

For the future work, the main objective is to increase the number of different types of datasets to achieve better generalization on the behaviors of attention mechanisms across diverse datasets. Another objective is to explore the impacts of the residual networks in the attention mechanisms which is proved to reduce the information loss

in some of the studies [24], and the other future objective is exploring the performance of self-attention mechanism [19, 18], which is the main structure of the groundbreaking transformer architectures.

REFERENCES

[1] W. Cheng, Y. Shen, and L. Huang, "Adaptive factorization network: Learning adaptive-order feature interactions," 2020.

[2] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "Deepfm: a factorization-machine based neural network for ctr prediction," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, p. 1725–1731, AAAI Press, 2017.

[3] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *Proceedings of the ADKDD'17*, ADKDD'17, (New York, NY, USA), Association for Computing Machinery, 2017.

[4] W. Song, C. Shi, Z. Xiao, Z. Duan, Y. Xu, M. Zhang, and J. Tang, "Autoint: Automatic feature interaction learning via self-attentive neural networks," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, (New York, NY, USA), p. 1161–1170, Association for Computing Machinery, 2019.

[5] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, S. Chikkerur, D. Liu, M. Wattenberg, A. M. Hrafnkelsson, T. Boulos, and J. Kubica, "Ad click prediction: a view from the trenches," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2013.

[6] K. Mao, J. Zhu, L. Su, G. Cai, Y. Li, and Z. Dong, "Finalmlp: An enhanced two-stream mlp model for ctr prediction," 2023.

[7] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, S. Chikkerur, D. Liu, M. Wattenberg, A. M. Hrafnkelsson, T. Boulos, and J. Kubica, "Ad click prediction: a view from the trenches," in *Proceedings of the 19th ACM SIGKDD International Conference*

on Knowledge Discovery and Data Mining, KDD '13, (New York, NY, USA), p. 1222–1230, Association for Computing Machinery, 2013.

[8] M. Richardson, E. Dominowska, and R. Ragno, "Predicting clicks: estimating the click-through rate for new ads," in *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, (New York, NY, USA), p. 521–530, Association for Computing Machinery, 2007.

[9] S. Rendle, "Factorization machines," in *2010 IEEE International Conference on Data Mining*, pp. 995–1000, 2010.

[10] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah, "Wide & deep learning for recommender systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, DLRS 2016, (New York, NY, USA), p. 7–10, Association for Computing Machinery, 2016.

[11] J. Zhu, Q. Dai, L. Su, R. Ma, J. Liu, G. Cai, X. Xiao, and R. Zhang, "Bars: Towards open benchmarking for recommender systems," 2022.

[12] M. Blondel, A. Fujino, N. Ueda, and M. Ishihata, "Higher-order factorization machines," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, (Red Hook, NY, USA), p. 3359–3367, Curran Associates Inc., 2016.

[13] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun, "xdeepfm: Combining explicit and implicit feature interactions for recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, (New York, NY, USA), p. 1754–1763, Association for Computing Machinery, 2018.

[14] T. Huang, Z. Zhang, and J. Zhang, "Fibinet: combining feature importance and bilinear feature interaction for click-through rate prediction," in *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, (New York, NY, USA), p. 169–177, Association for Computing Machinery, 2019.

[15] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *2018*

IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7132–7141, 2018.

[16] X. He and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, (New York, NY, USA), p. 355–364, Association for Computing Machinery, 2017.

[17] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of the 32nd International Conference on Machine Learning* (F. Bach and D. Blei, eds.), vol. 37 of *Proceedings of Machine Learning Research*, (Lille, France), pp. 2048–2057, PMLR, 07–09 Jul 2015.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 6000–6010, Curran Associates Inc., 2017.

[19] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[20] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, "Attentional factorization machines: learning the weight of feature interactions via attention networks," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, p. 3119–3125, AAAI Press, 2017.

[21] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, "Deep interest network for click-through rate prediction," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, (New York, NY, USA), p. 1059–1068, Association for Computing Machinery, 2018.

[22] Y. Feng, F. Lv, W. Shen, M. Wang, F. Sun, Y. Zhu, and K. Yang, "Deep session interest network for click-through rate prediction," in *Proceedings of the 28th In-*

ternational Joint Conference on Artificial Intelligence, IJCAI'19, p. 2301–2307, AAAI Press, 2019.

[23] H. Liu, J. Ma, and K. Qian, "Dafm: A ctr prediction model based on attention mechanism," in *Proceedings of the 3rd International Conference on Industrial Control Network and System Engineering Research*, ICNSER '22, (New York, NY, USA), p. 1–6, Association for Computing Machinery, 2022.

[24] P. Zhang, Z. Zheng, and J. Zhang, "Fibinet++: Reducing model size by low rank feature interaction layer for ctr prediction," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23, (New York, NY, USA), p. 4425–4429, Association for Computing Machinery, 2023.

[25] J. Zhang, T. Huang, and Z. Zhang, "Fat-deepffm: Field attentive deep field-aware factorization machine," 2019.

[26] Y. Juan, D. Lefortier, and O. Chapelle, "Field-aware factorization machines in a real-world online advertising system," in *Proceedings of the 26th International Conference on World Wide Web Companion*, WWW '17 Companion, (Republic and Canton of Geneva, CHE), p. 680–688, International World Wide Web Conferences Steering Committee, 2017.

[27] X. Shi, Y. Yang, and C. Tao, "Ctr prediction model considering the importance of embedding vector," in *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 167–171, 2021.

[28] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Eca-net: Efficient channel attention for deep convolutional neural networks," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 11531–11539, IEEE Computer Society, jun 2020.

[29] H. Saribas, C. Yesil, S. Dilbaz, and H. Orenbas, "Mmbattn: Max-mean and bit-wise attention for ctr prediction," 2023.

[30] F. Zhou, W. Sheng, Z. Lu, B. Kang, M. Chen, and G. Qiu, "Super-resolution image visual quality assessment based on structure–texture features," *Image Commun.*, vol. 117, sep 2023.

[31] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," in *Interspeech 2018*, ISCA, Sep. 2018.

[32] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016.

[33] J. Huang, K. Hu, Q. Tang, M. Chen, Y. Qi, J. Cheng, and J. Lei, "Deep position-wise interaction network for ctr prediction," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, (New York, NY, USA), p. 1885–1889, Association for Computing Machinery, 2021.

[34] Criteo, "The criteo dataset." https://www.kaggle.com/c/criteo-display-ad-challenge/data, 2021. Accessed: 2022-07-31.

[35] Avazu, "The avazu dataset." https://www.kaggle.com/c/avazu-ctr-prediction/data, 2021. Accessed: 2022-07-31.

[36] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context." https://grouplens.org/datasets/movielens, 2015. Accessed: 2024-04-01.

[37] L. Baltrunas, K. Church, A. Karatzoglou, and N. Oliver, "Frappe: Understanding the usage and perception of mobile app recommendations in-the-wild." https://www.baltrunas.info/_files/archives/be17b7_474755a5e9684a749a012516d04fb456.zip?dn=Mobile_Frappe.zip, 2015. Accessed: 2024-04-01.

[38] J. Zhu, J. Liu, S. Yang, Q. Zhang, and X. He, "Open benchmarking for click-through rate prediction," 2021.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

APPENDIX A

ATTENTION MECHANISM EXPERIMENTS

A. Experiment vs Base Values for Option 1 Settings

Table A.1: Experiment vs Base Values for Option 1 Settings

Base Model	Attention Mechanism	DataSet	Layer Normalization	Interaction Component Feeding	Experimented Attention Mechanism AUC	Base Model AUC
DCN	Bitwise Attention	Avazu	-	Yes	0.7633	0.7623
DCN	Bitwise Attention	Criteo	-	Yes	0.8141	0.8137
DCN	Bitwise Attention	Movielens	-	Yes	0.9689	0.9686
DCN	Bitwise Attention	Frappe	-	Yes	0.9842	0.9843
DeepFM	Bitwise Attention	Avazu	-	Yes	0.7633	0.7642
DeepFM	Bitwise Attention	Criteo	-	Yes	0.8142	0.8137
DeepFM	Bitwise Attention	Movielens	-	Yes	0.968	0.9683
DeepFM	Bitwise Attention	Frappe	-	Yes	0.9842	0.9844
DNN	Bitwise Attention	Avazu	-	-	0.7638	0.7635
DNN	Bitwise Attention	Criteo	-	-	0.814	0.8136
DNN	Bitwise Attention	Movielens	-	-	0.968	0.9676
DNN	Bitwise Attention	Frappe	-	-	0.9844	0.9844
DCN	Max Attention	Avazu	-	Yes	0.7643	0.7623
DCN	Max Attention	Criteo	-	Yes	0.8141	0.8137
DCN	Max Attention	Movielens	-	Yes	0.9691	0.9686
DCN	Max Attention	Frappe	-	Yes	0.9846	0.9843
DeepFM	Max Attention	Avazu	-	Yes	0.7631	0.7642
DeepFM	Max Attention	Criteo	-	Yes	0.814	0.8137
DeepFM	Max Attention	Movielens	-	Yes	0.9685	0.9683
DeepFM	Max Attention	Frappe	-	Yes	0.9844	0.9844
DNN	Max Attention	Avazu	-	-	0.7633	0.7635
DNN	Max Attention	Criteo	-	-	0.8139	0.8136
DNN	Max Attention	Movielens	-	-	0.9653	0.9676
DNN	Max Attention	Frappe	-	-	0.9844	0.9844
DCN	Mean Attention	Avazu	-	Yes	0.7635	0.7623
DCN	Mean Attention	Criteo	-	Yes	0.8141	0.8137
DCN	Mean Attention	Movielens	-	Yes	0.9688	0.9686
DCN	Mean Attention	Frappe	-	Yes	0.9844	0.9843
DeepFM	Mean Attention	Avazu	-	Yes	0.7636	0.7642
DeepFM	Mean Attention	Criteo	-	Yes	0.8141	0.8137
DeepFM	Mean Attention	Movielens	-	Yes	0.9681	0.9683
DeepFM	Mean Attention	Frappe	-	Yes	0.9845	0.9844
DNN	Mean Attention	Avazu	-	-	0.7636	0.7635
DNN	Mean Attention	Criteo	-	-	0.814	0.8136
DNN	Mean Attention	Movielens	-	-	0.9652	0.9676
DNN	Mean Attention	Frappe	-	-	0.9846	0.9844

Table A.2: Table 1 (continued)

Base Model	Attention Mechanism	DataSet	Layer Normalization	Interaction Component Feeding	Experimented Attention Mechanism AUC	Base Model AUC
DCN	Mean Max Summation Attention	Avazu	-	Yes	0.7612	0.7623
DCN	Mean Max Summation Attention	Criteo	-	Yes	0.814	0.8137
DCN	Mean Max Summation Attention	Movielens	-	Yes	0.9651	0.9686
DCN	Mean Max Summation Attention	Frappe	-	Yes	0.9704	0.9843
DeepFM	Mean Max Summation Attention	Avazu	-	Yes	0.7635	0.7642
DeepFM	Mean Max Summation Attention	Criteo	-	Yes	0.814	0.8137
DeepFM	Mean Max Summation Attention	Movielens	-	Yes	0.9583	0.9683
DeepFM	Mean Max Summation Attention	Frappe	-	Yes	0.982	0.9844
DNN	Mean Max Summation Attention	Avazu	-	-	0.7625	0.7635
DNN	Mean Max Summation Attention	Criteo	-	-	0.8136	0.8136
DNN	Mean Max Summation Attention	Movielens	-	-	0.9647	0.9676
DNN	Mean Max Summation Attention	Frappe	-	-	0.9755	0.9844
DCN	Mean Max Concat Attention	Avazu	-	Yes	0.7627	0.7623
DCN	Mean Max Concat Attention	Criteo	-	Yes	0.8138	0.8137
DCN	Mean Max Concat Attention	Movielens	-	Yes	0.9647	0.9686
DCN	Mean Max Concat Attention	Frappe	-	Yes	0.9754	0.9843
DeepFM	Mean Max Concat Attention	Avazu	-	Yes	0.763	0.7642
DeepFM	Mean Max Concat Attention	Criteo	-	Yes	0.8141	0.8137
DeepFM	Mean Max Concat Attention	Movielens	-	Yes	0.9495	0.9683
DeepFM	Mean Max Concat Attention	Frappe	-	Yes	0.9786	0.9844
DNN	Mean Max Concat Attention	Avazu	-	-	0.7631	0.7635
DNN	Mean Max Concat Attention	Criteo	-	-	0.8137	0.8136
DNN	Mean Max Concat Attention	Movielens	-	-	0.9617	0.9676
DNN	Mean Max Concat Attention	Frappe	-	-	0.9772	0.9844
DCN	Mean Std Summation Attention	Avazu	-	Yes	0.5	0.7623
DCN	Mean Std Summation Attention	Criteo	-	Yes	0.8137	0.8137
DCN	Mean Std Summation Attention	Movielens	-	Yes	0.9555	0.9686
DCN	Mean Std Summation Attention	Frappe	-	Yes	0.982	0.9843
DeepFM	Mean Std Summation Attention	Avazu	-	Yes	0.762	0.7642
DeepFM	Mean Std Summation Attention	Criteo	-	Yes	0.814	0.8137
DeepFM	Mean Std Summation Attention	Movielens	-	Yes	0.9522	0.9683
DeepFM	Mean Std Summation Attention	Frappe	-	Yes	0.9788	0.9844
DNN	Mean Std Summation Attention	Avazu	-	-	0.7623	0.7635