Operation-aware Neural Networks for User Response Prediction

### 

**Abstract** The proliferation of online advertising and recommendation systems has made accurate prediction of user responses (e.g., click-through rate, conversion rate) a critical component of modern digital platforms. Traditional approaches, such as Factorization Machines (FM) and their extensions, capture pairwise feature interactions but remain shallow, while recent deep neural network models like PNN and DeepFM share a single embedding across multiple interaction operations, limiting their flexibility. In "Operation-aware Neural Networks for User Response Prediction," the authors propose a novel architecture—ONN—which assigns separate embeddings to each feature-operation pair, enabling the model to learn operation-specific representations. Through extensive experiments on large-scale real-world datasets, ONN achieves state-of-the-art performance in both offline and online settings, demonstrating faster convergence and improved predictive accuracy. This report provides a detailed critical analysis of the paper’s methodology, experiments, and contributions, and reflects on the practical lessons learned from implementing and studying this work.

### **1. Introduction**

Accurate user response prediction, such as click-through rate (CTR) and conversion rate (CVR), lies at the heart of online advertising and recommendation systems. The primary challenge in this domain stems from the extremely high-dimensional and sparse nature of categorical features—attributes like user demographics, ad content, and query context—that must be transformed into dense representations for modeling. Earlier solutions, such as Factorization Machines (FM) and Field-aware Factorization Machines (FFM), explicitly model pairwise interactions but lack depth and struggle to generalize beyond simple linear combinations. More recent deep learning models—e.g., Product-based Neural Networks (PNNs), DeepFM, and xDeepFM—leverage neural architectures to capture higher-order interactions but still rely on a **shared embedding** for each feature, regardless of the operation applied (e.g., inner product, outer product, or copying). This uniform embedding forces the model to compromise between conflicting interaction patterns, limiting its expressiveness. The ONN framework addresses this limitation by introducing **operation-aware embeddings**: each feature is associated with a distinct embedding vector for each type of operation it participates in. By disentangling representations in this manner, ONN improves the model’s capacity to learn nuanced feature interactions and achieves superior performance with faster convergence, particularly on large-scale industrial datasets.

### **2. Related Work**

* **Factorization Machines & Extensions**: GM and FFM learn pairwise interactions via latent embeddings but remain linear in their combination of features.
* **Deep Neural Models**: PNN, DeepFM, and xDeepFM introduce neural architectures to model interactions of higher order but continue to share a single embedding per feature, restricting their flexibility.
* **Multi-sense Embeddings**: Drawn from NLP, context-aware embeddings demonstrate the utility of multiple representations per word; ONN adapts this idea to feature-operation contexts.

### **3. Operation-aware Neural Network (ONN) Architecture**

**3.1 Operation-aware Embedding:** ONN replaces the single embedding matrix ViV\_i for feature ii with separate sub-embeddings Vi,oV\_{i,o} for each operation oo. A mapping function Γ(o,i)\Gamma(o,i) selects the appropriate vector during feature interaction computation (see Figure 2).  
 **3.2 Feature Extraction:** Two classes of operations—copy and inner-product—generate incipient features: copy operations directly return Vi,copyV\_{i,\text{copy}}, while inner-product operations compute ⟨Vi,inner,Vj,inner⟩\langle V\_{i,\text{inner}} , V\_{j,\text{inner}} \rangle.  
 **3.3 MLP & Prediction:** The concatenated incipient features feed into a three-layer MLP (batch-norm, ReLU) culminating in a sigmoid output. Log-loss is minimized during training.

### **4. Theoretical Connections**

* **FM & FFM:** FM is a special case of ONN with a single inner-product operation; FFM resembles ONN restricted to field-aware inner products.
* **PNN & DeepFM:** ONN generalizes PNN by learning operation-specific embeddings instead of reusing a shared vector across all operations.

### **5. Experiments**

**Datasets:** Criteo (45M samples) and Tencent Ads (22M train / 2M test).  
 **Baselines:** FM, FFM, DNN, PNN, DeepFM.  
 **Metrics:** Log-loss, AUC, RMSE, Pearson’s R.  
 **Results:** ONN outperforms all baselines with AUC gains of ~0.001 on Criteo and ~0.002 on Tencent Ads (Tables 1–2). In online A/B tests, ONN converges faster and yields higher click rates (Figure 7).

### **6. What We Learned**

Implementing ONN reinforces several crucial insights about modeling user response prediction tasks. First, **disentangling feature representations per operation** significantly enhances the model’s ability to capture diverse interaction patterns, leading to measurable gains in accuracy and convergence speed. Second, drawing parallels from **multi-sense word embeddings** in NLP highlights a general principle: when a single embedding is forced to serve multiple roles, its capacity is diluted. Third, the **modular design** of ONN—where new operations can be added by simply introducing new sub-embeddings and computational logic—facilitates extensibility for future interaction functions. Finally, practical experience shows that **batch normalization** often provides more stable learning in large-scale CTR models than dropout, an implementation detail that can significantly affect performance.

**Key Takeaways:**

* Operation-aware embeddings unlock richer feature interactions without manual feature engineering.
* Modular operation design allows easy extension to novel interaction types.
* Applying NLP-inspired multi-sense representation concepts to recommender systems is both feasible and beneficial.

### 

### 

### **7.Our Work**

Dataset: <https://www.kaggle.com/datasets/arashnic/ctr-in-advertisement/versions/1>







