DeepFM

A Factorization-Machine based Neural Network for CTR prediction

The proposed model, DeepFM, combines the power of factorization machines for recommendation and deep learning for feature learning in a new neural network architecture.

Here, the model able to emphasizes oth low- and high-order feature interactions.

> All existing models like **Wide and Deep model**, **FNN(Factorization-Machine Based Neural Network), PNN(Product based Neural Network),** are biased to low- or high order feature interaction, or rely on feature engineering. In this paper, we show it is possible to derive learning model that is able to learn feature interactions of all orders in an end to-end manner, without any feature engineering besides raw features.

Summariztion:

1. DeepFM integrates the architectures of FM and deep neural networks (DNN). It models low-order feature interactions like FM and models high-order feature interactions like DNN. Unlike the wide & deep model , DeepFM can be trained end to-end without any feature engineering.

Approach:

1. Input:
    a. $X = [X_{field_1}, X_{field_2}, ...X_{field_j}, ...X_{field_m}]$.
    - Filed can be categorical (gender, location) and continuous fields(age). where categorical field is represented by one-hot vector or continuous field can be represented as one hot vector after discretization.
    - X is high-dimensinal and extremely sparse.

2. Output

$$y^{`} = CTR - model(x)$$

Architecture

1. FM Component

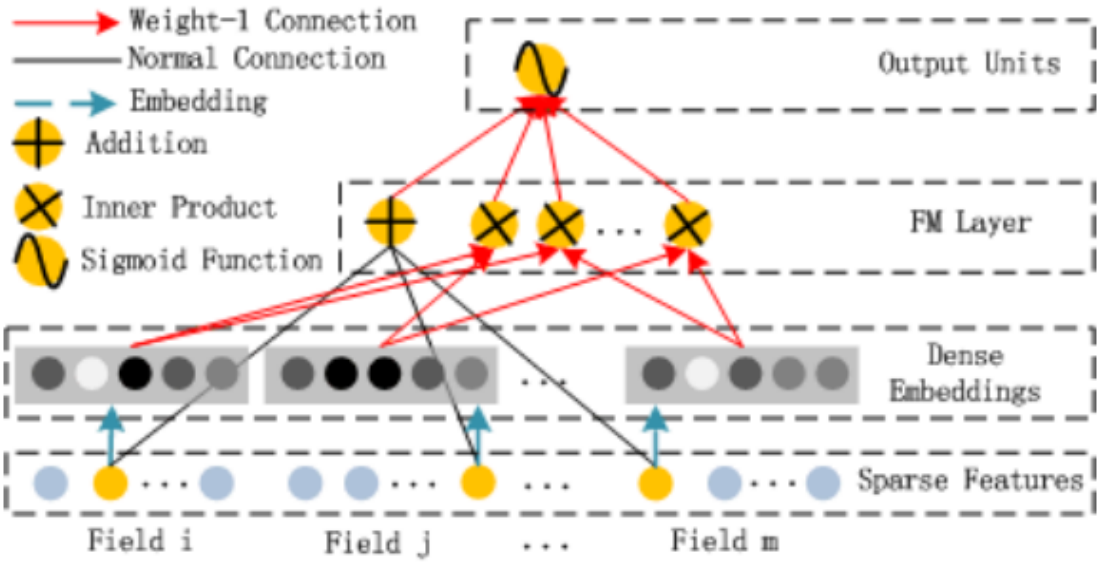


Figure 2: The architecture of FM.

2. Deep Component

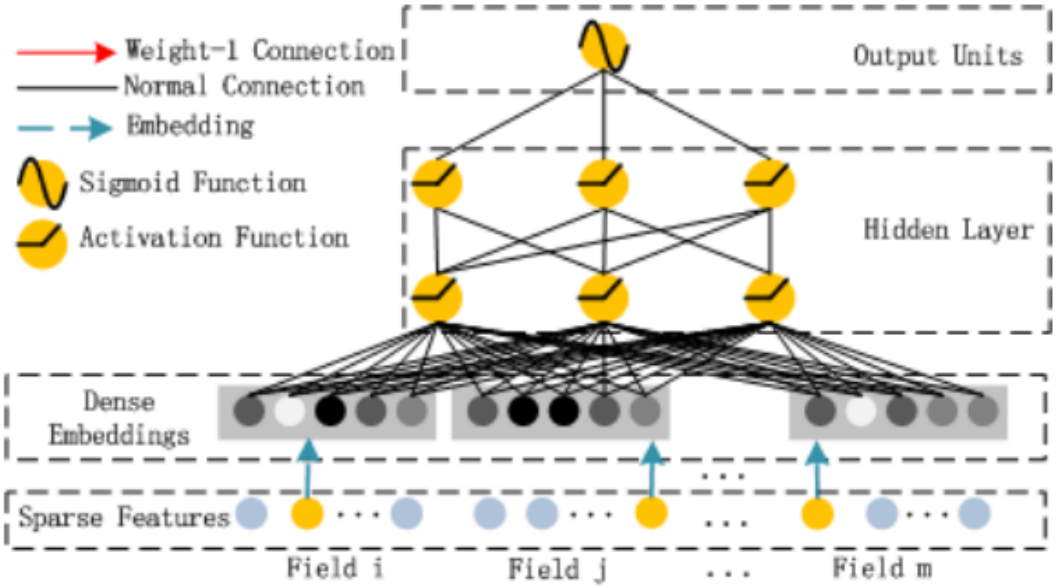
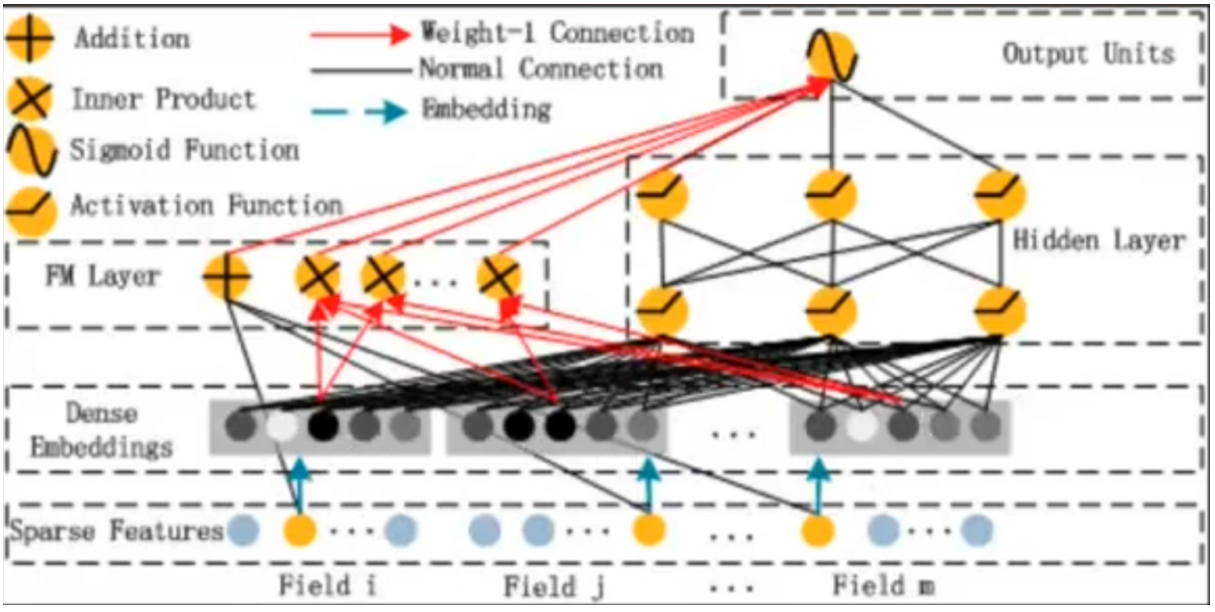
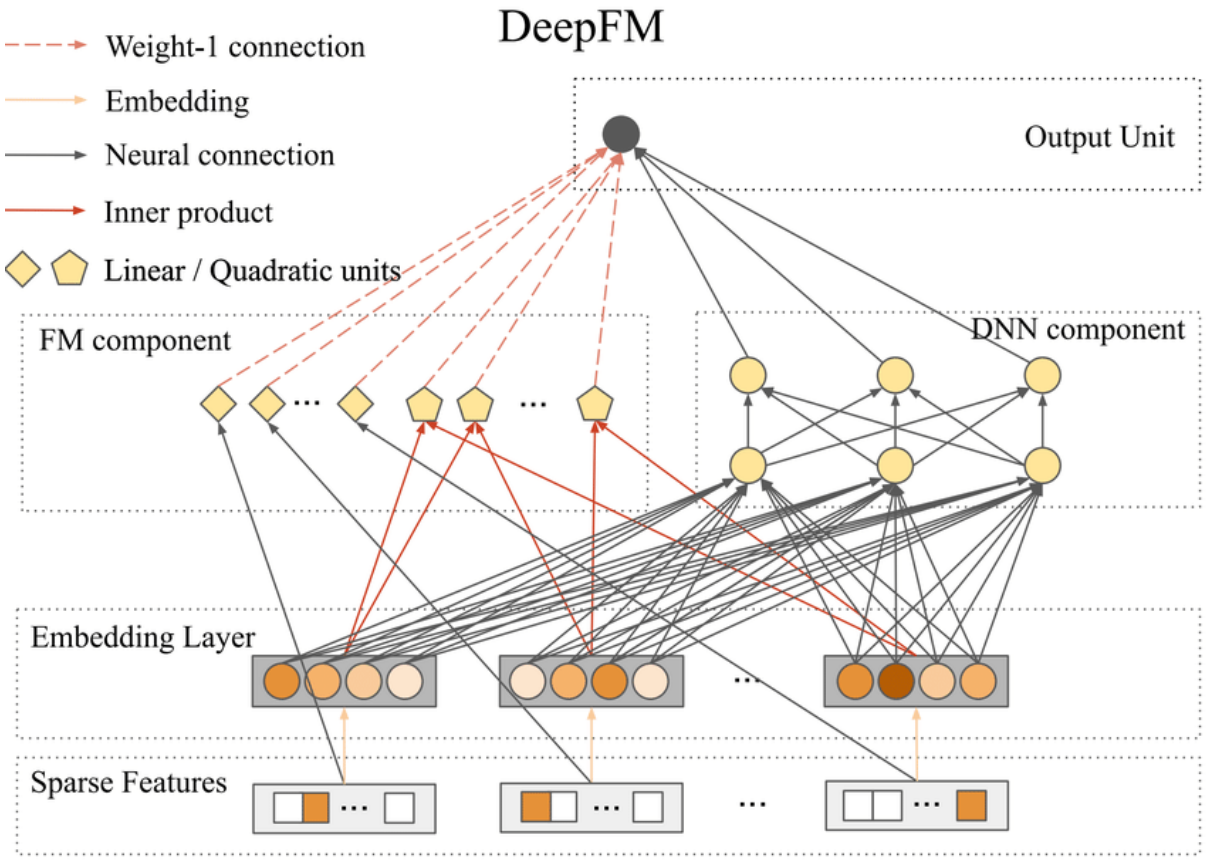


Figure 3: The architecture of DNN.

DeepFM





Structure of Embedding Layer

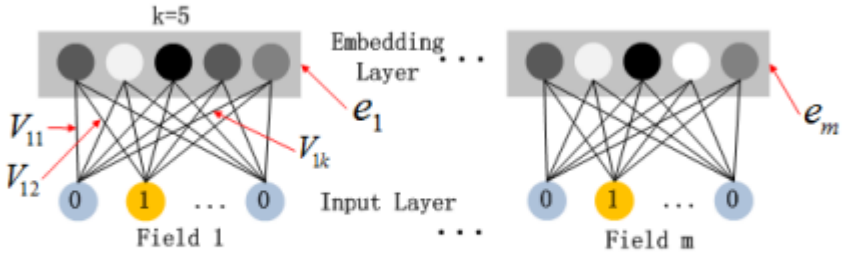


Figure 4: The structure of the embedding layer

Interesting Features of this network structure:

1. Length of input field vectors can be different, but their embeddings are of same size k.
2. The latent vectors(V) in FM now serve as network weights which are learned and used to compress the input fileld vectors to the embedding vectors.

3. FM component and Deep component share the same feature embedding, which brings two important benefits:
    a. It learns low and high order feature interactions from raw features.
    b. There is no need for expertise feature engineering of the input as required in Weed and Deep model.

Output:

$$\hat{y} = sigmoid(y_{FM} + y_{DNN}),$$