Evaluación e implementación de DeepFM

Tomas Borchers
Pontificia Universidad Catolica de Chile
Santiago, Chile
tborchers@uc.cl

Juan Pablo de Vicente
Pontificia Universidad Catolica de Chile
Santiago, Chile
jpdevicente@uc.cl

ABSTRACT

Maximizar el Click-Through-Rate se ha vuelto esencial para sostener una empresa hoy en día, y Guo et. al. [5] ha realizado un modelo que aprovecha la interacción entre una Factorization Machine y una DNN para asi sacar el beneficio de ambas y superar a la competencia. En este documento se utilizó el modelo en datasets con características distintas al paper original para comprobar la eficiencia del modelo. La diferencia que este dataset tiene pocas entradas por usuario. El rendimiento del DeepFM, fue inferior a la Factorization Machine pero superior a la DNN. Por lo que se puede concluir que si se tienen pocos datos por usuarios, la red neuronal no funciona eficientemente bajando el rendimiento de la DeepFM.

1 INTRODUCCIÓN

La publicidad es algo que se ve en el día a día. Esto permite que el publico se entere de nuevos productos que les puede ser útil y permite que las empresas sean descubiertas. Con la aparición del internet, esto también cambio la manera de hacer anuncios. Estos aparecen durante su uso diario y se pueden cliquear para obtener mayor información sobre el producto publicitado. También con este click, la pagina web que hospeda la publicidad recibe beneficios. Por lo tanto, predecir la probabilidad con la cual un usuario hará click en un anuncio, le permite a un sitio maximizar sus ganancias, dado que ofrecerá los anuncios que tienen mayor probabilidad de ser cliqueados. A la vez, los usuarios se verán beneficiados dado que se les ofrecerá anuncios que van acorde a sus gustos e intereses.

Debido a esto, se han diseñado diversos métodos que buscan predecir el CTR a través de información recopilada por los proveedores de anuncios. Uno de estos métodos, es DeepFM [5], el cual busca combinar la capacidad de detectar relaciones de bajo nivel de las máquinas de factorización (FM) con las relaciones de alto nivel que las redes neuronales profundas (DNN) son capaces de encontrar.

Este modelo tuvo resultados superiores a su competencia pero solo se realizó en un dataset, por lo que en este documento se revisará la eficiencia del modelo con datasets de distintas características al usado en el paper original.

2 DATASET

Se investigó la forma del dataset de Criteo utilizado en el paper original, y se encontró que este consistía de 400 millones de datos con 40 entradas promedio por usuario. Entonces se decidió utilizar datasets donde se encuentren pocas entradas por usuarios. Los datasets elegidos son los siguientes:

Avazu-CTR: [1] un dataset ofrecido por Avazu, una plataforma de anuncios para dispositivos móviles. Entre los campos que este dataset ofrece, se encuentran 8 variables categóricas sobre el anuncio, información del sitio donde se encontraba el anuncio e información del dispositivo que vio el anuncio, además de una variable que indica si el usuario del dispositivo hizo click o no en el anuncio. Contiene 999999 entradas, y el número promedio de entradas por usuario es de 3.19 con una desviación estándar de 21.39.

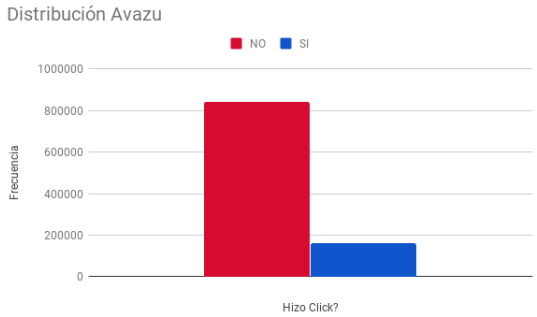

Figure 1: Distribución de datos del dataset Avazu [1]

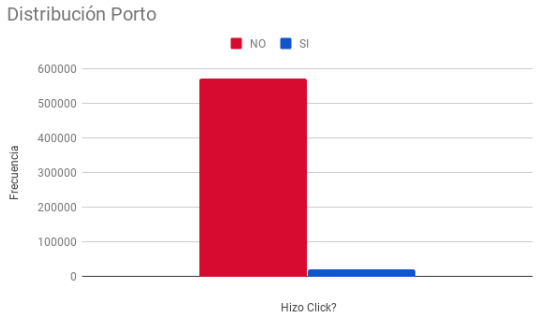

Figure 2: Distribución de datos del dataset Porto Seguro [2]

Porto Seguro's Safe Driver Prediction: [2] Un dataset entregado por la compañía de seguros automovilísticos Porto Seguro con el cual se busca predecir la probabilidad con la que un usuario tomara una póliza de seguro para su automóvil. Entre los parámetros de este dataset, se encuentran distintas variables tanto categóricas como binarias de los autos. Contiene 595215 entradas, pero cada usuario tiene una sola entrada. A pesar de que este dataset no es exactamente de CTR, la forma y distribución que tienen los datos más la forma que tiene la predicción, hacen que este dataset se comporte de la misma forma que uno de CTR.

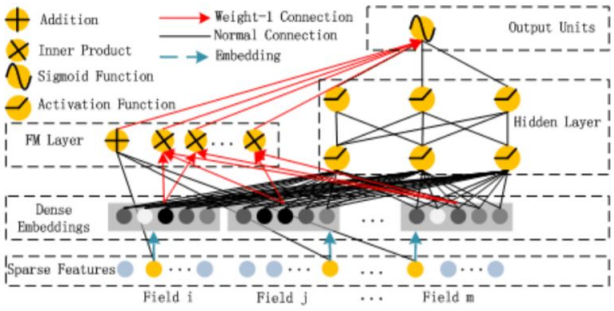

Figure 3: DeepFM [5]

3 MODELO

DeepFM es un modelo entrenable end-to-end el cual se compone de dos partes:

Componente FM: consiste en una máquina de factorización la cual además de contener interacciones de primer orden entre las features, modela también interacciones de segundo orden a través del producto entre los respectivos vectores latentes de las features. Gracias a esto, interacciones entre features que pueden aparecer pocas veces en el dataset, pueden ser captados por FM.

Componente Profundo: consiste de una red neuronal densa que busca detectar relaciones de alto nivel entre las features.

Ambas partes comparten un embedding denso, el cual es entrenado en conjunto con las otras partes del modelo.

Finalmente, ambas la sección de orden 1 y 2 de la FM en conjunto a la salida de la red neuronal densa, son alimentadas a una neurona final con una sigmoide como función de activación.

3.1 Baselines

Factorization Machine: Se utilizó el componente FM del modelo, manteniendo los mismos parámetros del modelo original, de esta forma podemos revisar cuanto aporta al resultado las relaciones de bajo nivel.

Deep Neural Network: Se utilizó el componente DNN del modelo, manteniendo los mismos parámetros del modelo original, de esta forma podemos revisar cuanto aporta al resultado las relaciones de alto nivel.

Básicamente se utilizó los componentes FM y la DNN separados, pero manteniendo los mismos parámetros que en el modelo combinado. Esto, con el fin de evaluar como mejora el rendimiento cuando ambas partes se combinan en el modelo completo DeepFM.

4 MÉTRICAS DE EVALUACIÓN

Para evaluar la eficiencia del modelo se utilizaron las siguientes métricas:

Log Loss: es una métrica de precisión que mide la precisión del modelo penalizando los falsos positivos y premiando la seguridad al predecir. Mientras más bajo sea este valor, mejor será el modelo.

$$LL = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log p_i + (1-y_i)\log(1-p_i)]$$

Donde N es le numero de instancias, y_i es la clasificación del modelo (si el usuario hizo click) y p_i la probabilidad de clasificar la instancia i como y_i.

ROC AUC: es una métrica de precisión que mide el área bajo una curva ROC. Una curva ROC mide el ratio entre predicciones correctas e incorrectas, donde el eje y es el ratio de predicciones correctas con respecto al total de predicciones y el eje x las predicciones incorrectas con respecto al total de las predicciones. Por ende mientras mayor su valor significa que clasificó mayores instancias de manera correcta.

Intralist Diversity: es una métrica de diversidad que mide que tan distintos son los anuncios recomendados entre si. Se eligió los 10 anuncios más probable que nos recomiende para cada usuario de los datos a testear y medimos su rendimiento. Mientras más bajo es el valor más diversas son las recomendaciones.

$$ILS_u = \frac{1}{2}\sum_{i\in L}\sum_{j\in L} sim(i,j)$$

Donde u es el usuario y L es la lista de los 10 mejores anuncios para el usuario.

Para la similitud usamos similitud coseno:

$$sim(x,y) = \frac{x \cdot y}{||x|| \cdot ||y||}$$

Novelty: para medir la novedad (la novedad es la probabilidad de recomendar un anuncio que el usuario no haya visto.) se utilizó la métrica propuesta por Pablo Castells *et al* [4] la cual es:

$$Novelty_u = -\sum_{i\in L} p(i|L)\cdot \log_2 p_i$$

Donde u es el usuario, L es la lista de los 10 mejores anuncios para el usuario, $p(i|L)$ es la probabilidad de recomendar el anuncio dado L y p_i es la probabilidad de recomendar el anuncio para el público general.

Mientras mas alto el valor, hay mas probabilidad de mostrar anuncios nuevos.

5 ANÁLISIS DE PARÁMETROS

Se evaluó el modelo de DeepFM para distintos números de neuronas, capas de neuronas y tamaño del embedding para ver el efecto de cada parámetro en el resultado. Se tomo como resultado el mejor rendimiento que obtuvo el modelo hasta la época 20.

Se eligió Avazu como el dataset sobre el cual se evaluarían los parámetros debido a su cantidad de datos y numero de entradas por usuario.

Avazu	30 neuronas	100 neuronas	150 neuronas
Log loss	0.3998	0.3992	0.4000
ROC AUC	0.7135	0.7130	0.7101
ILS	0.9956	0.9970	0.9947
Novelty	0.4905	0.4283	0.4842

Table 1: DeepFM para distinto numero de neuronas con 3 capas y embedding de 12

Avazu	2 capas	3 capas	4 capas
Log loss	0.4013	0.3992	0.4005
ROC AUC	0.7078	0.7130	0.7125
ILS	0.9963	0.9970	0.9956
Novelty	0.4856	0.4283	0.4937

Table 2: DeepFM para distinto numero de capas con 100 neuronas y embedding de 12

Avazu	Embedding de 8	Embedding de 12	Embedding de 20
Log loss	0.4017	0.3992	0.3992
ROC AUC	0.7116	0.7130	0.7131
ILS	0.9970	0.9970	0.9962
Novelty	0.4859	0.4283	0.4234

Table 3: DeepFM para distintos tamaños del embedding con 3 capas de 100 neuronas

Analizando estos resultados, se decidió dejar los siguientes parámetros para el modelo final de DeepFM:

- Tamaño del embedding de 12.
- 3 capas neuronales densas de 100 neuronas cada una para el componente deep.
- Dropout de 0.3 y 0.5 para el componente FM y deep respectivamente.

6 RESULTADOS

Para DeepFM se utilizo como base la implementación en github de Chen [3]. Para evaluar el componente FM y deep por su cuenta, se recortó la componente que no se quería considerar del modelo para evaluar la otra.

Los resultados que se han obtenido hasta ahora para la versión reducida de Avazu se pueden ver en la tabla 1, y los resultados para Porto seguro se encuentran en la tabla 2.

Avazu	DeepFM	FM	DNN
Log loss	**0.3992**	0.3995	0.4292
ROC AUC	0.7130	**0.7147**	0.6970
ILS	**0.9970**	0.9949	0.9950
Novelty	0.4283	**0.4750**	0.3853

Table 4: Resultados Avazu-CTR

Porto	DeepFM	FM	DNN
Log loss	0.1525	**0.1524**	0.1558
ROC AUC	0.6309	**0.6324**	0.5714
ILS	0.9992	**0.9993**	0.9990
Novelty	**0.4950**	0.4622	0.2318

Table 5: Resultados Porto seguro

7 ANÁLISIS

Al analizar los resultados obtenidos, podemos ver que para el dataset reducido de Avazu, DeepFM obtuvo el mejor resultado en log loss, pero este es casi idéntico a lo que obtuvo FM. Es más, la componente FM tuvo un rendimiento considerablemente mejor en todas las otras métricas, por lo que podemos concluir que FM dio los mejores resultados en Avazu. Especialmente en novedad.

Paralelamente para el caso de Porto seguro, se observo que FM nuevamente obtuvo los mejores resultados excepto en novedad, donde DeepFM lo supero. De todas formas, cabe mencionar que la novedas es una métrica que va de 0 a 0.5, por lo que un resultado de 0.4622 para FM sigue siendo un buen rendimiento en novedad.

Finalmente, se puede observar que en ambos casos DeepFM tuvo una tendencia a comportarte de igual forma que FM, solo que con un poco peor rendimiento, mientras que la DNN dio los peores resultados en ambos datasets.

Si se analizan estos resultado teniendo en cuenta la naturaleza de los datasets es fácil identificar la causa. La componente deep del modelo busca identificar las relaciones de alto nivel entre las features para cada usuario, por lo que cuando se tienen pocas entradas por usuario, como es en el caso de los dos datasets usados para evaluar, esta es incapaz de recolectar suficiente información para aprender relaciones útiles. He aquí porque el modelo DNN obtuvo considerablemente peores resultados en ambos datasets. Por otra parte, la componente FM que busca encontrar relaciones de bajo orden para las predicciones, puede hacerlo incluso cuando se tienen pocos datos de cada usuario.

8 CONCLUSIONES

A partir de los resultados y su análisis se puede concluir que para datasets en que se tienen pocos datos por usuario, es mejor no utilizar una componente deep. Esto debido a que las relaciones de alto nivel que las redes neuronales profundas son especialistas en capturar, no se alcanzan a reconocer cuando tenemos poca información de cada usuario, por lo que esta termina bajando el rendimiento de DeepFM en comparación a utilizar solo la componente FM. En cambio, cuando si se tiene una cantidad considerablemente de entradas por usuario, DeepFM si logra aprovechar su componente Deep, obteniendo así un mejor rendimiento que FM tal como se puede observar en Guo et al[5] para el dataset de Criteo, donde se tienen aproximadamente 40 datos por usuario.

9 TRABAJO FUTURO

Una posible forma de mejorar los resultados y poder aprovechar la parte deep para pocas muestras por usuario, es la de utilizar Bayesian Personalized Ranking (BPR) al final del modelo. Entregándole a este una entrada con target positivo (En el caso de los datasets

evaluados, un anuncio donde el usuario hizo click o una póliza de seguro contratada) mas una entrada donde se desconoce el target.

Otra posible cambio al modelo seria el de reemplazar la red neuronal profunda densa por una red neuronal recurrente y entregarle los datos en forma cronológica. Esto con el objetivo de que la red sea capaz de predecir la probabilidad del siguiente dato de ser gustado o no, conociendo aquellos datos que previamente fueron gustados por el usuario. Esto le daría a la red cierta capacidad de memoria que podría mejorar los resultados del modelo incluso cuando se tienen pocos datos por usuario.

REFERENCES

[1] 2018. https://www.kaggle.com/c/avazu-ctr-prediction

[2] 2018. https://www.kaggle.com/c/porto-seguro-safe-driver-prediction

[3] 2018. https://github.com/ChenglongChen/tensorflow-DeepFM

[4] P. Castells, S. Vargas, and J. Wang. 2011. Novelty and Diversity Metrics for Recommender Systems: Choice, Discovery and Relevance. In *International Workshop on Diversity in Document Retrieval (DDR 2011) at the 33rd European Conference on Information Retrieval (ECIR 2011)*. Dublin, Ireland. http://ir.ii.uam.es/rim3/publications/ddr11.pdf

[5] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. *ArXiv e-prints* (March 2017). arXiv:cs.IR/1703.04247