

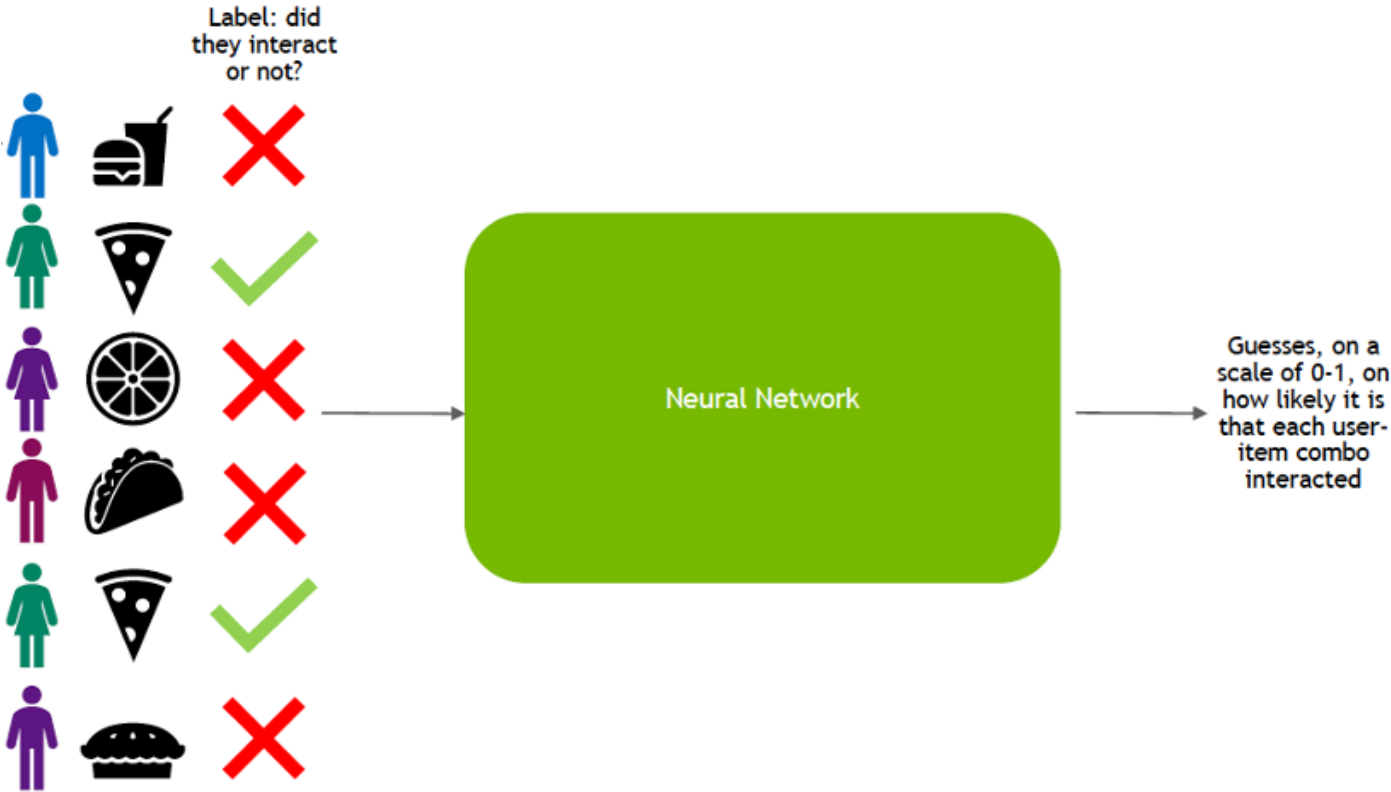
Recommenders / Personalization

How to Build a Deep Learning Powered Recommender System, Part 2

May 02, 2021

+8 Like Discuss (0)

By [Carol McDonald](#)



Recommender systems (RecSys) have become a key component in many online services, such as e-commerce, social media, news service, or online video streaming. However with the growth in importance, the growth in scale of industry datasets, and more sophisticated models, the bar has been raised for computational resources required for recommendation systems.

To meet the computational demands for large-scale deep learning recommender systems, NVIDIA introduced Merlin – a Framework for Deep Recommender Systems. Now NVIDIA teams have won two consecutive RecSys competitions in a row: the ACM RecSys Challenge 2020, and more recently the WSDM WebTour 21 Challenge organized by Booking.com. The Booking.com challenge focused on predicting the last city destination for a traveler’s trip given their previous booking history within the trip. NVIDIA’s interdisciplinary team included colleagues from NVIDIA’s KGMON (Kaggle Grandmasters), NVIDIA’s RAPIDS (Data Science), and NVIDIA’s Merlin (Recommender Systems) who collaborated on the winning solution.

This post is the second installment of a three-part series of articles on recommender systems. The first post gives an overview of recommender system concepts. This post discusses deep learning for recommender systems. The third post will discuss the winning solution, the steps involved, and also what made a difference in the outcome.

NVIDIA uses cookies to deliver and improve the website experience. See our [Cookie Policy](#) to learn more.

Cookies Settings

Accept All Cookies

moving from more traditional machine learning methods to highly expressive deep learning models to improve the quality of their recommendations.

Broadly, the life-cycle of deep learning for recommendation can be split into two phases: training and inference. In the training phase, the model is trained to predict user-item interaction probabilities (calculate a preference score) by presenting it with examples of interactions (or non-interactions) between users and items from the past.

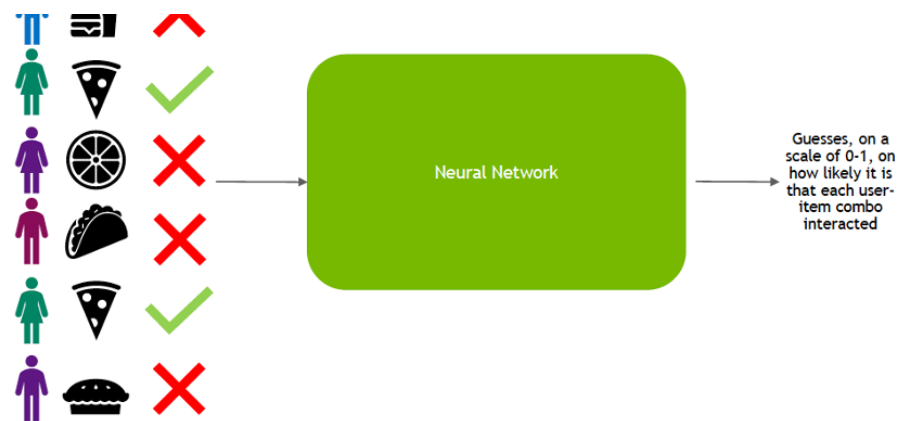


Figure 1: Deep learning for recommendation training.

Once it has learned to make predictions with a sufficient level of accuracy, the model is deployed as a service to infer the likelihood of new interactions.

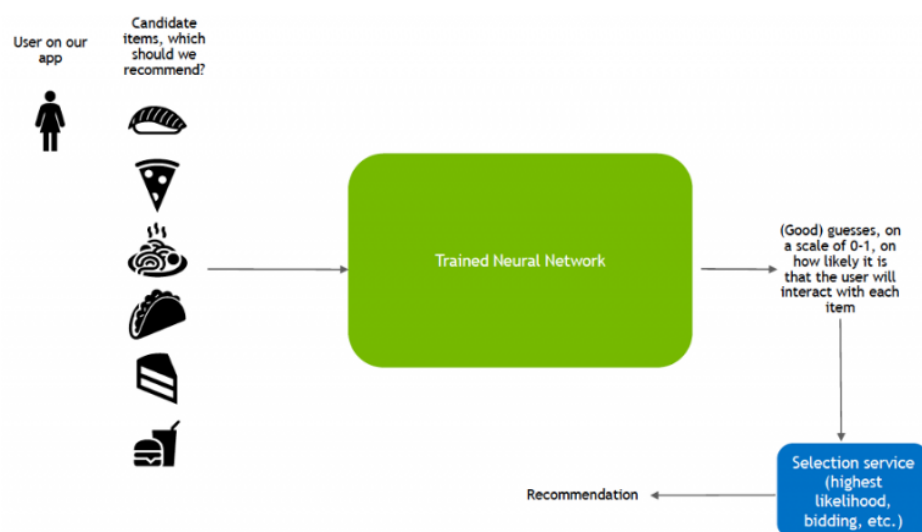
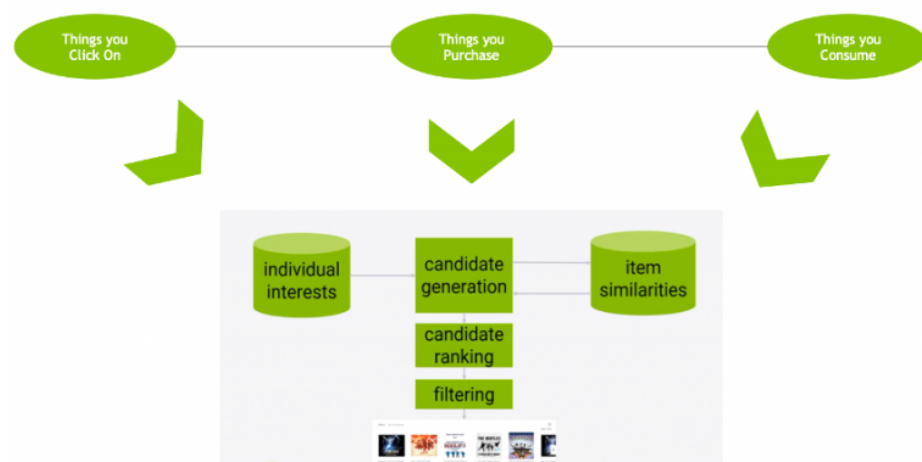


Figure 2: Deep learning for recommendation inference.

This inference stage utilizes a different pattern of data consumption than during training:

- Candidate generation: pair a user with hundreds or thousands of candidate items based on learned user-item similarity.
- Candidate ranking: rank the likelihood that the user enjoys each item.
- Filter: show the user the item they are rated most likely to enjoy.



NVIDIA uses cookies to deliver and improve the website experience. See our [Cookie Policy](#) to learn more.

Deep Neural Network Models for Recommendation

Deep learning (DL) recommender models build upon existing techniques such as factorization to model the interactions between variables and embeddings to handle categorical variables. An embedding is a learned vector of numbers representing entity features so that similar entities (users or items) have similar distances in the vector space. For example, a deep learning approach to collaborative filtering learns the user and item embeddings (latent feature vectors) based on user and item interactions with a neural network.

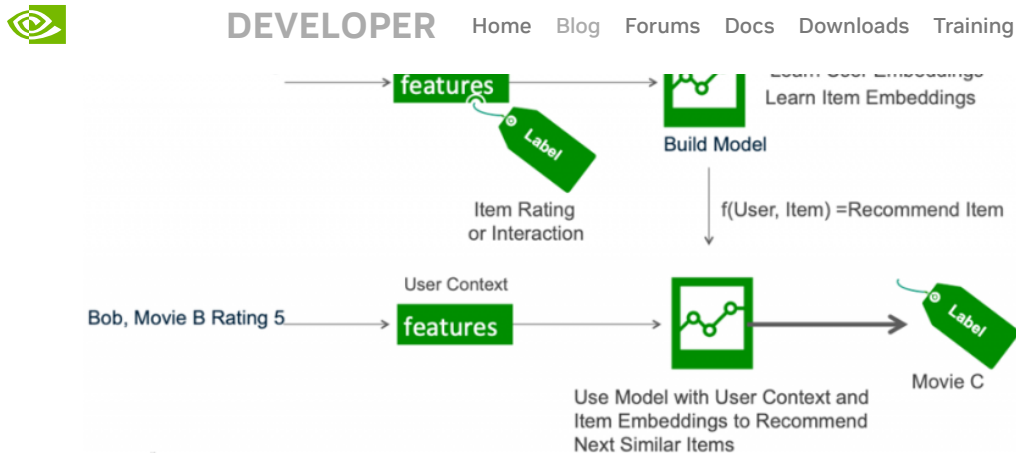


Figure 4: A deep learning approach to collaborative filtering learns the user and item embeddings based on user and item interactions.

DL techniques also tap into the vast and rapidly growing novel network architectures and optimization algorithms to train on large amounts of data, use the power of deep learning for feature extraction, and build more expressive models. DL-based models build upon the different variations of artificial neural networks (ANNs), such as the following:

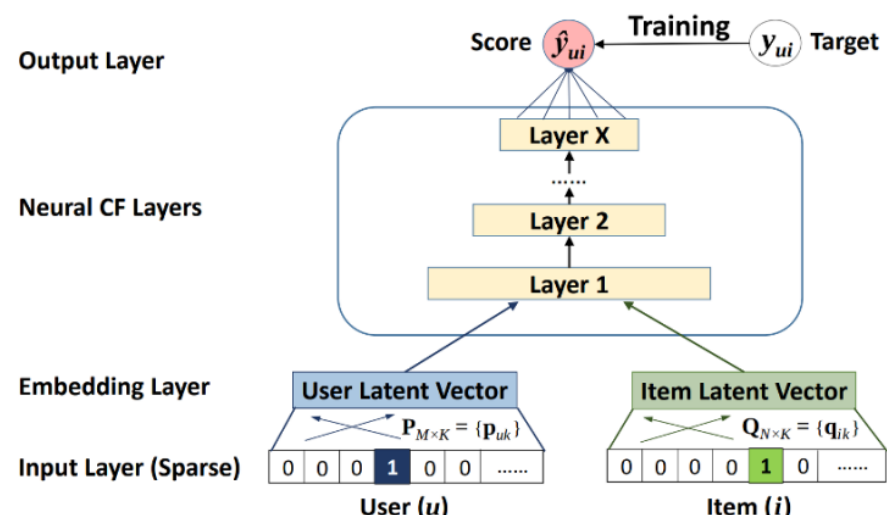
- Feedforward neural networks are ANNs where information is only fed forward from one layer to the next.
- Multilayer perceptrons (MLPs) are a type of feedforward ANN consisting of at least three layers of nodes: an input layer, a hidden layer, and an output layer. MLPs are flexible networks that can be applied to a variety of scenarios.
- Convolutional Neural Networks are the image crunchers to identify objects.
- Recurrent neural networks are the mathematical engines to parse language patterns and sequenced data.

GPUs, with their massively parallel architecture, are driving the advancement of deep learning (DL) and RecSys DL in the past several years. With GPUs, you can exploit data parallelism through columnar data processing instead of traditional row-based reading designed initially for CPUs. This provides higher performance and cost savings. Current DL-based models for recommender systems like DLRM, Wide and Deep (W&D), Neural Collaborative Filtering (NCF), Variational AutoEncoder (VAE) are part of the NVIDIA GPU-accelerated DL model portfolio that covers a wide range of network architectures and applications in many different domains beyond recommender systems, including image, text and speech analysis.

Neural Collaborative Filtering

The Neural Collaborative Filtering (NCF) model is a neural network that provides collaborative filtering based on user and item interactions. The NCF model treats matrix factorization from a non-linearity perspective. NCF TensorFlow takes in a sequence of (user ID, item ID) pairs as inputs, then feeds them separately into a matrix factorization step (where the embeddings are multiplied) and into a multilayer perceptron (MLP) network.

The outputs of the matrix factorization and the MLP network are then combined and fed into a single dense layer which predicts whether the input user is likely to interact with the input item.



NVIDIA uses cookies to deliver and improve the website experience. See our [Cookie Policy](#) to learn more.

An autoencoder neural network reconstructs the input layer at the output layer by using the representation obtained in the hidden layer. An autoencoder for collaborative filtering learns a non-linear representation of a user-item matrix and reconstructs it by determining missing values.

The NVIDIA GPU-accelerated Variational Autoencoder for Collaborative Filtering (VAE-CF) is an optimized implementation of the architecture first described in Variational Autoencoders for Collaborative Filtering. VAE-CF is a neural network that provides collaborative filtering based on user and item interactions. The training data for this model consists of pairs of user-item IDs for each interaction between a user and an item.

The model consists of two parts: the encoder and the decoder. The encoder is a feedforward, fully connected neural network that transforms the input vector, containing the interactions for a specific user, into an n-dimensional variational distribution. This variational distribution is used to obtain a latent feature representation of a user (or embedding). This latent representation is then fed into the decoder, which is also a feedforward network with a similar structure to the encoder. The result is a vector of item interaction probabilities for a particular user.

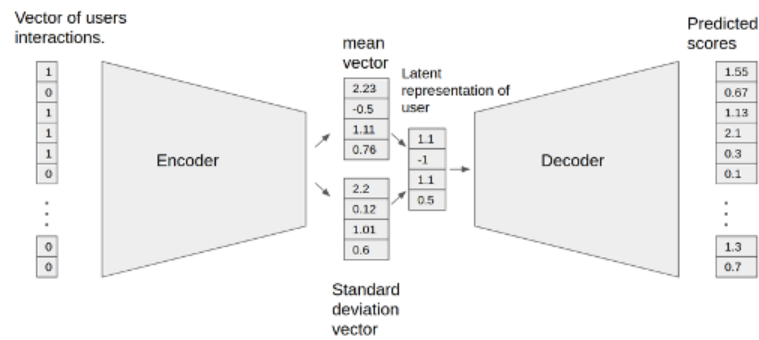


Figure 6: VAE-CF model.

Wide and Deep

Wide & Deep refers to a class of networks that use the output of two parts working in parallel—wide model and deep model—whose outputs are summed to create an interaction probability. The wide model is a generalized linear model of features together with their transforms. The deep model is a Dense Neural Network (DNN), a series of hidden MLP layers, each beginning with a dense embedding of features. Categorical variables are embedded into continuous vector spaces before being fed to the DNN via learned or user-determined embeddings.

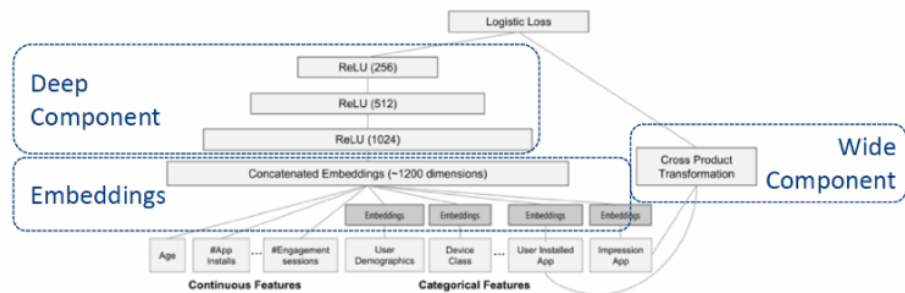
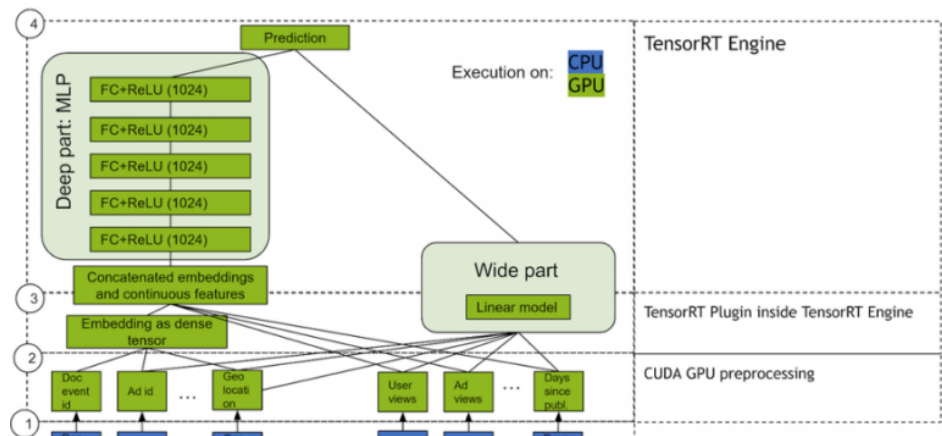


Figure 7: Wide and Deep model

What makes this model so successful for recommendation tasks is that it provides two avenues of learning patterns in the data, “deep” and “shallow”. The complex, nonlinear DNN is capable of learning rich representations of relationships in the data and generalizing to similar items via embeddings but needs to see many examples of these relationships in order to do so well. The linear piece, on the other hand, is capable of “memorizing” simple relationships that may only occur a handful of times in the training set.

In combination, these two representation channels often end up providing more modeling power than either on its own. NVIDIA has worked with many industry partners who reported improvements in offline and online metrics by using Wide & Deep as a replacement for more traditional machine learning models.



NVIDIA uses cookies to deliver and improve the website experience. See our [Cookie Policy](#) to learn more.

DLRM

DLRM is a DL-based model for recommendations introduced by Facebook [research](#). It’s designed to make use of both categorical and numerical inputs that are usually present in recommender system training data. To handle categorical data, embedding layers map each category to a dense representation before being fed into multilayer perceptrons (MLP). Numerical features can be fed directly into an MLP.

At the next level, second-order interactions of different features are computed explicitly by taking the dot product between all pairs of embedding vectors and processed dense features. Those pairwise interactions are fed into a top-level MLP to compute the likelihood of interaction between a user and item pair.

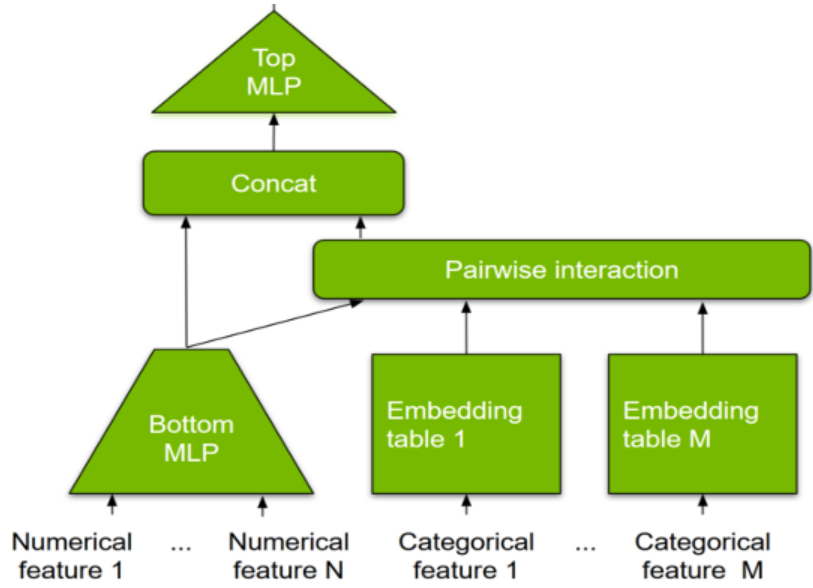


Figure 9: DLRM model.

Compared to other DL-based approaches to recommendation, DLRM differs in two ways. First, it computes the feature interaction explicitly while limiting the order of interaction to pairwise interactions. Second, DLRM treats each embedded feature vector (corresponding to categorical features) as a single unit, whereas other methods (such as Deep and Cross) treat each element in the feature vector as a new unit that should yield different cross terms. These design choices help reduce computational/memory cost while maintaining competitive accuracy.

Contextual Sequence Learning

A Recurrent neural network (RNN) is a class of neural network that has memory or feedback loops that allow it to better recognize patterns in data. RNNs solve difficult tasks that deal with context and sequences, such as natural language processing, and are also used for contextual sequence recommendations. What distinguishes sequence learning from other tasks is the need to use models with an active data memory, such as LSTMs (Long Short-Term Memory) or GRU (Gated Recurrent Units) to learn temporal dependence in input data. This memory of past input is crucial for successful sequence learning. Transformer deep learning models, such as BERT (Bidirectional Encoder Representations from Transformers), are an alternative to RNNs that apply an attention technique—parsing a sentence by focusing attention on the most relevant words that come before and after it. Transformer-based deep learning models don’t require sequential data to be processed in order, allowing for much more parallelization and reduced training time on GPUs than RNNs.

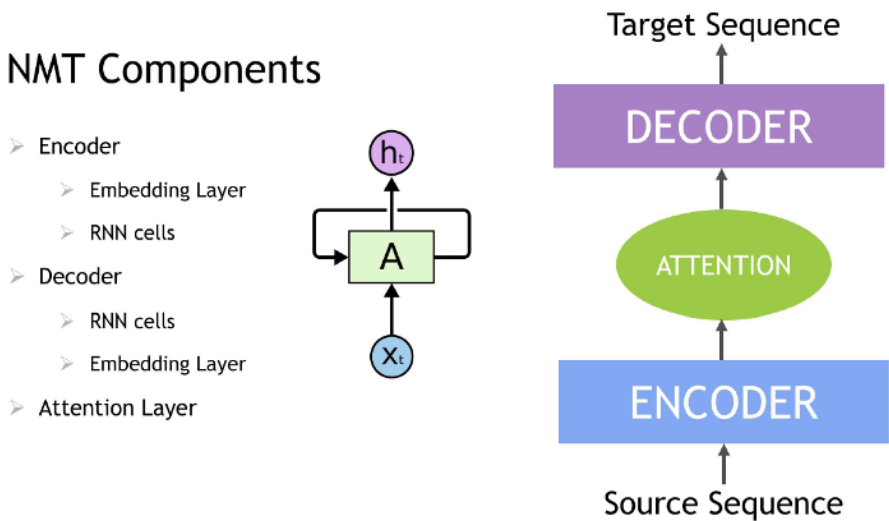


Figure 10: Neutral machine translation Model.

NVIDIA uses cookies to deliver and improve the website experience. See our [Cookie Policy](#) to learn more.

semantics and contextual information for each word, so that similar words are close to each other in this number space, and dissimilar words are far apart. These DL models provide an appropriate output for a specific language task like next-word prediction and text summarization, which are used to produce an output sequence.

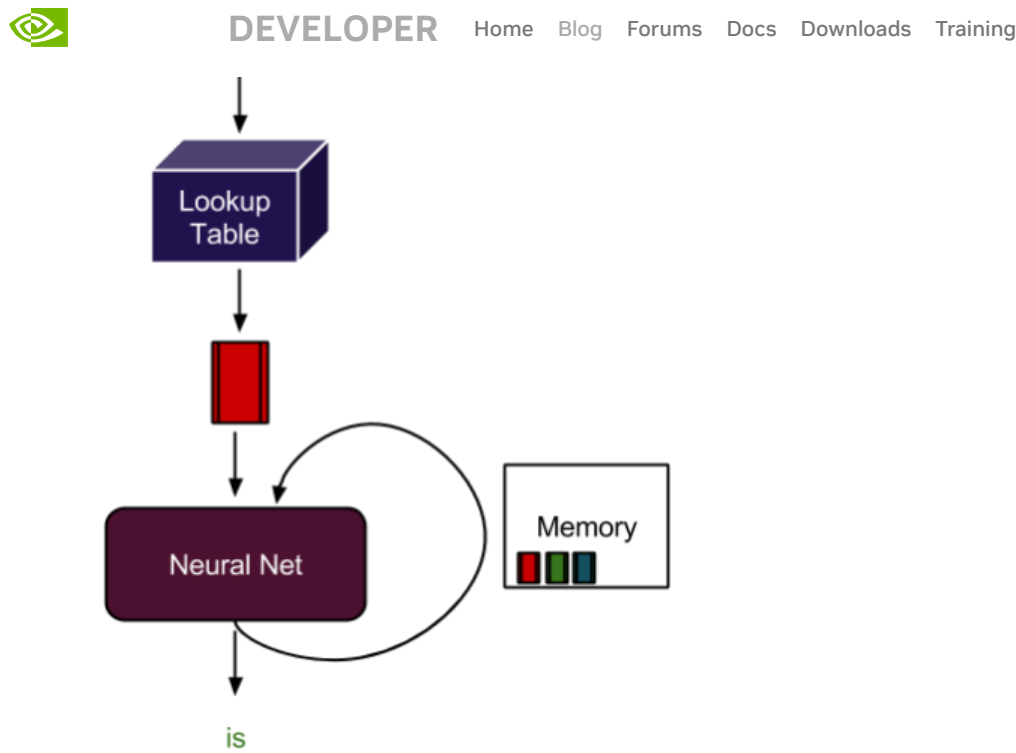


Figure 11: A recurrent neural network has memory of past experiences. The recurrent connection preserves these experiences and helps the network keep a notion of context.

Session-based recommendations apply the advances in sequence modeling from deep learning and NLP to recommendations. RNN models train on the sequence of user events in a session (e.g. products clicked, date, and time of interactions) in order to predict the probability of a user clicking the candidate or target item. User item interactions in a session are embedded similarly to words in a sentence before being fed into RNN variants such as LSTM, GRU, or Transformer to understand the context. For example, Square's deep learning-based product recommendation system shown below leverages the transformer-based model BERT, GRUs, and NVIDIA GPUs to create a vector representation of their sellers.

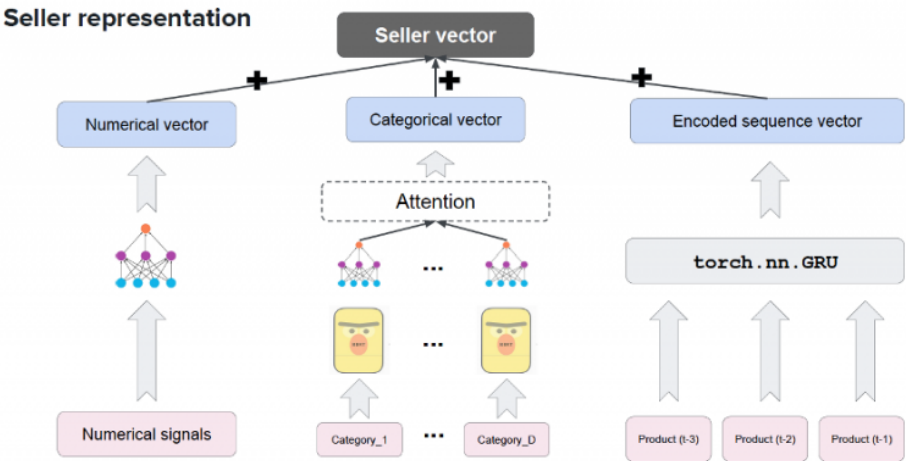
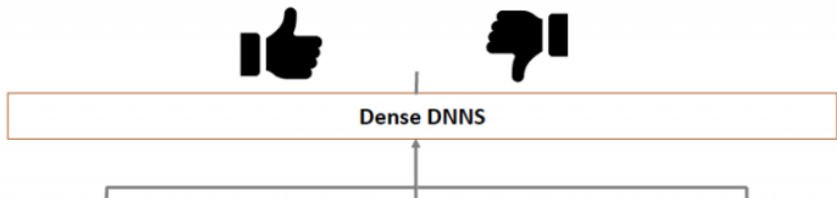


Figure 12: Square's model architecture leverages the transformer-based model BERT and GRUs to create the vector representation of their sellers.

Alibaba also uses a model architecture with DNNs, GRUs and NVIDIA GPUs to support its e-commerce recommendation system which has a catalog of two billion products and can serve as many as 500 million customers per day.



NVIDIA uses cookies to deliver and improve the website experience. See our [Cookie Policy](#) to learn more.

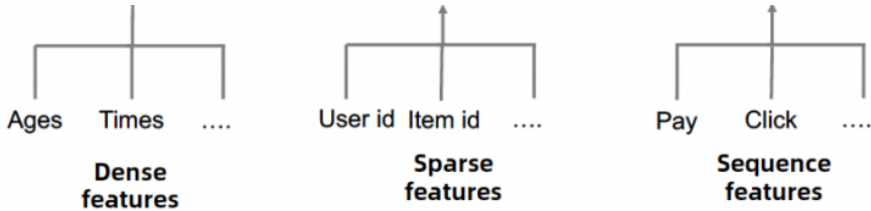



Figure 13: Alibaba's recommender system model architecture.

In the more detailed model diagram below, you can see that GRUs are added to learn and capture the relations among the items in the user behavior sequences in order to predict if a user will click on an advertised product.

HugeCTR +

Subscribe >

Accelerating Embedding with the HugeCTR TensorFlow



DEVELOPER

Home Blog Forums Docs Downloads Training

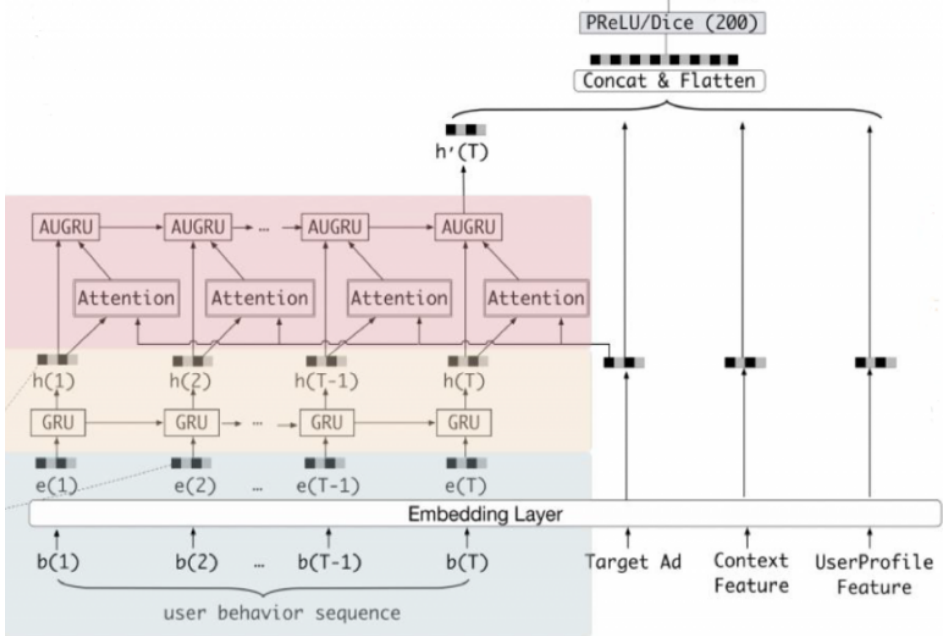


Figure 14: Alibaba's recommender system model architecture uses GRUs to capture user sequence behavior.

Alibaba is using thousands of T4 GPUs across its infrastructure with TensorRT to support the entire recommendation query AI pipeline on a real-time basis.

Session based recommender system architectures such as Alibaba's Behavior Sequence Transformer follow the same general transformer architecture as for NLP, but model and embedding sizes between NLP and recommender systems vary significantly which means that you need to make sure that the entire recommendation AI pipeline is well tuned for the use case.

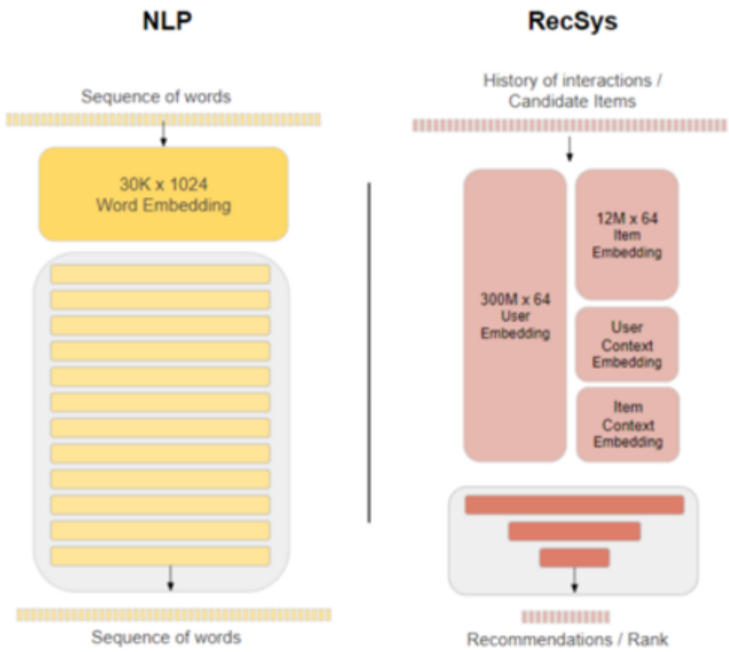


Figure 15: Model and embedding sizes between NLP and recommender systems vary significantly.

NVIDIA GPU Accelerated, End-to-End Data Science

NVIDIA developed RAPIDS™—an open-source data analytics and machine learning acceleration platform—for executing end-to-end data science training pipelines completely in GPUs. It relies on NVIDIA® CUDA® primitives for low-level compute optimization, but exposes that GPU parallelism and high memory bandwidth through user-friendly Python interfaces.

NVIDIA uses cookies to deliver and improve the website experience. See our [Cookie Policy](#) to learn more.

pipelines—from data prep to machine learning to deep learning. RAPIDS also includes support for multi-node, multi-GPU deployments, enabling vastly accelerated processing and training on much larger dataset sizes.

Compared to similar CPU-based implementations, RAPIDS delivers 50x performance improvements for classical data analytics and machine learning (ML) processes at scale which drastically reduces the total cost of ownership (TCO) for large data science operations.

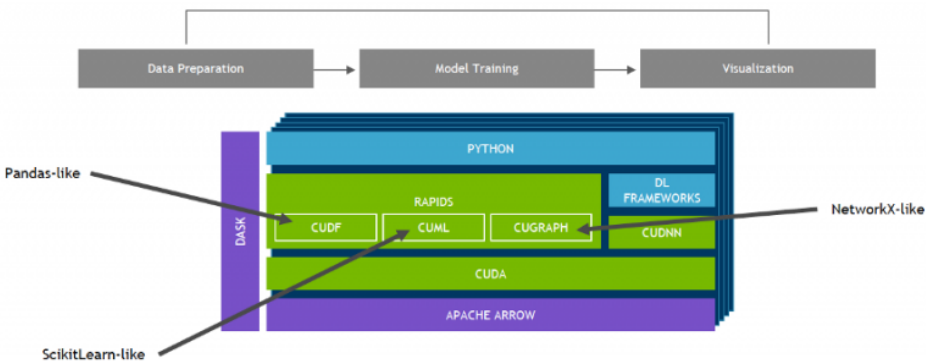


Figure 16: End-to-End Data science pipeline with GPUs and RAPIDS.

Embedding Plugin



How to Build a Winning Deep Learning Powered Recommender System-Part 3



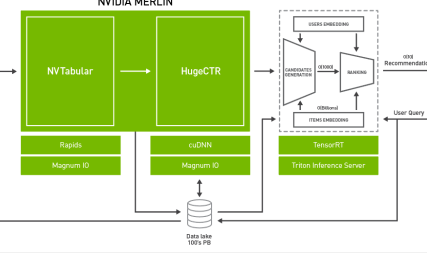
User

How to Build a Winning Recommendation System, Part 1



Learn How to Build Intelligent Recommender Systems

NVIDIA MERLIN



Announcing NVIDIA Merlin: An Application Framework for Deep Recommender Systems

NVIDIA Merlin

NVIDIA [Merlin](#) is an open-source application framework for building high-performance, DL-based recommender systems, built on NVIDIA RAPIDS™, NVIDIA CUDA® Deep Neural Network library (cuDNN), and Triton. Merlin facilitates and accelerates recommender systems on GPU, speeding up common ETL tasks, training of models, and inference serving by ~10x over commonly used methods.

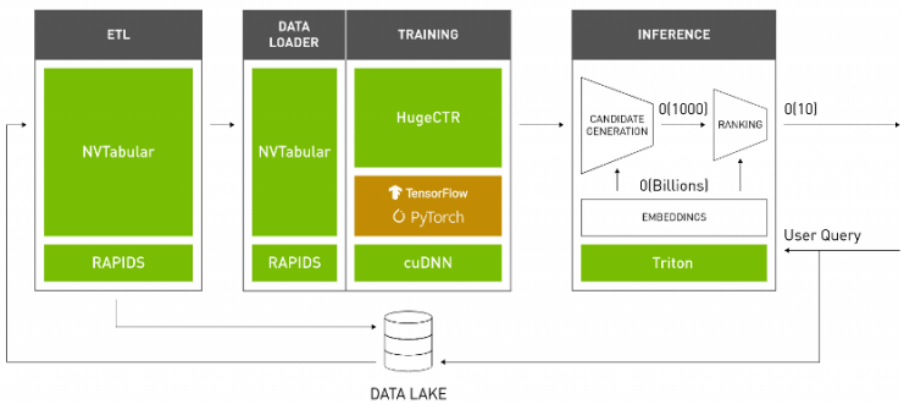


Figure 17: NVIDIA Merlin Open Beta Recommender System Framework.

[NVTabular](#) is a feature engineering and preprocessing library for recommender systems. It provides a high-level abstraction to simplify code and accelerates computation on the GPU using the [RAPIDS](#) GPU-accelerated DataFrame cuDF library.

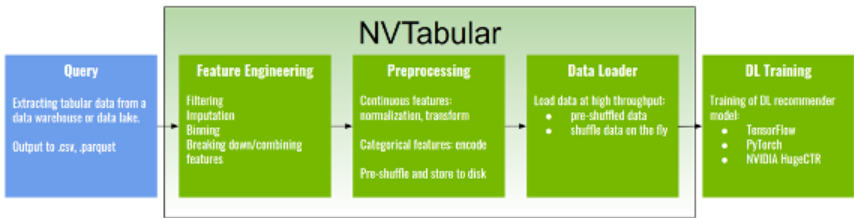
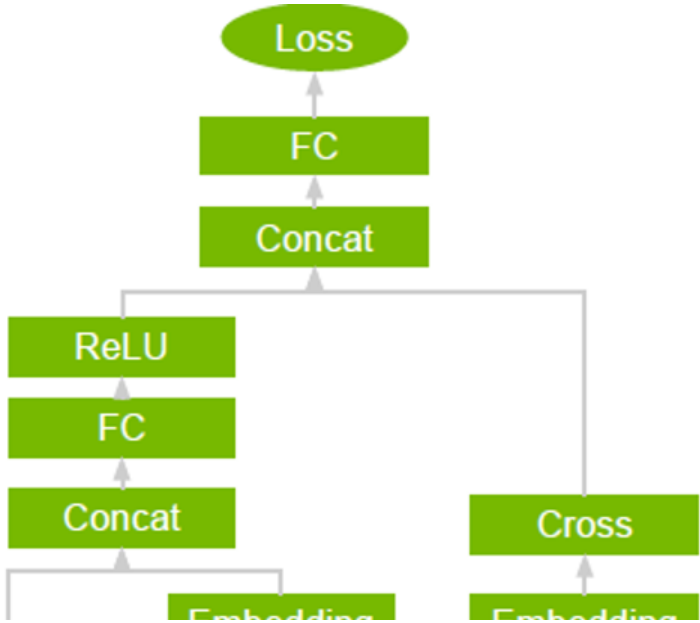


Figure 18: Recommender system training pipeline with NVTabular.

[HugeCTR](#) is a GPU-accelerated deep neural network training framework designed to distribute training across multiple GPUs and nodes. It supports state-of-the-art hybrid model-parallel embedding tables and data-parallel neural networks and their variants, such as [Wide and Deep Learning \(WDL\)](#), [Deep Cross Network \(DCN\)](#), [DeepFM](#), [xDeepFM](#), [Variational Autoencoder \(VAE\)](#), and [Deep Learning Recommendation Model \(DLRM\)](#).



NVIDIA uses cookies to deliver and improve the website experience. See our [Cookie Policy](#) to learn more.

Figure 19: An example model expressible by HugeCTR: A hybrid model with two embeddings and two different types of inputs.

NVIDIA Triton™ Inference Server and NVIDIA® TensorRT™ accelerate production inference on GPUs for feature transforms and neural network execution.

Beyond providing better performance, these libraries are also designed to be easy to use and integrate with existing recommendation pipelines.

Conclusion

In this blog, we gave an overview of deep learning models for recommender systems. Part three will discuss the NVIDIA teams’ winning solution for the ACM WSDM WebTour 21 Challenge organized by Booking.com.

Next steps

- Go to the [Merlin home page](#)
- Read:
 - [Introduction to NVIDIA Merlin](#)
 - [Like Magic: NVIDIA Merlin Gains Adoption for Training and Inference](#)



- [NVIDIA Merlin Accelerates Recommender Workflows with .4 Release](#)
- [Achieving High-Quality Search and Recommendation Results with DeepNLP](#)
- [NVIDIA GPUs Accelerate World's Biggest Online Shopping Event](#)
- Watch GTC sessions: [NVIDIA Deepens Commitment to Streamlining Recommender Workflows with GTC Spring Sessions](#)
- NVIDIA Deep Learning Institute: [Building Intelligent Recommender Systems](#)

Related resources

- **DLI course:** Building Intelligent Recommender Systems
- **GTC session:** Serving Large Recommender Models with 10x Performance Gain (Spring 2023)
- **GTC session:** Merlin Updates - Build and Deploy Recommender Systems at Any Scale (Spring 2023)
- **GTC session:** Building Session-Based Recommendation Models with NVIDIA Merlin* (Spring 2023)
- **Webinar:** NVIDIA NGC Jupyter Notebook Day: Recommender System
- **Webinar:** Develop and Optimize Deep Learning Recommender Systems

Discuss (0)

+8 Like

Tags

[Computer Vision / Video Analytics](#) | [Data Science](#) | [Recommenders / Personalization](#) | [Tutorial](#) | [Featured](#)

About the Authors



About Carol McDonald

Carol McDonald is a product marketing manager focusing on Spark and data science. Carol has experience in many roles, including technical marketing, software architecture and development, training, technology evangelism, and developer outreach. Carol writes industry architectures, best practices, patterns, prototypes, tutorials, demos, blog posts, whitepapers, and ebooks. She has traveled worldwide, speaking and giving hands-on labs; and has developed complex, mission-critical applications in the banking, health insurance, and telecom industries. Carol holds an MS in computer science from the University of Tennessee and a BS in geology from Vanderbilt University. Carol is fluent in English, French, and German.

View all posts by Carol McDonald >

Comments

Start the discussion at forums.developer.nvidia.com

The In-Person GTC Experience Is Back

March 18-21 | San Jose

Stay Informed

NVIDIA uses cookies to deliver and improve the website experience. See our [Cookie Policy](#) to learn more.