**Tên bài báo: A Neural Autoregressive Appoarch to Collaborative Filtering (2016)**

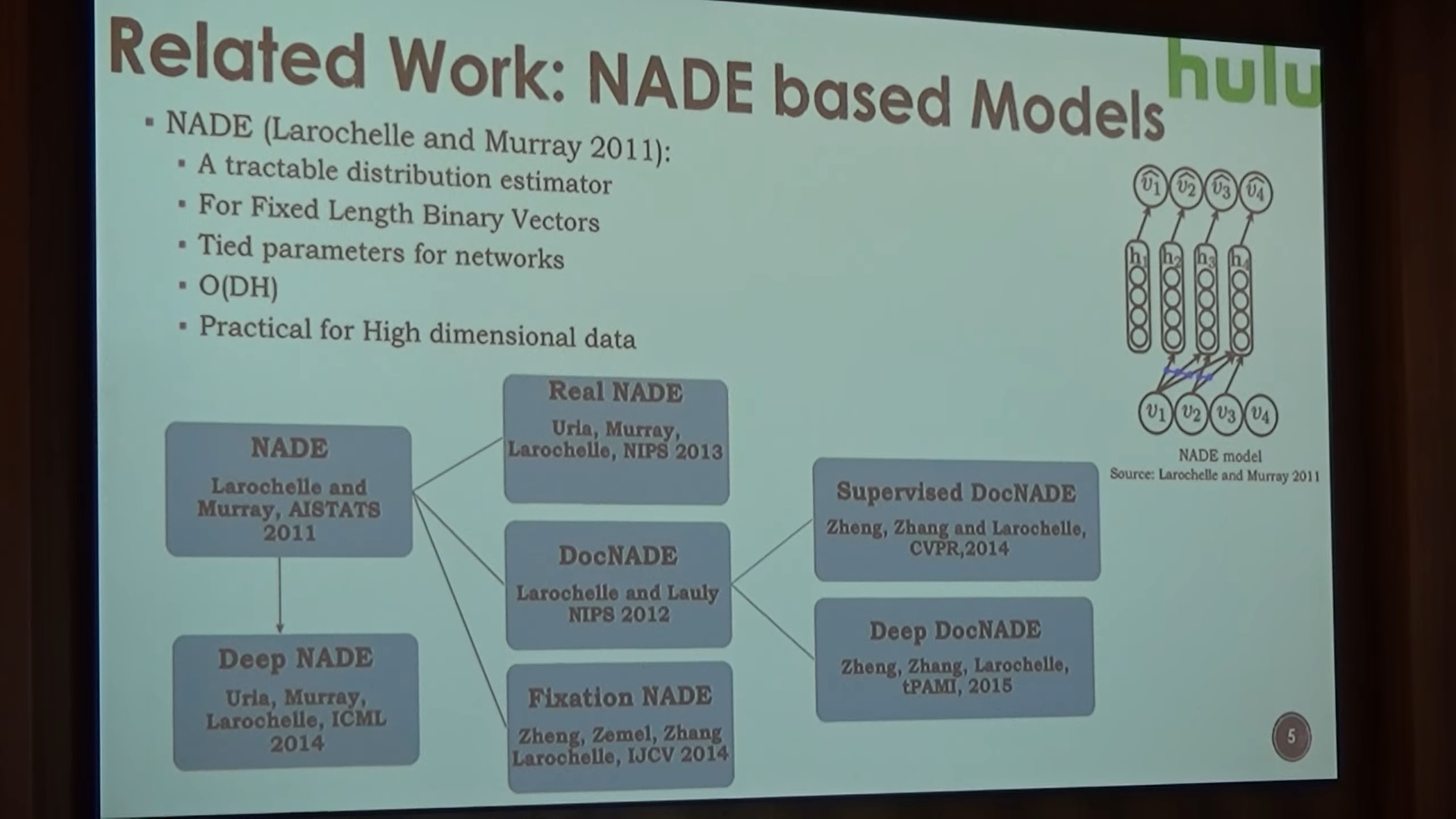
1. **Paper**

* **Giới thiệu:**

Có hai loại CF: model-based CF và memory-based CF. Memory-based CF là dựa vào dữ liệu cũ. Ta tính được mối tương quan giữa các item hoặc user và từ mối tương quan đó sẽ đưa ra tư vấn cho người dùng. Model-based CF là xây dựng mô hình dựa trên dữ liệu đã có ở quá khứ và từ đó cho mô hình dự đoán với dữ liệu mới.

CF-NADE thuộc memory-based.

* **Công trình liên quan**: một số công trình cùng loại như: RBM-CF (RBM-CF tốn nhiều thời gian để huấn luyện hơn là CF-NADE), NNMF (Neural Network Matrix Factorication). Có một số cải tiến của NADE như hình sau:



Source: Yin Zheng @ eHarmony June 28, 2016

Có một bài báo có tên gần giống với bài báo này.

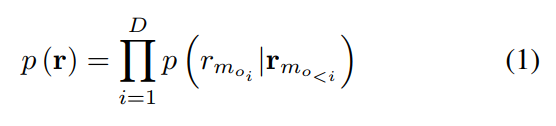
<https://link.springer.com/chapter/10.1007/978-3-319-59081-3_33>

* **Mô hình:**

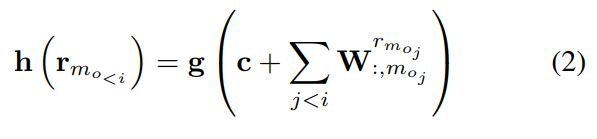
Với mỗi user sẽ có một mô hình CF-NADE. Tuy nhiên, các mô hình CF-NADE này giống y chang nhau về cấu trúc và trọng số sẽ được chia sẻ với nhau.

* + Ratings và User sẽ được đưa vào lần lượt theo thời gian ratings. Mặc dù trong khi thực nghiệm, việc đưa vào ngẫu nhiên cũng cho kết quả tốt.

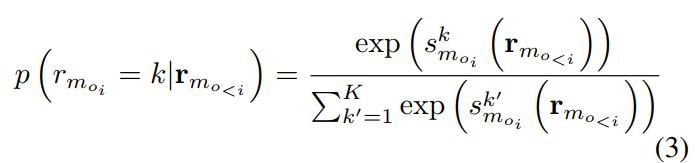
Xác suất của p(r) được tính như sau



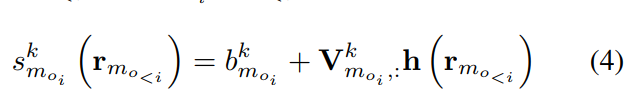
Để tính được công thức (1), ta phải tìm dựa vào công thức (2).



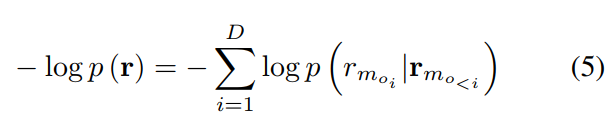
Công thức (1) được chuyển về công thức số (3).



Để tính , ta sử dụng công thức (4) với tính được ở (2).



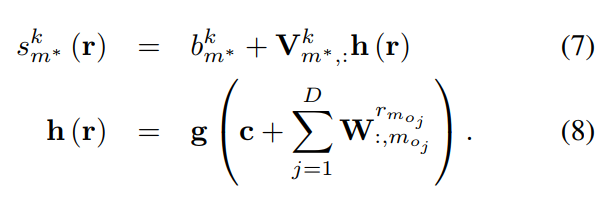
**Optimize dựa vào minimun negative likelihood của p(r)**



Dự đoán ratings dựa vào thông tin đã biết.

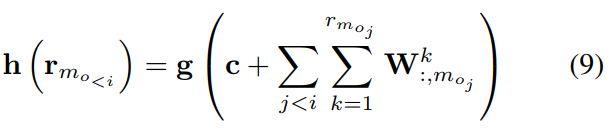


Với tính bởi (3) với h(r) và tính bởi:

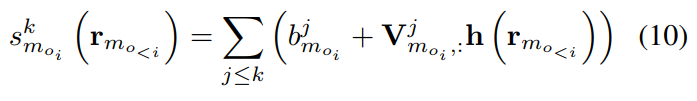


Sau đó, tác giả đã đề xuất chia sẻ tham số. bởi vì dữ liệu thưa thớt nên một mô hình cho một người dùng rất khó để tối ưu. Khi chia sẻ tham số như vậy, một số công thức sẽ sửa đổi. Như là:

Công thức tính thay đổi như sau:

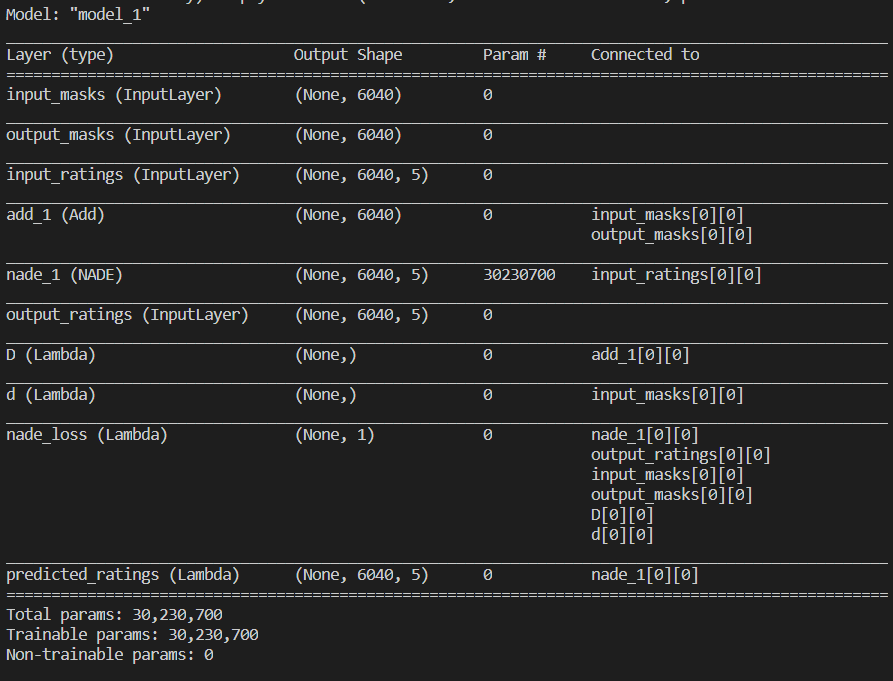


Và công thức của thay đổi như sau.



Với tập Netflix, dữ liệu lớn với 88.850.000 tham số, nên sẽ chia nhỏ các ma trận trọng số và thực hiện optimize với negative likelihood.

1. Source Code.
   * Có nhiều nguồn source code:
     + <https://github.com/Ian09/CF-NADE>: source code viết lại từ đầu các hàm optimize, sử dụng block và theano.
     + <https://github.com/JoonyoungYi/CFNADE-keras>: source code sử dụng keras để xây dựng NADE.
   * Hiện tại sv đang sử dụng source code thứ 2 để chạy. Source code bao gồm hai giai đoạn.
   * Đầu tiên file data\_prep.py sẽ thực hiện chia file ratings.dat làm 3 phần: train, val, test để có thể chạy mô hình NADE trên ba tập đó.
   * Sau đó file run.py sẽ thực hiện train NADE trên tập train, thực hiện validation trên tập val và test trên tập test. Mô hình của NADE được viết trong file nade.py. Mô hình khi chạy:



* + Kết quả thu được sau khi chạy 30 epochs gần giống với kết quả đã được nêu ra trong bài báo.