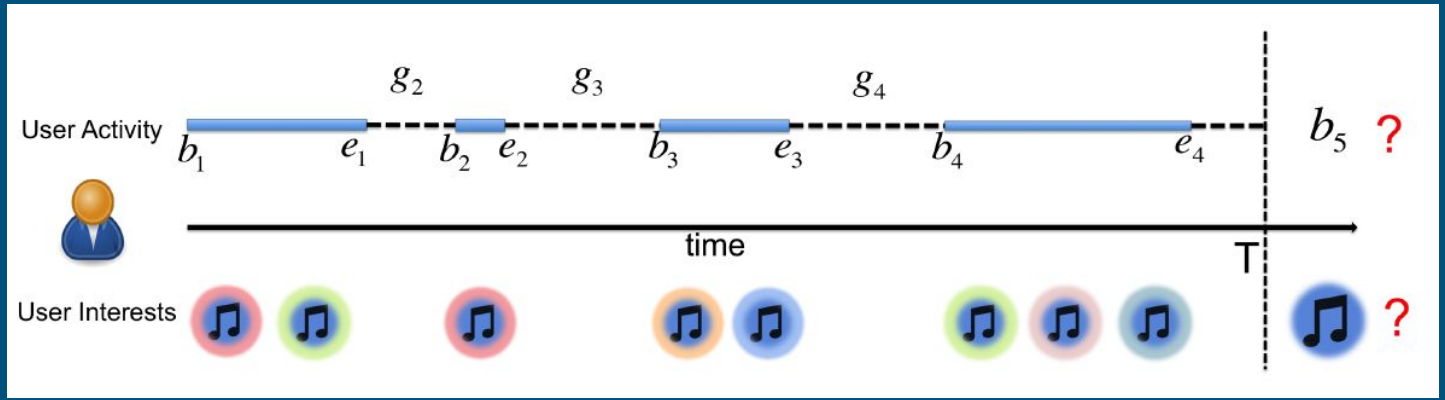
Neural Survival Recommender

How Jing; Alexander J. Smola (2017)

Contexto: Servicios Online

- Servicios como Netflix, Spotify, etc.
- Usuario de servicio se conecta en diversos momentos del tiempo al servicio→ **Sesión**.
- En cada sesión, el usuario realiza ciertas **actividades** como escuchar una canción, ver un artículo, comprar un libro, etc.



Fuente: Jing, How; Smola, Alexander J - Neural Survival Recommender. Proceedings of the Tenth ACM International Conference on Web Search and Data Mining

Objetivos

- Predecir conjuntamente **tiempo de retorno** y **acciones** de un usuario en un servicio online.
- Todo en base a la información de **sesiones previas** del usuario.
- Se basarán en una disciplina matemática llamada **Survival Analysis** y **Redes Neuronales Recurrentes** (RNN) para resolverlo.

Marco Teórico: Trabajos Relacionados

- Métodos de Factorización con modelación temporal[1][2]
  - Tiempos de variación en los meses y años. No corto plazo
- Redes recurrentes para recomendar dentro de la sesión[3]
  - No entre sesiones
- *Next basket recommendation*, predice orden de eventos futuros
  - Pero no el tiempo en que sucederán
  - FPMC[4] y HRM[5]
- Survival Analysis para recomendar un ítem entre sesiones
  - Predicen tiempo de retorno a un ítem, pero no tiempo de retorno al servicio
  - Semi-Hidden Markov Model (sHMM)[6] y Proceso Hawkes[7]
- Modelo de Redes Recurrentes para modelar secuencias[8]
  - No modelan conjuntamente el tiempo de retorno y las recomendaciones

Contribuciones

- Son capaces de modelar un problema de **Survival Analysis** de manera de poder ser resuelto por redes neuronales.
- Ofrecen una arquitectura capaz de predecir el **tiempo de retorno** y las **acciones** a tomar conjuntamente.
- Superan al estado del arte en tareas de predicción de tiempo de retorno y clasificación.

Marco Teórico: Survival Analysis

- Disciplina que intenta estimar la ***probabilidad de supervivencia de un individuo*** hasta un tiempo T.
- En nuestro caso definiremos que un usuario “muere” cuando **vuelve al servicio**, y vive cuando no está en él.
- De cara a nosotros nos va a ayudar en dos cosas:
  - Dar un sustento teórico a nuestra solución
  - Definir una función de pérdida apropiada L_{Sur}

Marco Teórico: Survival Analysis

- Permite modelar la verosimilitud de sesiones de un usuario
- Y nos dice que esa verosimilitud dependerá de una función λ(T)
  - Ésta es la tasa de supervivencia de la población en el tiempo
  - Existe una función distinta por cada usuario
  - ¡Una red neuronal será quien tratará de aproximar esta función!
- La verosimilitud está dada por la siguiente expresión:

$$p(\mathcal{S}_u|T,u) = \Big[\prod_{i=1}^{m_u} \lambda_u(b_i)\Big] \exp\Big(-\sum_{i-1}^{m_u}\int_{e_i-1}^{b_i}\lambda_u(t)dt - \int_{e_{m_u}}^{T}\lambda_u(t)dt\Big)$$

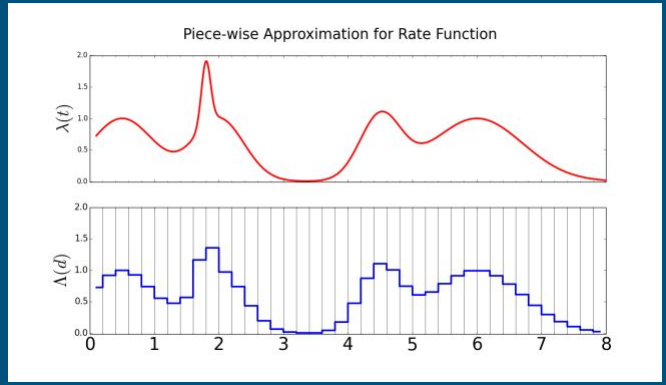
Fuente: Jing, How; Smola, Alexander J - Neural Survival Recommender. Proceedings of the Tenth ACM International Conference on Web Search and Data Mining

Marco Teórico: Survival Analysis

- Función de Pérdida para entrenar algoritmo:

$$L_{sur}(S,T) \approx \sum_{u}\sum_{i=1}^{m_u}\Big(\sum_{t=e_{i-1}}^{t\le b_i;t+=\delta t}\lambda_u(t)\delta t - \log\lambda_u(b_i)\Big) + \sum_{t=e_{m_u}}^{t\le T;t+=\delta t}\lambda_u(t)\delta t$$

- Aproximación de $\lambda(T)$, $\Lambda(t)$

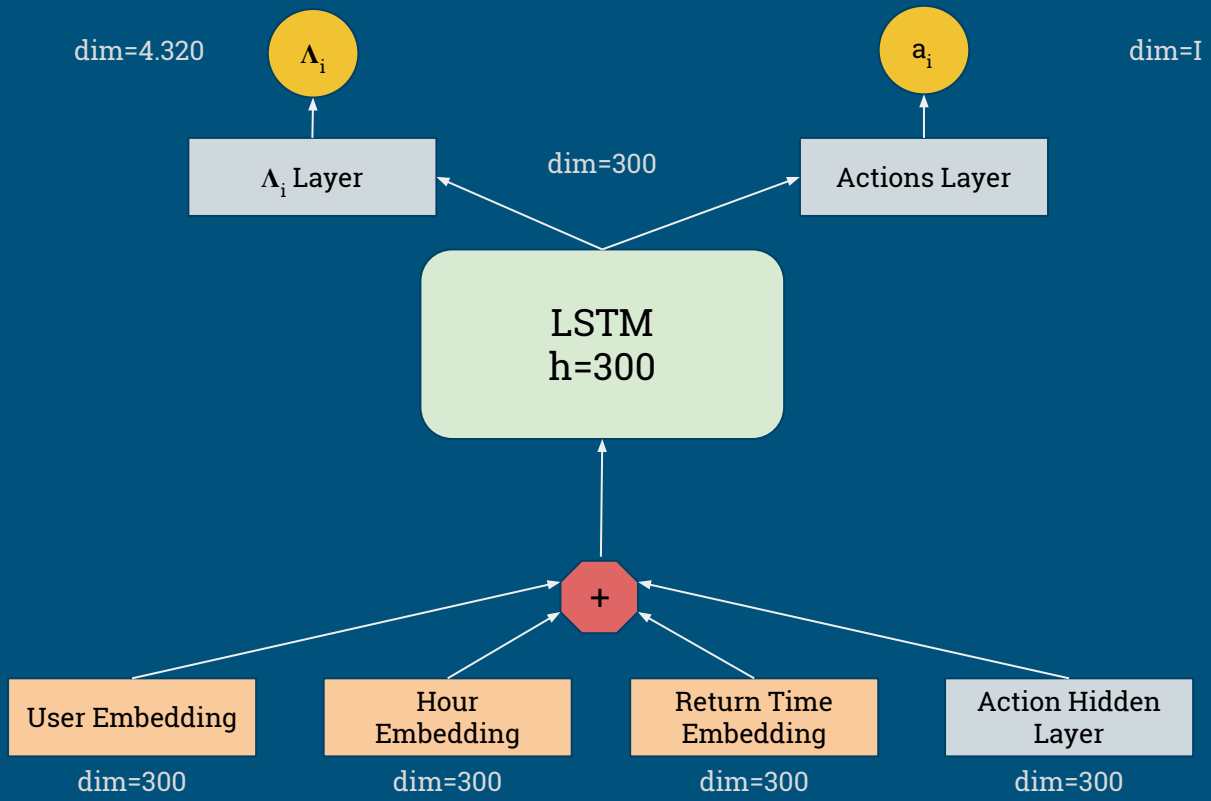


Fuente: Jing, How; Smola, Alexander J - Neural Survival Recommender. Proceedings of the Tenth ACM International Conference on Web Search and Data Mining

Solución: Implementación como LSTM

- Se trabaja con una LSTM, un subtipo de RNN
- Input es un vector concatenado con información de:
  - usuario→ Matriz **U** (U x 300)
  - hora y día de la semana de última sesión → Matriz **D** (168 x 300)
  - tiempo desde última sesión → Matriz **G** (4320 x 300)
  - acciones de última sesión
- Output de la red es Λ_i y el vector de acciones a_i

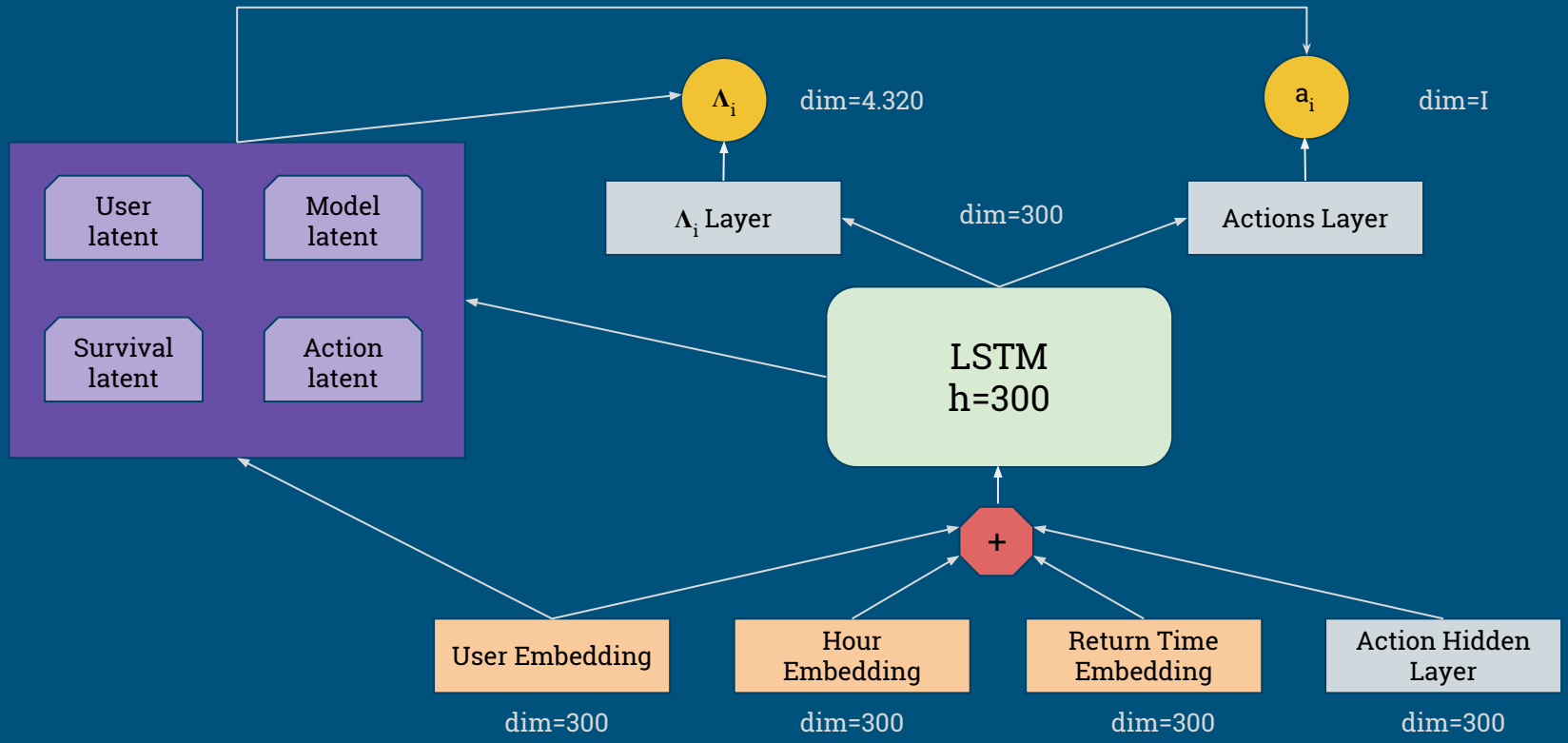
Solución: Arquitectura Inicial



Solución: Mejora Final - 3 Way Factorization

- Se aprenden matrices de factores latentes[9] para:
  - Embedding usuario
  - Respuesta LSTM (modelo)
  - Survival
  - Acciones
- Es similar a lo que hacemos en SVD
- En vez de una matriz de *ratings*, factorizamos:
  - Matriz de usuario-Λ_i
  - Matriz de usuario-acciones

Solución: Arquitectura Final



Experimentos: Setup

- Parámetros
  - 50 iteraciones
  - 300 neuronas de capa oculta, *embedding* y factorización.
  - μ=0,001
  - δt=1 hora
  - Horizonte de predicción: 180 días = 4.320 horas
- Sólo hacen split de *training* y *test*
- Funciones de pérdida ponderadas por α

Experimentos: Datasets

- *Lastfm*:
  - 632.304 sesiones de training, 97.935 sesiones de test
  - 1.000 usuarios y 20.000 ítems.
- *Tianchi-mobile*:
  - 289.399 sesiones de training, 122.119 sesiones de test
  - 10.000 usuarios y 8.917 ítems.
- Sesiones no explícitas:
  - Se decide un Δt máximo entre actividades para hacer el corte.

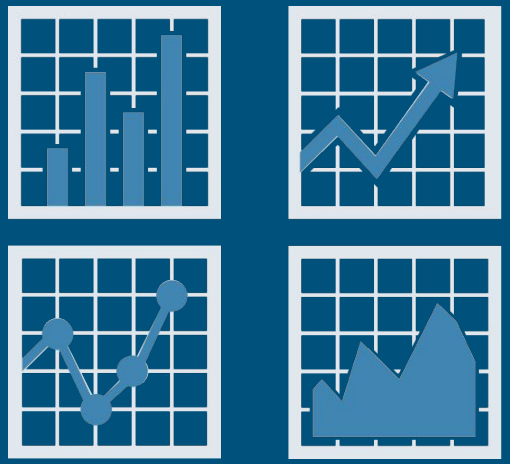
Experimentos: Baselines

- Hierarchical Poisson Factorization (HPF)
- Dynamic Poisson Factorization (DPF)
- Hierarchical Representation Model (HRM)
- Linear Regression with Huber Loss (LR-H)
  - Regresión Lineal sobre el tiempo de gap
- Proceso de Poisson (PP)
  - $\lambda(t)$ se asume constante.
- Proceso Hawkes (HP)

Sólo Ranking

Sólo Predicción Próxima Sesión

Experimentos: Métricas



- Calculamos:
  - MAE
  - Precision@K
  - Recall@K
  - MRR
  - MAR

Predicción

K = 15

Recomendación

- Predecimos el **tiempo de retorno** tomando el punto en que la *probabilidad de vivir y morir son iguales*, iguales a 0.5

Fuente imagen: https://www.ezshield.com/customer-communications-in-fraud-protection-ezshield/

Resultados: Comparación con Baselines

Precisión (MAE)

	tianchi-mobile	*lastfm*
Poisson Process	8.99	17.12
LR-Huber	8.37	17.12
Hawkes Process	8.26	15.64
Our Model	**7.53**	**14.66**

Resultados: comparación con Baselines

Recomendación

	Prec@15	*Recall@15*	*MRR*	*MAR*
HPF	0.092	0.290	0.110	366.01
DPF	0.0836	0.2369	0.0825	506.86
HRM	0.0925	0.3064	0.1261	418.83
Our Model	**0.107**	**0.365**	**0.167**	**356.02**
	Prec@15	*Recall@15*	*MRR*	*MAR*
HPF	0.0612	0.124	0.044	1608.88
DPF	0.0643	0.134	0.0562	1753.90
HRM	0.0840	0.2134	0.0923	1263.93
Our Model	**0.100**	**0.247**	**0.113**	**1077.01**

Conclusión y Puntos de Mejora

- Logran aprender conjuntamente un modelo que predice acciones y tiempo de retorno
- Survival Analysis parece una buena opción teórica en qué basarse
- Mejoran el estado del arte, pero no de manera revolucionaria
- Mejoras
  - No predice orden de acciones en la nueva sesión
  - No toma en cuenta efectos temporales absolutos. Ej: modas de películas, artistas, etc.
- Problemas:
  - Escalabilidad→ Netflix: 130MM users, Spotify: 83MM. Demasiados parámetros.
  - Cold Start
    - Usuario e ítem → Hay que reentrenar
  - No hubo análisis de ablación entre modelo sin factorización y con
  - No hay análisis de sensibilidad sobre δ y UT

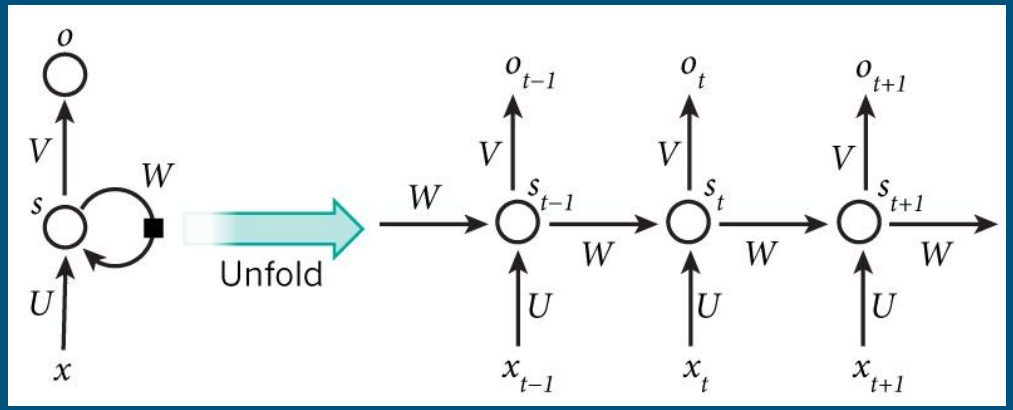
FIN

Referencias

[1] Y. Koren. Collaborative filtering with temporal dynamics. In knowledge discovery and data mining KDD, pages 447–456, 2009.
[2] L. Xiong, X. Chen, T.-K. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In SDM, pages 211–222. SIAM, 2010.
[3] Y. K. Tan, X. Xu, and Y. Liu. Improved recurrent neural networks for session-based recommendations. CoRR, 2016.
[4] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In Proceedings of the 19th International Conference on World Wide Web, 2010
[5] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng. Learning hierarchical representation model for nextbasket recommendation. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015
[6] K. Kapoor, M. Sun, J. Srivastava, and T. Ye. A hazard based approach to user return time prediction. In Knowledge discovery and data mining, pages 1719–1728. ACM, 2014.
[7] K. Kapoor, K. uian, J. Srivastava, and P. Schrater. Just in time recommendations: Modeling the dynamics of boredom in activity streams. In Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, 2015.
[8] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song. Recurrent marked temporal point processes: Embedding event history to vector. In Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, pages 1555–1564, New York, NY, USA, 2016. ACM.
[9] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. CoRR, 2014.

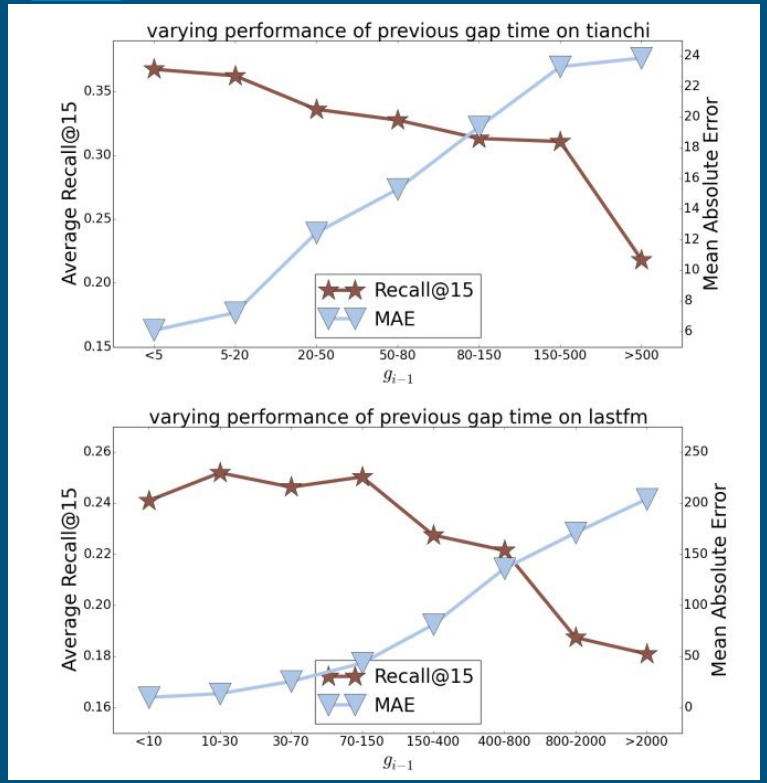
Solución: Redes Recurrentes

- Similares a una red neuronal feedforward
- La única diferencia es que también reciben como input el output de una computación de la misma red
- Especializadas en modelar relaciones secuenciales



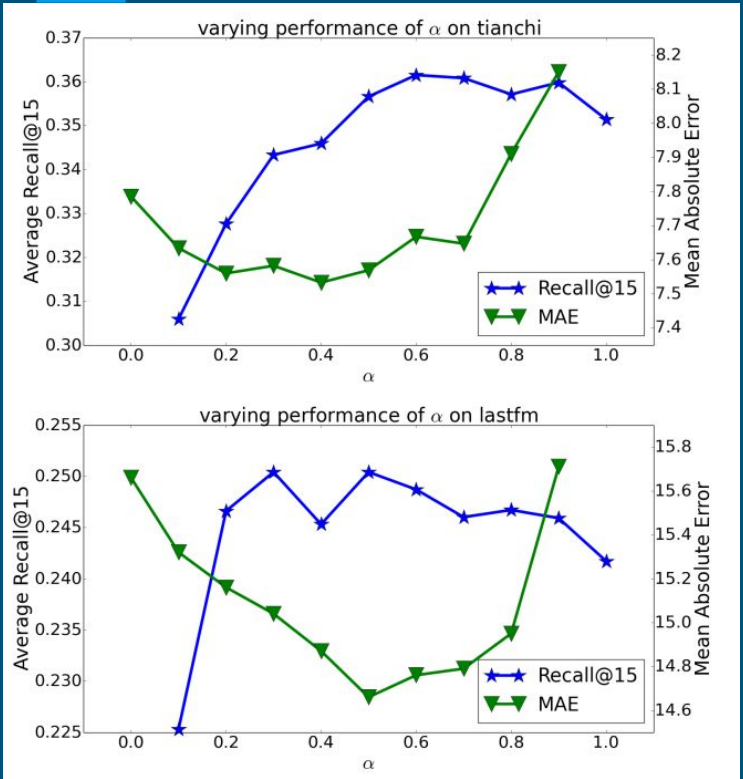
Fuente: http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Resultados: Tiempo desde última sesión



- Aumento tiempo entre sesiones
- Cae la precisión y el recall

Resultados: sensibilización sobre α



- Cambio de importancia relativa entre recomendación y predicción.
- alpha=0.5, buen tradeoff.

Solución: Ajustes para trabajar con LSTM (1/2)

- Ciertos inputs se cuantizan.
  - Tiempo entre sesión anterior y la actual (*gap time*)
  - Hora de la semana.
- Luego, todo input pasa por su propia capa de embedding de dimensión d=300
  - Usuario
  - Hora de la Semana
  - Gap time
  - Acciones Sesión Anterior*

Resultados: cantidad de sesiones

- Mayor cantidad de sesiones por usuario
- Mayor precisión y recall