Impacto de métodos de NLP sobre Sistemas Recomendadores

Doc2Vec, Sentiment Analysis y LDA[*]

Alain Raymond[**]
Pontificia Universidad Católica de Chile
Santiago, RM, Chile
afraymon@uc.cl

Rodrigo Rivera
Pontificia Universidad Católica de Chile
Santiago, RM, Chile
rgrivera@uc.cl

ABSTRACT

En este artículo se procede a explorar el impacto de un conjunto de métodos de procesamiento de lenguaje natural (NLP) sobre diversas métricas habituales de sistemas de recomendación. Para esto, se toma como base un modelo de Máquinas de Factorización[17] (FM) sobre el dataset de reseñas de restaurantes de Yelp![20] Se procede a modificar este modelo con features de cada uno de los métodos propuestos. Los resultados son interesantes: se descubre que la polaridad de las reseñas es el factor más crítico para mejorar tanto la Precisión y *Recall* de las predicciones. A su vez, se estudia la diversidad de nuestras recomendaciones utilizando la Diversidad Intralista. Dada la utilización del producto punto entre vectores latentes para generar las listas de recomendación, los valores de la similaridad se vuelven inherentemente cercanos a 1. Respecto a un set de Cold Start se propone un modelo de recomendación, sin embargo, los resultados no son buenos, incapaces de superar a un método Random.

CCS CONCEPTS

• **Information systems → Recommender systems**; *Sentiment analysis*; • **Computing methodologies → Natural language processing**;

KEYWORDS

RecSys, NLP, FM, Doc2Vec, LDA, Sentiment Analysis, AI

ACM Reference Format:
Alain Raymond and Rodrigo Rivera. 2018. Impacto de métodos de NLP sobre Sistemas Recomendadores: Doc2Vec, Sentiment Analysis y LDA. In *Proceedings of IIC3633 - RecSys (SANTIAGO'18),* Denis Parra, Antonio Ossa, and Manuel Cartagena (Eds.). ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

[*]Código disponible en el repositorio https://github.com/PUC-RecSys-Class/proyecto-raymond-rivera
[**]Ambos autores contribuyeron de igual forma

SANTIAGO'18, Diciembre 2018, Santiago, Chile

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1. INTRODUCCIÓN

Los Sistemas de Recomendación son poderosas herramientas para encontrar nuevos productos o servicios que consumir. Sin embargo, una de las informaciones más relevantes para determinar la calidad de un ítem consiste en las reseñas que otros usuarios han dejado. Éstas contienen información muy rica que no suele poder ser extrapolada de otros indicadores cuantitativos, como la cantidad de ratings, rating promedio o votos a favor. En general, esta información no es utilizada pues no es sencillo convertir esta información cualitativa en información que podamos usar en nuestros algoritmos automáticamente. Y si es utilizada, los usuarios consumen algunos pocos ejemplos, ignorando la mayoría de otras reseñas que podrían haberles sido útiles. En los últimos años varias técnicas se han desarrollado para atacar este problema. Algunas basadas en métodos probabilísticos y otras sustentadas en avances derivados de Aprendizaje Profundo. Se propone aplicar algunas de estas técnicas para observar su impacto sobre diversas métricas de recomendación, en particular, aquellas relacionados a la precisión y diversidad de las recomendaciones.

1.1. Estructura del Artículo

En la sección 2 se hará una revisión del Estado del Arte. En la sección 3 se efectuará una revisión del dataset utilizado. En la sección 4 se hablará de la metodología y detalles de la implementación. En la sección 5 se visualizarán resultados de cada uno de los modelos implementados. Finalmente, en la sección 6 se mostrarán las conclusiones y alternativas de trabajo futuro.

2. ESTADO DEL ARTE

Se procede a revisar los avances en tres materias:

- Métodos de interpretación de Lenguaje Natural
- Sistemas de Recomendación
- Métricas de Recomendación

2.1. Métodos de interpretación de Lenguaje Natural

Los primeros intentos de atacar el problema utilizaron métodos probabilísticos. El área de Filtrado Colaborativo basado en Contenido ofreció técnicas como TF-IDF, que miraban la frecuencia de palabras en texto y Latent Dirichlet Allocation (LDA)[4] que considera el texto como el resultado de una distribución probabilística y descubre tópicos bajo los cuales clasificar cada documento. Cada tópico está asociado a un conjunto de palabras, luego un documento puede estar asociado a más de un tópico.

Más adelante, otros propusieron tratar de codificar el valor sentimental de las palabras, tratando de obtener un vector que representara las cualidades del texto. Esto es el área de Análisis de Sentimientos[19]. En este campo se han propuesto algoritmos heurísticos como VADER[10] o algoritmos basados en implementaciones de Naive Bayes.

Doc2Vec[14] es una extensión natural del trabajo implementado en Word2Vec[16], un trabajo de aprendizaje profundo seminal, que logró generar codificaciones de las palabras que mantenían la semántica de ellas. A su vez, Doc2Vec logra lo mismo pero a nivel de documentos, pudiendo codificar textos de tamaño variable en un vector de tamaño fijo preservando la semántica general de éste.

Actualmente, de acuerdo a Barnes et al[1], los sistemas de mejor rendimiento son relacionados a Redes Neuronales Recurrentes (RNN). En particular, los modelos basados en LSTMs[8] y BiLSTMs[6].

2.2. Sistemas de Recomendación

Posterior al Netflix Prize[2], los métodos de factorización matricial empezaron a ganar popularidad con SVD para feedback explícito y ALS[9] para feedback implícito. Éstos métodos alcanzan su expresión definitiva en el trabajo de Rendl, con la introducción de las Máquinas de Factorización[17]. Éstas poseen una capacidad de expresión mayor que la factorización matricial estándar, pues pueden modelar relaciones no lineales de un orden arbitrario entre sus variables. Además tienen la capacidad de hacer esto de manera eficiente.

Sin embargo, actualmente -como en muchas otras áreas- estos métodos han sido dejados de lado a favor de métodos basados en Aprendizaje Profundo[7, 13]. Éstos poseen la capacidad de modelar teóricamente cualquier función, por lo que su poder expresivo es mayor que el de las Máquinas de Factorización. Ejemplos incluyen el Youtube Recommender[5], DeepCoNN[21] y el Neural Survival Recommender[11], que nos muestran aplicaciones en redes profundas estándar, convolucionales y recurrentes, respectivamente.

2.3. Métricas de Recomendación

Existen diversas propiedades[18] sobre las cuales evaluar el rendimiento de un sistema recomendador. En este trabajo nos enfocaremos en dos:

2.3.1. Precisión de la Predicción. La predicción de un rating es la categoría más inmediata, pues consiste en medir alguna desviación entre un rating predicho y uno real. Este fue el foco durante un gran tiempo de los sistemas recomendadores. Entre las métricas usuales se encuentran el Root Mean Square Error (RMSE) y el Mean Absolute Error (MAE).

Junto con la predicción de un rating, la evaluación de la calidad de una lista de recomendación también es pertinente. En este ámbito, se da importancia a la relevancia de la predicción. La precisión considerada como la cantidad de elementos relevantes recomendados sobre el total de elementos recomendados. Las métricas asociadas a esta dimensión son Precision@K, MAP@K, Recall@K, nDCG@K.

2.3.2. Diversidad. Las métricas asociadas a esta categoría tienen que ver con qué tan distintas entre ellas son las recomendaciones hechas por un sistema recomendador. En general se computan utilizando la similaridad entre ítems de la lista bajo alguna métrica de distancia y alguna función de ésta [18]. Una de las métricas más simples y utilizadas es la *Diversidad Intralista* (ILD). Se calcula con la siguiente fórmula:

$$diversity(R|u) = \frac{2}{|R|(|R-1|)} \sum_{k<n} d(i_n, i_k) \quad (1)$$

Donde R es la lista de recomendación evaluada y $d(i_n, i_k)$ es la distancia entre los elementos i_n e i_k.

3. DATASET

El dataset utilizado fue el de Yelp![20] Yelp! es un servicio de búsqueda local basado en reseñas generadas por la comunidad de usuarios del servicio. Éste permite buscar elementos en categorías tan variadas como spas, peluquerias, cines y restaurantes. El dataset está acotado a información de restaurantes en particular, repartidos en 9 ciudades de Estados Unidos. Cuenta con alrededor de 6 millones de reseñas y el nivel de detalle es rico tanto para la información de los restaurantes, usuarios y reseñas. En particular, estas últimas se encuentran escritas en lenguaje natural junto con una calificación numérica.

Para acotar el problema - y facilitar la ejecución de los experimentos- se decidió mantener sólo los datos de la ciudad de Phoenix, la segunda ciudad con más reviews. Dentro de esta ciudad se redujo aún más el universo de reseñas tomando dos grupos de datos:

- **Hot Start:** todos los usuarios y restaurantes con más de 15 reseñas.
- **Cold Start:** usuarios no vistos con 2 reviews y restaurantes no vistos con 2 reviews.

Cuadro 2: Datos Generales Dataset

Métrica	Hot Start	Cold Start	Total
Cantidad de Tuplas	117.346	60.312	177.658
Usuarios Únicos	3.498	30.156	33.654
Restaurantes Únicos	2.191	6.761	8.952
Sparsity	1,53 %	0,029 %	0,059 %
Rating Promedio	3,86	3,81	-
Varianza Rating	1,31	2,41	-

4. METODOLOGÍA

Se implementó un modelo Base de FM. Éste utilizó las siguientes variables:

- UserId
- RestaurantId
- Rating en escala de 1 a 5
- Vector de Categorías ("Mexicana", 'India', "Pizzas", etc.). Representado como BoW.
- Cantidad de usuarios que encontró útil la reseña.
- Cantidad de fans del usuario que posteó la reseña.
- Tiempo desde que se escribió la reseña.
- Estrellas promedio del usuario

Para complementar el modelo Base, también se probaron modelos *Random* y *MostPopular* para determinar efectivamente que los modelos entrenados capturaran patrones valiosos de los datos y

	Hot Start					Cold Start				
Modelo	RMSE	MAE	Prec@10	Recall@10	ILD	RMSE	MAE	Prec@10	Recall@10	ILD
Base	1,2298	0,9538	0,0777	0,0829	0,9610	3,3863	2,8169	0,0002	0,0009	0,9670
Random	2,1350	1,7650	0,0107	0,0108	-	2,3224	1,9728	**0,0006**	**0,0030**	-
Most Popular	-	-	0,0047	0,0048	-	-	-	0,0004	0,0018	-
LDA - 5	1,1179	0,8573	0,0709	0,0745	0,9546	2,6886	2,0603	0,0002	0,0009	0,9776
LDA - 10	1,0922	0,8414	0,0739	0,0775	0,9662	3,1790	2,5922	0,0003	0,0014	0,9621
LDA - 20	1,0661	0,8179	0,0782	0,0820	0,9622	3,0070	2,4201	0,0003	0,0013	**0,9388**
LDA - 30	1,0575	0,8080	0,0746	0,0781	0,9636	2,6313	1,9887	0,0001	0,0007	0,9758
Doc2Vec - 50	**0,9474**	**0,7002**	0,0603	0,0629	0,9601	**2,2770**	**1,5233**	0,0003	0,0013	0,9755
Doc2Vec - 100	1,4772	1,0967	0,0067	0,0069	0,9603	2,3892	1,7016	0,0002	0,0010	0,9720
Doc2Vec - 200	1,6979	1,2689	0,0056	0,0057	0,9638	2,8830	2,2656	0,0002	0,0008	0,9756
Doc2Vec - 300	2,0402	1,5350	0,0074	0,0075	**0,9517**	3,0045	2,3235	0,0002	0,0008	0,9672
Sentiment - Polarity	1,0678	0,8222	**0,1052**	**0,1101**	0,9614	2,5772	1,9425	0,0002	0,0010	0,9669
Sentiment - Subjectivity	1,2241	0,9430	0,0600	0,0645	0,9628	2,5700	1,9129	0,0004	0,0018	0,9606
Sentiment-Vader	1,0599	0,8118	0,0622	0,0656	0,9591	2,9450	2,3078	0,0005	0,0027	0,9442

Cuadro 1: Resultados para todos los modelos - Grupos Hot Start y Cold Start

pudiesen generar mejores predicciones que estos modelos básicos. A continuación, se entrenaron modelos basados en el modelo Base aumentados con features derivados de los siguientes métodos de interpretación de Lenguaje Natural:

- LDA
- Doc2Vec
- Sentiment Analysis

4.1. Generación de Listas de Recomendación

Para el grupo Hot Start se utilizó el producto punto de los vectores latentes de usuario con los de restaurantes, lo que es el método usual para Máquinas de Factorización. Para el grupo de Cold Start, dada la imposibilidad de tener estos vectores se utiliza un método alternativo:

1. Para cada restaurant se calcula el vector promedio entre los vectores latentes asociados a sus categorías:
$$R_i = \frac{\sum_{i \in c} C_i}{n} \quad (2)$$
2. Para cada usuario se genera un vector basado en los vectores latentes de cantidad de estrellas promedio por usuario y fans, multiplicados por los features asociados:
$$U_i = (S * V_S + F * V_F) \quad (3)$$
3. Finalmente la recomendación es el TOP N del producto punto entre el vector usuario y el vector de cada restaurant:
$$Recommendation(U_i) = TopN(U_i \cdot R_i) \quad (4)$$

4.2. Detalles de implementación

4.2.1. Modelos. Todos los modelos basados en FMs fueron implementados utilizando TFFM[15]. En estos casos se utilizó ADAM[12] como optimizador, se ajustaron en 50 las dimensiones de factorización y sólo se entrenaron interacciones de orden 2. El número de épocas fue de 500, el tamaño de Batch de 2.048 y la tasa de aprendizaje se dejó en 0,01. Los modelos *Random* fueron corridos 50 veces sobre los datos. Para la predicción de ratings se utilizó la función `randint` de la librería `random` de Python. Ésta predice enteros entre 1 y 5. Para la recomendación de restaurantes se eligió entre el total de restaurantes de manera uniforme sin reemplazo.

4.2.2. Preprocesamiento de los datos. Todas las reseñas fueron preprocesadas para ser tokenizadas, se eliminaron *stopwords*, se hizo stemming y lemmatization. No hubo ningún corte de palabras infrecuentes. Todo este trabajo se hizo utilizando NLTK[3]. Se eligieron las categorías de restaurantes con más de 100 apariciones entre los restaurantes del dataset.

4.2.3. Datos de Entrenamiento y Test. Se efectuó una separación 80-20 entre los sets de entrenamiento y test. Esta separación se hizo por antigüedad de la reseña por cada usuario. Luego cada set fue dividido en 5 y se obtuvieron K=3 permutaciones de éstos para generar cada fold. Todos los modelos fueron entrenados utilizando K-Folds con K=3. El set de entrenamiento sólo utilizó información del grupo de datos Hot Start. Se generaron archivos de test distintos para el grupo Hot Start y Cold Start.

4.3. Métricas de Rendimiento

Todos los modelos fueron evaluados respecto a las siguientes métricas: RMSE, MAE, Precision@10, Recall@10 e Intralist Diversity. La métrica de distancia utilizada en ILD fue la distancia coseno entre los vectores latentes de restaurantes. Para el grupo Hot Start se determinaron como relevantes aquellos restaurantes asociados a reseñas de 3 puntos o más. Para el grupo Cold Start no se hizo ningún corte, pues sólo había dos elementos relevantes por usuario.

5. ANÁLISIS DE PARÁMETROS Y RESULTADOS

5.1. Análisis de Parámetros

Para cada uno de los métodos de interpretación de Lenguaje Natural se entrenaron modelos variando las dimensiones o tópicos asociados al método. Para LDA se entrenaron modelos de 5,10,20 y

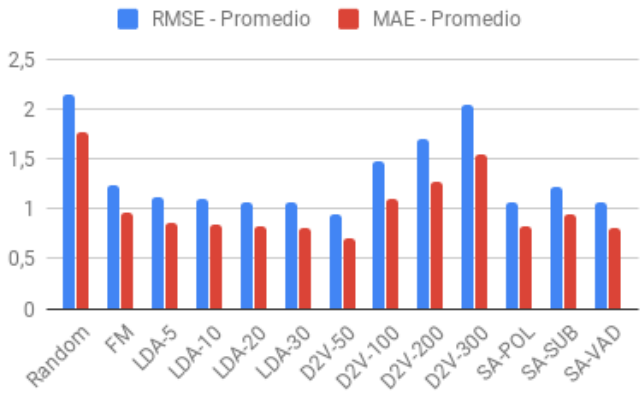

Figura 1: Hot Start: RMSE y MAE para todos los modelos

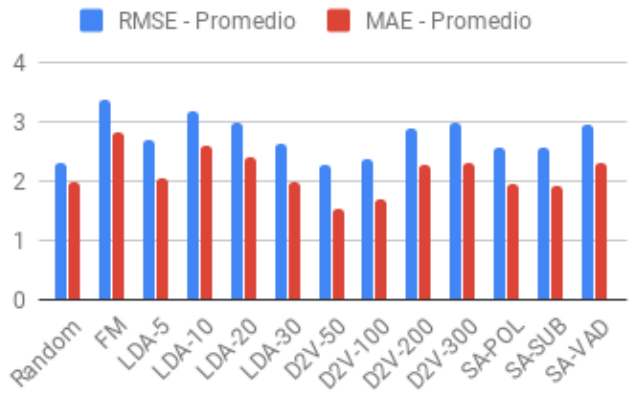

Figura 2: Cold Start: RMSE y MAE para todos los modelos

30 tópicos. Para Doc2Vec se entrenaron modelos basados en codificaciones de 50, 100, 200 y 300 dimensiones. Para Sentiment Analysis se probaron dos métodos distintos: VADER y TextBlob. El primero ofrece como salida cuatro resultados: porcentaje de palabras positivas, negativas y neutrales, junto con un indicador compuesto derivado de los porcentajes y la intensidad de cada palabra. Por su lado, el algoritmo de TextBlob ofrece solo dos indicadores: Polaridad, que indica qué tan positivo o negativo es el comentario; Subjetividad, que indica qué tan objetivo o subjetivo es el comentario. Para los modelos de Máquinas de Factorización, basados en el paper de Rendl[17], no se utilizaron órdenes de factorización mayores a 2 pues los resultados serían marginalmente superiores, y los tiempos de entrenamiento se disparan para órdenes mayores.

Todos los modelos fueron evaluados en las siguientes métricas: RMSE, MAE, Prec@10, Recall@10 e ILD.

5.2. Predicción de Ratings

A nivel de reducción del error de predicción de rating, casi todos los modelos se comportan de manera razonable y reducen el error respecto al modelo Base. Sin embargo, el caso más notable es Doc2Vec que desde las 100 dimensiones en adelante parece hacer *overfitting* cada vez más severo en los datos. Por su lado, métodos más sencillos como LDA funcionan mejor en general a medida que aumentamos los tópicos y Sentiment Analysis parece funcionar mejor en general. En el caso del modelo sólo basado en Subjetividad, tiene sentido que no se vea una mejora pues no se esperaría que esta métrica por sí sola genere mejores predicciones.

5.3. Precision y Recall

Respecto a la *Precisión* y el *Recall*, en general se observa una tendencia de los modelos a empeorar respecto al modelo Base. Existen varias posibilidades para este resultado, entre ellas falta de datos para Doc2Vec o la no necesaria correlación entre mejores predicciones de rating vs mejores listas de recomendación. La única excepción a esta tendencia a generar peores listas de recomendación, es el modelo aumentado con la Polaridad de los comentarios. Este modelo presenta en promedio una mejora del 35 % y 32 % en Precision y

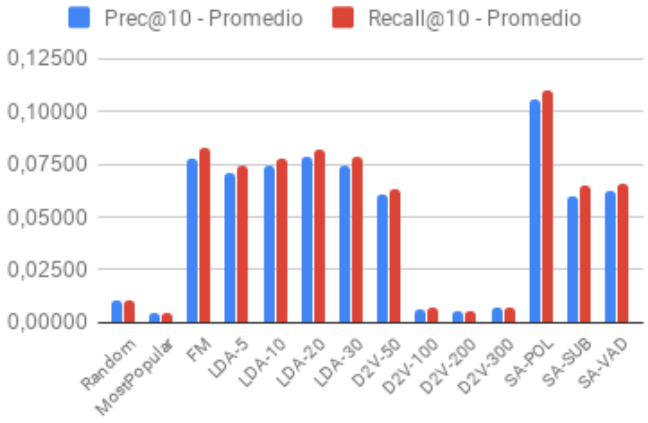

Figura 3: Hot Start: Precision@10 y Recall@10 para todos los modelos

Recall respectivamente respecto al modelo Base. Esto es muy interesante pues nos indica que un feature de una sola dimensión tiene un impacto clave versus otros features en más dimensiones. Además, Sentiment Analysis tiene menores requisitos de preprocesamiento que métodos como LDA y Doc2Vec, haciéndolo aún más práctico.

5.4. Diversidad

Respecto a la Diversidad, se observa que los diversos modelos ofrecen listas de recomendación bastante uniformes con Similaridades Intralista cercanas al 95 %. El único modelo que se desvía del resto es el asociado a Doc2vec con 300 dimensiones. Sin embargo, este aumento en la Diversidad es más probable que sea derivado del mal poder predictivo del modelo que de una tendencia natural de éste a dar recomendaciones diversas. Los porcentajes de similaridad tan altos entre listas se estipula que tienen que ver con el método de generación de éstas. Pues al utilizar los resultados más altos del producto de los vectores latentes del usuario con los restaurantes, es esperable que el valor del producto punto entre los candidatos elegidos sea también alto.

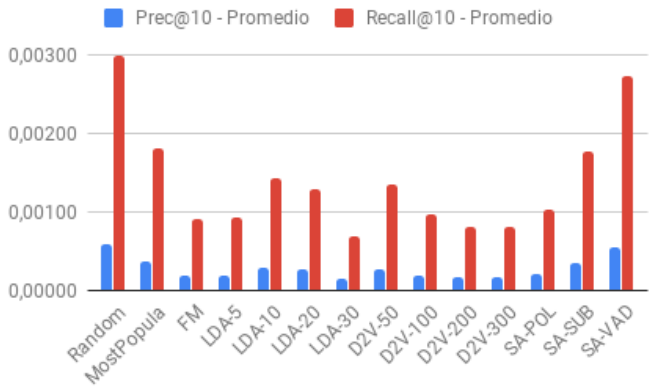

Figura 4: Cold Start: Precision@10 y Recall@10 para todos los modelos

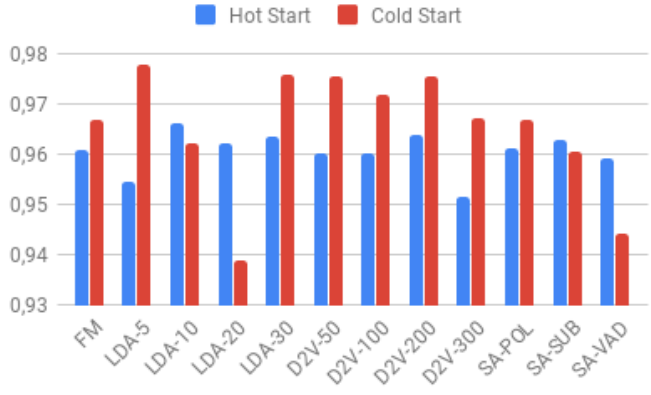

Figura 5: Intralist Diversity (ILD): Hot Start y Cold Start para todos los modelos

Para alterar la diversidad de las listas sería necesario generar modelos de generación de listas de recomendación distintos a aquellos generados basados simplemente en el producto punto de los vectores latentes.

6. CONCLUSIONES

6.1. General

Se analizó el impacto de diversos métodos de interpretación de lenguaje natural. Vemos que el uso de estos features mejora el rendimiento de las recomendaciones en distintas métricas.

6.2. Grupo Hot Start

En términos de predicción de ratings, casi todos generaron reducciones en el error, como sería lo esperable. Sólo Doc2Vec aumentó el error de predicción, probablemente por hacer *overfitting* o por no tener suficientes datos para generar una codificación aceptable. Se sugirió también que la implementación de Doc2Vec de Gensim es subóptima.

Se pudo observar que la variable más interesante derivada del lenguaje natural a la hora de generar mejores listas de recomendación es la **Polaridad** del comentario, siendo capaz de generar mejores resultados que modelos con decenas o cientos de dimensiones semánticas calculadas.

6.3. Grupo Cold Start

El rendimiento de los modelos en el conjunto de datos de Cold Start es muy pobre, lo que es esperable pues el modelo de Máquinas de Factorización está pensados para trabajar con usuarios e ítems conocidos. Además el dataset de Cold Start es particularmente difícil, haciendo la tarea aún más complicada.

El método de recomendación generado para este grupo de datos genera recomendaciones que ni siquiera logran superar a recomendaciones *Random*. En vista de esto, se recomienda o utilizar recomendaciones aleatorias, o utilizar algún método híbrido de Content Filtering para ayudar a resolver estos casos.

6.4. Diversidad

Respecto a la Diversidad, se concluye que el método de generación de listas de recomendación basado en el producto punto de vectores latentes generará listas inherentemente similares. Luego, si se desea generar listas con mayor diversidad, es necesario ocupar un método alternativo de recomendación.

6.5. Trabajo Futuro

Se propone entrenar los modelos con mayores cantidades de datos, para descartar la falta de datos como el problema de rendimiento de Doc2Vec, junto con probar otras implementaciones de éste. Ademas, se sugiere hacer pruebas en más datasets para verificar que el resultado obtenido (en particular, el resultado asociado a la polaridad) no esté acotado al mundo de los restaurantes sino que sea extrapolable a otros dominios.

REFERENCIAS

[1] Jeremy Barnes, Roman Klinger, and Sabine Schulte im Walde. 2017. Assessing State-of-the-Art Sentiment Models on State-of-the-Art Sentiment Datasets. *CoRR* abs/1709.04219 (2017). arXiv:1709.04219 http://arxiv.org/abs/1709.04219

[2] J. Bennett and S. Lanning. 2007. The Netflix Prize. In *Proceedings of the KDD Cup Workshop 2007*. ACM, New York, 3–6. http://www.cs.uic.edu/~liub/KDD-cup-2007/NetflixPrize-description.pdf

[3] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python* (1st ed.). O'Reilly Media, Inc.

[4] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* 3 (March 2003), 993–1022. http://dl.acm.org/citation.cfm?id=944919.944937

[5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. ACM, New York, NY, USA, 191–198. https://doi.org/10.1145/2959100.2959190

[6] Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *NEURAL NETWORKS* (2005), 5–6.

[7] Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. 2006. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation* 18, 7 (2006), 1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[9] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM '08)*. IEEE Computer Society, Washington, DC, USA, 263–272. https://doi.org/10.1109/ICDM.2008.22

[10] Clayton J. Hutto and Eric Gilbert. 2014. VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text.. In *ICWSM*, Eytan Adar, Paul Resnick, Munmun De Choudhury, Bernie Hogan, and Alice H. Oh (Eds.). The AAAI Press. http://dblp.uni-trier.de/db/conf/icwsm/icwsm2014.html#HuttoG14

[11] How Jing and Alexander J. Smola. 2017. Neural Survival Recommender. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM '17)*. ACM, New York, NY, USA, 515–524. https://doi.org/10.1145/3018661.3018719

[12] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). arXiv:1412.6980 http://arxiv.org/abs/1412.6980

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12)*. Curran Associates Inc., USA, 1097–1105. http://dl.acm.org/citation.cfm?id=2999134.2999257

[14] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. *CoRR* abs/1405.4053 (2014). arXiv:1405.4053 http://arxiv.org/abs/1405.4053

[15] Alexander Novikov Mikhail Trofimov. 2016. tffm: TensorFlow implementation of an arbitrary order Factorization Machine. https://github.com/geffy/tffm.

[16] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., USA, 3111–3119. http://dl.acm.org/citation.cfm?id=2999792.2999959

[17] Steffen Rendle. 2010. Factorization Machines. In *Proceedings of the 2010 IEEE International Conference on Data Mining (ICDM '10)*. IEEE Computer Society, Washington, DC, USA, 995–1000. https://doi.org/10.1109/ICDM.2010.127

[18] Guy Shani and Asela Gunawardana. 2011. Evaluating Recommendation Systems. In *Recommender Systems Handbook*, Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer, 257–297. http://dblp.uni-trier.de/db/reference/rsh/rsh2011.html#ShaniG11

[19] Peter D. Turney. 2002. Thumbs Up or Thumbs Down?: Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL '02)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 417–424. https://doi.org/10.3115/1073083.1073153

[20] Yelp! 2018. Yelp Dataset Challenge. https://www.yelp.com/dataset/challenge

[21] Lei Zheng, Vahid Noroozi, and Philip S. Yu. 2017. Joint Deep Modeling of Users and Items Using Reviews for Recommendation. *CoRR* abs/1701.04783 (2017). arXiv:1701.04783 http://arxiv.org/abs/1701.04783