ANALISIS DAN IMPLEMENTASI *CLUSTER-SMOOTHED* PADA *COLLABORATIVE FILTERING*

ANALYSIS AND IMPLEMENTATION OF *CLUSTER-SMOOTHED* FOR *COLLABORATIVE FILTERING*

Aulia Rahmawati[1], Agung Toto Wibowo, ST., MT.[2], Gia Septiana W., S.Si., M.Sc. [3]

[1,2,3]Prodi S1 Teknik Informatika, Fakultas Teknik, Universitas Telkom

[1]**auliasuwito@gmail.com**, [2]**atwbox@gmail.com**, [3]**giaseptiana@gmail.com**

Abstrak

***Collaborative filtering* adalah salah satu pendekatan yang biasa digunakan pada sistem rekomendasi untuk mendapatkan nilai prediksi dari item oleh user aktif. Dalam *collaborative filtering* terdapat beberapa masalah yang sering muncul salah satunya adalah *data sparsity*. *Data sparsity* adalah permasalahan keterbatasan user untuk memberikan rating terhadap keseluruhan item. Untuk memecahkan masalah tersebut terdapat beberapa model diantaranya *memory-based*, *model-based* dan *hybrid model*. Tiap-tiap model memiliki kelebihan dan kekurangan masing-masing. Oleh karena itu, dalam tugas akhir ini akan digunakan model gabungan yaitu akan menggabungkan *clustering* dan *smoothing* untuk menangani masalah *data sparsity* dalam prediksi rating user. Dari hasil pengujian didapatkan nilai Mean Absolute Error (MAE) terkecil sebesar 0,732.**

Kata kunci : ***collaborative filtering, sparsity, clustering, smoothing***

Abstract

Collaborative filtering is one of the commonly approach in recommender system to obtain the predicted value of item by active user. In collaborative filtering there are several problem that often arise, one of them is data sparsity. Data sparsity; is the problem of limited user to rate overall item. To overcome these problem, there are several models including memory-based, model-based and hybrid model. Each model has the advantages and disadvantages. Therefore in this final project used the combined models that will combine clustering and smoothing to handling the problem of data sparsity in the prediction of user ratings. The result show Mean Absolute Error (MAE) is about 0,732.

Keywords: ***collaborative filtering, sparsity, clustering, smoothing***

1. Pendahuluan

Collaborative Filtering memiliki peranan penting dalam membangun sebuah sistem rekomendasi karena pendekatan *collaborative filtering* memiliki tujuan untuk mendapatkan prediksi rating dari user aktif terhadap item-item dalam sistem berdasarkan user-user lain yang dianggap mirip. Untuk mendapatkan prediksi dari user aktif diperlukan pengolahan terhadap data rating user dimana keterbatasan user dalam memberikan rating item (*data sparsity*) juga memberikan pengaruh pada data dan sistem prediksi [1].

Beberapa pendekatan yang dapat digunakan dalam *collaborative filtering* diantaranya adalah *memory-based*, *model-based* dan *hybrid model*. Pendekatan *memory-based* melakukan komputasi pada data matriks rating user untuk mendapatkan beberapa user yang paling mirip dengan user aktif dalam memprediksi rating user aktif. Beberapa metode *memory-based* diantaranya adalah *Pearson-Correlation* [2], *The Vector Similarity* [3] dan *Generalized Vector Space Model* [4]. Kelebihan pendekatan ini adalah sangat mudah untuk diimplementasikan dan kemudahan dalam penambahan data baru. Namun performansi pada model ini akan menurun seiring bertambahnya data. Sedangkan pendekatan *model-based* melakukan pemodelan terhadap data yaitu dengan membuat kelompok-kelompok user berdasarkan kemiripan ratingnya. Untuk memprediksi rating dari user aktif *model-based* memasukkan user aktif kedalam kelompok tertentu dan menggunakan data kelompok tersebut dalam melakukan prediksi seperti pada *Bayesian Network* [3], *Clustering* [5, 6] dan *Aspect Model* [7]. *Model-based* ini cukup baik dalam menangani data yang bertambah besar. Meskipun lebih baik dalam menangani hal tersebut, *model-based* membutuhkan waktu cukup lama untuk membangun model.

Hybrid model menggabungkan kelebihan-kelebihan yang ada pada *memory-based* dan *model-based* untuk mendapatkan prediksi rating user aktif [8]. Penggabungan model ini seperti yang dilakukan oleh metode *Cluster-Smoothed*. Metode ini menggunakan *clustering* untuk mengelompokkan user, sehingga pada proses prediksi tidak diperlukan komputasi pada keseluruhan user melainkan hanya fokus pada sekelompok user saja. Selain itu

penggunaan *smoothing* juga dilakukan untuk mengatasi user yang tidak memberikan rating pada item. Dengan menggunakan *clustering* ketika menentukan nilai *smoothing* menunjukan adanya integrasi antara kedua model [9].

Dalam penelitian ini digunakan penggabungan model *memory-based* dan *model-based*. Dimana pendekatan *clustering* dan *smoothing* ini cukup baik untuk menangani masalah *data-sparsity*. Data yang digunakan untuk studi kasus ini adalah data yang memiliki data id user, id item dan nilai rating misalnya data movie rating.

2. Landasan Teori

2.1 Collaborative Filtering

Collaborative filtering adalah salah satu metode yang biasa digunakan pada sistem rekomendasi untuk mendapatkan nilai prediksi dari item oleh user aktif. Misalkan saja diberikan data item berupa list film [illegible] *filtering* ini dapat [illegible] rating dari user [illegible] terhadap item [illegible]. Dengan *collaborative*

Tabel 2-1 Contoh Data pada Collaborative Filtering

	X-men	Spiderman	The Conjuring	Annabelle	Insidious
Aulia	5	4	?	4	5
Melisa	5	4	4	3	?
Putri	?	3	2	?	?
Galuh	4	?	4	3	4
Dwi	3	3	5	5	1

Pada kenyataannya user tidak memberikan rating terhadap keseluruhan item. Kebanyakan user akan memberikan rating terhadap item-item tertentu yang disukainya, masalah ini untuk selanjutnya disebut sebagai *data sparsity*. Terkadang beberapa user juga melakukan kecurangan terhadap sistem, seperti memberikan rating tinggi pada item yang menjadi produknya dan memberikan rating rendah pada item yang menjadi kompetitornya. Hal seperti ini dalam collaborative filtering disebut sebagai *shilling attacks*, yaitu serangan user palsu terhadap sistem rekomendasi. Seperti yang dilakukan user [illegible] pada tabel 2-1 menganggap [illegible] adalah saingan dari [illegible] dan [illegible] Adanya user palsu ini tentu saja akan merusak data yang nantinya akan mempengaruhi proses prediksi. Masalah lain dalam *collaborative filtering* adalah *scallability*, yaitu data user dan item yang sangat besar dan masih akan terus berkembang memberikan tantangan dalam hal komputasi data untuk memperoleh prediksi rating user aktif.

Beberapa pendekatan yang dapat digunakan pada *collaborative filltering* diantaranya adalah *memory-based*, *model-based* dan gabungan antara keduanya disebut *hybrid model* [1].

2.2 Memory-based

Pendekatan *memory-based* memanfaatkan database user-item untuk menentukan user atau item yang mirip yang nantinya digunakan untuk proses prediksi. Pertama-tama pendekatan ini melakukan perhitungan kemiripan atau bobot dari dua user atau item [illegible]dan [illegible] kemudian nilai prediksi didapat dari perhitungan bobot rata-rata user terhadap suatu item.

Cara lain yang dapat digunakan adalah dengan menentukan rekomendasi top-*N* item, yaitu mencari *K*-user yang mirip dengan menghitung nilai kemiripannya, kemudian menggabungkan item yang didapat menjadi top-*N* item sebagai rekomendasi [1]. Beberapa perhitungan kemiripan yang dapat digunakan diantaranya :

1. *Pearson Correlation*

Perhitungan pearson ini digunakan untuk melihat relasi linier terhadap dua variabel. *Pearson correlation* antara user [illegible]dan [illegible]adalah [1]

$$[illegible] = \frac{\sum_{[illegible]}([illegible] - \bar{[illegible]})([illegible] - \bar{[illegible]})}{\sqrt{\sum_{[illegible]}([illegible] - \bar{[illegible]})^2}\sqrt{\sum_{[illegible]}([illegible] - \bar{[illegible]})^2}} \quad (2\text{-}1)$$

Dimana,

[illegible] adalah nilai rating user [illegible]terhadap item [illegible]

$\bar{[illegible]}$adalah nilai rata-rata rating user [illegible]

[illegible] adalah nilai rating user [illegible]terhadap item [illegible]

$\bar{[illegible]}$adalah nilai rata-rata rating user [illegible]

[illegible]adalah himpunan item [illegible]yang diberi rating oleh user [illegible]

adalah himpunan item yang diberi rating oleh user

2. *Vector Cosine Similarity*

Pada cosine similarity data rating item dari user dianggap sebagai sebuah vektor. Sehingga untuk antara kedua vektor. Vector cosine similarity antara item dan adalah[11] menghitung sudut yang terbentuk

$$= \cos(\vec{}, \vec{}) = \frac{\vec{} \odot \vec{}}{\|\vec{}\| \otimes \|\vec{}\|} \quad (2\text{-}2)$$

sebagai contoh, jika vektor A={ , } dan vektor B={ , }, maka

$$= \cos(\vec{}, \vec{}) = \frac{\vec{} \odot \vec{}}{\|\vec{}\| \otimes \|\vec{}\|} = \frac{+}{\sqrt{^2+^2}\sqrt{^2+}} \quad (2\text{-}3)$$

2.3 Model-based

Pendekatan *model-based* membuat sistem dapat melakukan pembelajaran dalam memodelkan data kedalam kelompok-kelompok tertentu berdasarkan pola dari data, kemudian melakukan prediksi terhadap data testing berdasarkan model yang telah dipelajari. Beberapa metode yang menggunakan *model-based* adalah *Bayesian models* [3], *Clustering models* [5, 6] dan *Latent Semantic models* [7].

2.4 Hybrid model

Hybrid model menggabungkan keuntungan dua pendekatan sebelumnya yaitu *memory-based* dan *model-based*. Salah satu metode yang menggunakan pendekatan *hybrid* adalah metode *Cluster-Smoothed*. Metode ini menggunakan *clustering* untuk menentukan kelompok-kelompok dan melakukan smoothing terhadap data tiap user yang tidak memberikan rating. Dengan menggunakan data user yang memiliki kemiripan ditiap kelompok maka item yang tidak diberi rating pun dapat diperkirakan nilai ratingnya. Dan juga top-*N* item bisa didapat dari top-*N* cluster [9].

Tahapan-tahapan algoritma *cluster-smoothed* adalah sebagai berikut :

1. Menentukan kelompok-kelompok user (*clustering*).
2. Jika diberikan user aktif ( ) maka selanjutnya cari *cluster* yang memiliki karakteristik mirip dengan (*neighbor pre-selection*).
3. Dari *cluster* ambil sebanyak top-*K* user yang mirip dengan (*neighbor selection*).

Prediksi rating dari didapat dengan menghitung deviasi bobot rata-rata dari subset top-*K* user yang mirip dengan

2.4.1 Clustering (K-means)

Clustering berfungsi untuk mengelompokkan data kedalam beberapa kelompok (*cluster*) sesuai keinginan kita. Dalam mengelompokkan data diperlukan pengetahuan dari data yang akan dinyatakan dalam satu kelompok. Pada *clustering* pengetahuan tersebut bisa didapat menggunakan metode salah satunya adalah *Euclidean Distance*. Metode ini memberikan informasi berupa jarak antara *cluster* terhadap user yang akan kita kelompokkan. K-means akan memilih *cluster* dengan jarak terdekat. Jarak pada *Euclidean Distance* mewakili similaritas antara user dan *cluster*. Rumus untuk menentukan jarak pada *Euclidean Distance* adalah sebagai berikut [5, 6, 10].

$$= \frac{\sum_{\in \cap} (( ) - \bar{}) \cdot (( ) - \bar{})}{\sqrt{\sum_{\in \cap} (( ) - \bar{})^2} \sqrt{\sum_{\in \cap} (( ) - \bar{})^2}} \quad (2\text{-}4)$$

Dengan,

( ) adalah nilai rating user terhadap item
( ) adalah nilai rating user terhadap item
adalah item yang telah diberi rating oleh user
adalah item yang telah diberi rating oleh user

Pada *cluster-smoothed clustering* berfungsi untuk mengelompokkan user. Dimana nantinya prediksi rating akan dihitung berdasarkan *cluster* yang mirip dengan user aktif [9].

Algoritma K-means
Input : Data, jumlah centroid
Output : Data tercluster
Algoritma :
Set jumlah centroid yang diinginkan.
Inisialisasi centroid secara acak.
While dataIsStillMoving
For all centroid
Hitung jarak data ke-i & centroid ke-j.
Set data ke-i ke centroid terdekat.
Untuk semua centroid, hitung mean centroid.
EndWhile
End

Gambar 2-1 Pseudocode Algoritma K-means [11]

2.4.2 Data Smoothing

Data smoothing dilakukan untuk mengisi data rating user yang kosong, yaitu user yang tidak memberikan rating pada item. Untuk mengisi nilai tersebut dapat dilakukan dengan cara berikut [9].

$$P_{u,i} = \begin{cases} R_{u,i} & \text{jika } R_{u,i} \text{ tidak kosong} \\ \bar{R}_u + \Delta R_u & \text{jika } R_{u,i} \text{ kosong} \end{cases} \quad (2\text{-}5)$$

$P_{u,i}$ adalah nilai rating smoothing dari user u terhadap item i Nilai $P_{u,i}$ didapat dari persamaan (2-6).

$$P_{u,i} = \bar{R}_u + \Delta R_u \quad (2\text{-}6)$$

Dengan nilai ΔR_u adalah :

$$\Delta R_u = \frac{\sum_{i' \in I_u} (R_{u,i'} - \bar{R}_{i'})}{|I_u|} \quad (2\text{-}7)$$

ΔR_u adalah rata-rata deviasi rating untuk semua user pada cluster u Dimana $I_u \in U$ yaitu himpunan user pada U yang memberikan rating pada item i [9].

2.4.3 Neighbor Pre-Selection

Pemilihan tetangga terdekat sangat penting pada *collaborative filtering* dalam menentukan prediksi. Biasanya pemilihan tetangga ini dilakukan dengan melakukan komputasi pada keseluruhan database yanng dapat menyebabkan masalah skalabilitas. Namun hal itu dapat dihindari dengan melakukan komputasi pada *cluster* tertentu saja, yaitu pada *cluster* yang paling mirip dengan user aktif. Kemiripan *cluster* dapat dihitung dengan persamaan berikut [9].

$$sim_{a,u} = \frac{\sum_{i \in (I_a) \cap (I_u)} \Delta R_u (R_{a,i} - \bar{R}_a)}{\sqrt{\sum_{i \in (I_a) \cap (I_u)} (\Delta R_u)^2} \sqrt{\sum_{i \in (I_a) \cap (I_u)} (R_{a,i} - \bar{R}_a)^2}} \quad (2\text{-}8)$$

Proses ini akan menghasilkan cluster termirip dengan user aktif. Data rating pada *cluster* ini nantinya akan digunakan dalam menentukan top-*K* user [9].

2.4.4 Neighbor Selection

Setelah melakukan pemilihan cluster selanjutnya akan dilakukan komputasi kembali terhadap user pada cluster yang terpilih untuk mengambil sebanyak top-*K* user dengan menggunakan persamaan (2-9).

$$sim_{a,u} = \frac{\sum_{i \in (I_a)} w_i (R_{a,i} - \bar{R}_a)(R_{u,i} - \bar{R}_u)}{\sqrt{\sum_{i \in (I_a)} w_i^2 \cdot (R_{a,i} - \bar{R}_a)^2} \sqrt{\sum_{i \in (I_a)} (R_{u,i} - \bar{R}_u)^2}} \quad (2\text{-}9)$$

Pada persamaan (2-9) terdapat nilai w_i yang nilainya didapat dari persamaan berikut.

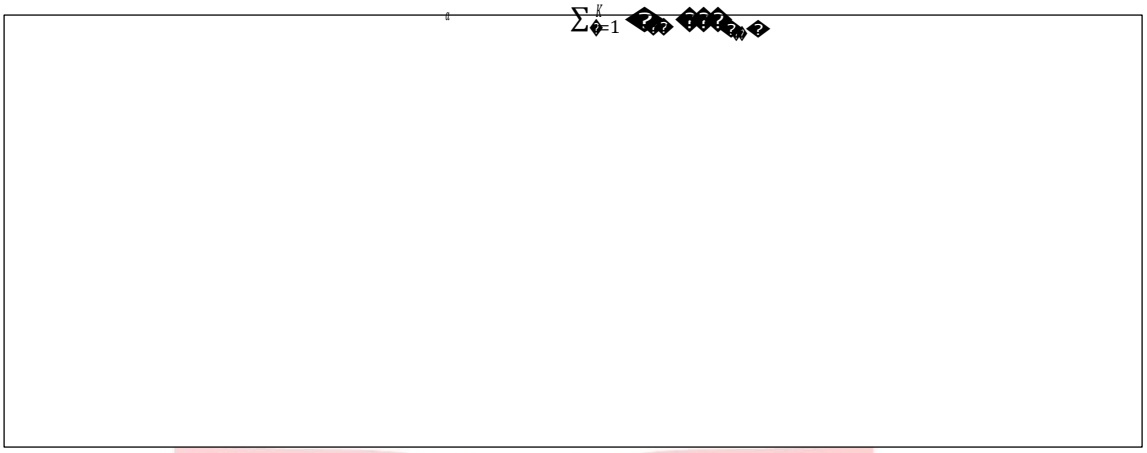
$$w_i = \begin{cases} 1 - \lambda & \text{jika } R_{u,i} \text{ tidak kosong} \\ \lambda & \text{jika } R_{u,i} \text{ kosong} \end{cases} \quad (2\text{-}10)$$

Dimana nilai λ bernilai antara 0-1. λ adalah parameter yang digunakan untuk mengatur nilai rating user ($R_{u,i}$). Karena pada proses *smoothing* sebelumnya mengakibatkan nilai rating user mengandung nilai rating user sebenarnya dan nilai rating dari grup *cluster* [9].

2.4.5 Prediksi

Prediksi rating user aktif dilakukan terhadap top-*K* user yang telah terpilih. Nilai prediksi dihitung menggunakan persamaan berikut.

$$P_{a,i} = \bar{R}_a + \frac{\sum_{u=1}^{n} sim_{a,u} \cdot (R_{u,i} - \bar{R}_u)}{\sum_{u=1}^{n} sim_{a,u}} \quad (2\text{-}11)$$



Dengan ����(�, �) adalah nilai similaritas antara user aktif dengan user top-*K* yang telah terpilih sebelumnya [9].

2.4.6 Mean Absolute Error

Mean Absolute error (MAE) adalah nilai rata-rata error dari hasil prediksi. MAE dapat dihitung menggunakan persamaan berikut.

$$MAE = \frac{\sum_{i=1}^{n} |p_i - \tilde{r}_i|}{|T|} \quad (2\text{-}12)$$

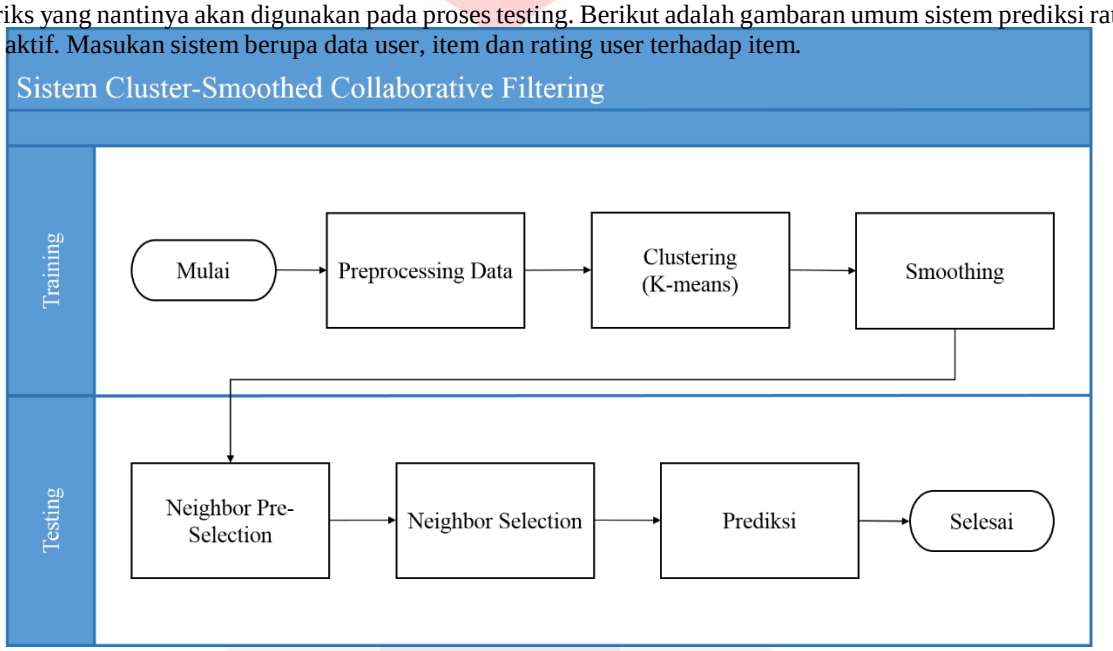
Dimana r_i adalah rating yang diberikan user terhadap item i, $\tilde{r}_i$ adalah rating prediksi yang diberikan user terhadap item i, dan T adalah himpunan data testing [9].

2.5 Dataset

Dataset yang digunakan berasal dari website *https://movielens.org/* dengan jumlah user 943, 1682 item dan 100.000 rating. Setiap user setidaknya memberikan rating terhadap 20 item dengan nilai rating antara 1-5. Untuk proses training menggunakan 80% dari total data dan proses testing menggunakan 20% dari keseluruhan data.

3. Perancangan Sistem

Pada tugas akhir ini dibuat sistem *collaborative filtering* yang berfungsi untuk prediksi rating dari user aktif. Sistem ini terdiri dari proses training dan proses testing. Proses training digunakan untuk membangun model matriks yang nantinya akan digunakan pada proses testing. Berikut adalah gambaran umum sistem prediksi rating user aktif. Masukan sistem berupa data user, item dan rating user terhadap item.

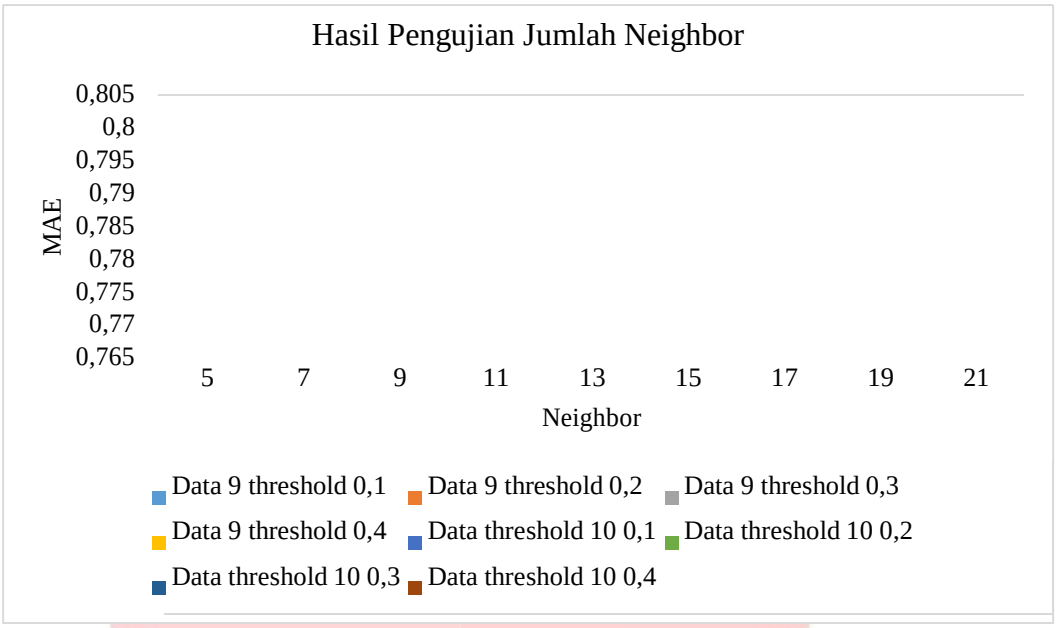


Gambar 3-2 Flowchart Sistem

Dari data awal yang terdiri dari user id, item id dan nilai rating diolah menjadi bentuk matriks *m* x *n* sehingga siap diolah pada proses *clustering*. Setelah terbentuk *cluster*, data memasuki proses *smoothing*. Proses smoothing ini yang akan mengubah data matriks menjadi data training. Data training inilah yang akan digunakan dalam membantu proses prediksi dari data testing sampai menghasilkan MAE sistem.

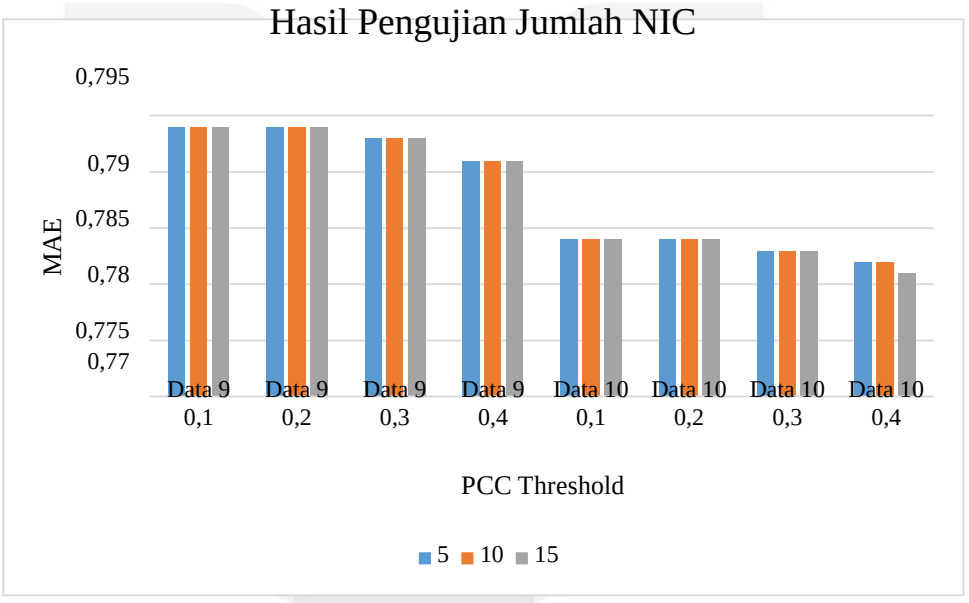
4. Hasil Pengujian dan Kesimpulan

Pengujian dilakukan dengan beberapa parameter uji seperti jumlah *neighbor*, jumlah NIC dan nilai lamda. Jumlah neighbor adalah banyaknya user yang akan dipilih menjadi top-*K* user sebagai bahan pertimbangan dan kalkulasi prediksi rating user aktif.



Gambar 4-3 Grafik Hasil Pengujian Jumlah Neighbor

Berdasarkan grafik terlihat bahwa jumlah neighbor sangat mempengaruhi sistem prediksi. Semakin banyak jumlah neighbor yang diambil disertai dengan nilai PCC threshold yang tinggi dapat memperbaiki nilai MAE sistem.

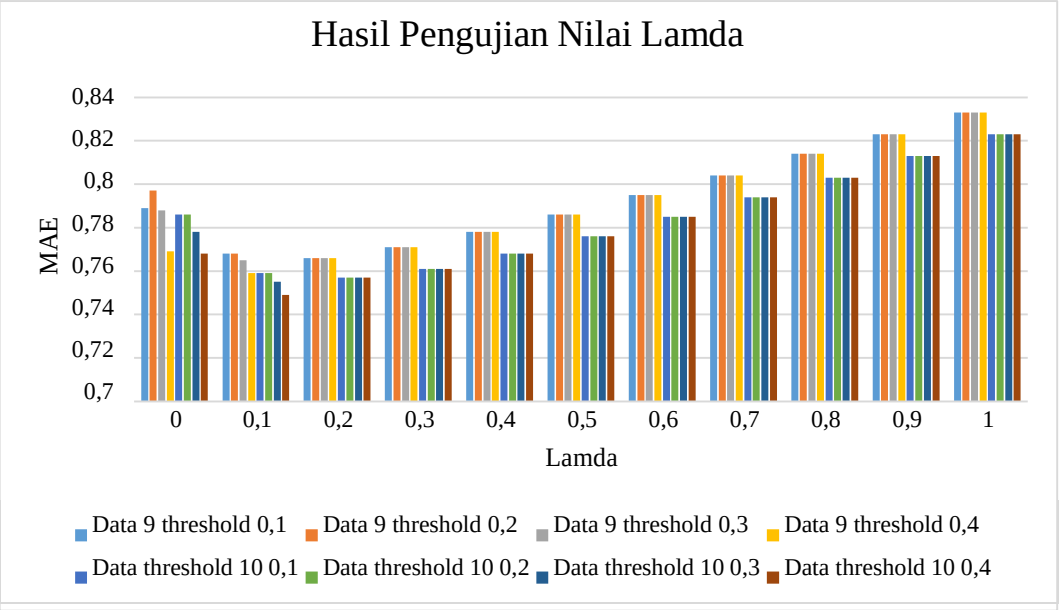


Gambar 4-2 Grafik Hasil Pengujian Jumlah NIC

Jumlah NIC merupakan banyaknya item yang saling beririsan antara user aktif dengan user pada neighbor. Variabel ini berpengaruh ketika proses pengambilan neighbor. Pada saat jumlah NIC bernilai 5 maka user yang masuk sebagai neighbor setidaknya telah memberikan rating pada item yang sama dengan user aktif sebanyak minimal 5 item.

Berdasarkan grafik hasil pengujian jumlah NIC terlihat bahwa banyaknya NIC tidak terlalu berpengaruh terhadap MAE sistem. Namun ada kalanya pengambilan neighbor yang mempertimbangkan jumlah NIC sangat berpengaruh dan dapat memperbaiki performansi sistem.

Terakhir adalah pengujian terhadap nilai lamda dengan kombinasi PCC threshold. Hasil pengujian ini disajikan dalam bentuk grafik berikut.



Gambar 4-3 Grafik Hasil Pengujian Nilai Lamda

Dari hasil pengujian dapat dilihat bahwa nilai lamda terbaik adalah ketika lamda bernilai 0,1. Karena ketika nilai lamda diatas nilai tersebut maka prediksi rating akan sangat bergantung dengan nilai rating cluster, hal ini dapat menyebabkan performansi sistem menurun. Sedangkan ketika nilai lamda kurang dari nilai tersebut maka prediksi rating kurang memperhatikan nilai rating cluster, dimana data rating akan menjadi cukup *sparse* sehingga performansi sistem pun dapat menurun.

Berikut merupakan hasil pengujian dari beberapa algoritma yang digunakan dalam penelitian ini sekaligus perbandingannya dengan algoritma lain :

Tabel 4-2 Perbandingan Hasil MAE Sistem

Algoritma	MAE	
	Data 9	Data 10
PCC dengan Data Non-*Smoothing*	0,754	0,754
Cluster-Smoothed dengan Semua *Neighbor*	0,764	0,765
Cluster-Smoothed dengan Beberapa *Neighbor*	0,809	0,787
Random Cluster-Smoothed	0,746	0,732
Min	0,746	0,732

Dari tabel diatas menunjukkan bahwa penanganan data sparsity menggunakan nilai smoothing dapat menurukan error sistem sebesar ± 0,015, namun melihat perancangan sistem yang dibangun selisih error yang dihasilkan tidak sebanding dengan biaya komputasi yang telah dikeluarkan.

Daftar Pustaka:

[1] X. T. M. K. Su, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, 2009.

[2] P. e. a. Resnick, "GroupLens: an open architecture for collaborative filtering of netnews," *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 1994.

[3] J. S. D. H. C. K. Breese, "Empirical analysis of predictive algorithms for collaborative filtering," *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, 1998.

[4] I. C. N. Soboroff, "Collaborative filtering and the generalized vector space model," *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, 2000.

[5] A. K.-B. Merialdo, "Clustering for Collaborative Filtering Applications," *In Computational Intelligence for Modelling, Control & Automation*, 1999.

[6] L. H. D. P. F. Ungar, "Clustering methods for collaborative filtering," *AAAI workshop on recommendation systems*, 1998.

[7] T. J. P. Hofmann, "Latent class models for collaborative filtering," *IJCAI*, vol. 99, 1999.

[8] D. M. e. a. Pennock, "Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach," *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, 2000.

[9] G.-R. e. a. Xue, "Scalable collaborative filtering using cluster-based smoothing," *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, 2005.

[10] M. J. H. O'Connor, "Clustering items for collaborative filtering," *Proceedings of the ACM SIGIR workshop on recommender systems*, 1999.

[11] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, 1967.

[12] M. e. a. Grcar, "Data sparsity issues in the collaborative filtering framework," 2006.

[13] B. e. a. Sarwar, "Item-based collaborative filtering recommendation algorithms," *Proceedings of the 10th international conference on World Wide Web*, 2001.

[14] B. e. a. Mobasher, "Effective attack models for shilling item-based collaborative filtering systems," *Proceedings of the 2005 WebKDD Workshop, held in conjuction with ACM SIGKDD'2005*, 2005.

[15] P.-A. W. N. C. Z. Chirita, "Preventing shilling attacks in online recommender systems," *Proceedings of the 7th annual ACM international workshop on Web information and data management*, 2005.