TESIS – TE142599

PENGGABUNGAN DATA AKADEMIK BERBASIS *ENTITY RESOLUTION* MENGGUNAKAN MARKOV LOGIC NETWORKS

M. LUKLUK
07111650067001

DOSEN PEMBIMBING
Dr. Ir. Achmad Affandi, DEA
Mochamad Hariadi, ST., M.Sc., Ph.D

PROGRAM MAGISTER
BIDANG KEAHLIAN TELEMATIKA
DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI ELEKTRO
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018

TESIS – TE142599

PENGGABUNGAN DATA AKADEMIK BERBASIS ENTITY RESOLUTION MENGGUNAKAN MARKOV LOGIC NETWORKS

M. LUKLUK
07111650067001

DOSEN PEMBIMBING
Dr. Ir. Achmad Affandi, DEA
Mochamad Hariadi, ST., M.Sc., Ph.D

PROGRAM MAGISTER
BIDANG KEAHLIAN TELEMATIKA
DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI ELEKTRO
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018

LEMBAR PENGESAHAN

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Teknik (M.T)
di
Institut Teknologi Sepuluh Nopember

oleh:

M. Lukluk
NRP. 07111650067001

Tanggal Ujian : 6 Juli 2018
Periode Wisuda : September 2018

Disetujui oleh:

1. Dr. Ir. Achmad Affandi, DEA (Pembimbing I)
   NIP. 196510141990021001
2. Mochamad Hariadi, S.T., M.Sc., Ph.D. (Pembimbing II)
   NIP. 196912091997031002
3. Prof. Dr. Ir. Mauridhi Hery Purnomo, M.Eng (Penguji)
   NIP. 195809161986011001
4. Dr. Ir. Endroyono, DEA (Penguji)
   NIP. 196504041991021001
5. Dr. Adhi Dharma Wibawa, S.T., M.T. (Penguji)
   NIP. 197605052008121003

Dekan Fakultas Teknologi Elektro

Dr. Tri Arief Sardjono, S.T., M.T.
NIP. 197002121995121001

Halaman ini sengaja dikosongkan

PERNYATAAN KEASLIAN TESIS

Dengan ini saya menyatakan bahwa isi keseluruhan Tesis saya dengan judul "**PENGGABUNGAN DATA AKADEMIK BERBASIS ENTITY RESOLUTION MENGGUNAKAN MARKOV LOGIC NETWORKS**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 6 Juli 2018

M. Lukluk

NRP. 07111650067001

Halaman ini sengaja dikosongkan

PENGGABUNGAN DATA AKADEMIK BERBASIS ENTITY RESOLUTION MENGGUNAKAN MARKOV LOGIC NETWORKS

Nama mahasiswa : M. Lukluk
NRP : 07111650067001
Pembimbing : 1. Dr. Ir. Achmad Affandi, DEA
2. Mochamad Hariadi, ST., M.Sc., Ph.D

ABSTRAK

Entity Resolution (ER) adalah permasalahan identifikasi objek yang mengacu pada entitas dunia nyata yang sama ke dalam satu bentuk representasi. Dalam konteks basis data, ER juga dikenal sebagai *record linkage* untuk menentukan rekord yang mengacu pada entitas yang sama, pendekatan probabilistik statistik dari jenis ER ini disebut *probabilistic record linkage* (PRL), dan PRL ini telah digunakan untuk berbagai permasalah ER, termasuk pendekatan turunannya yang menggunakan *machine learning* sebagai perbaikan metodenya. Namun pendekatan probabilistik ini memiliki satu masalah pada ER untuk menangani data kosong yang umumnya terjadi pada dataset yang tidak handal (*unreliable*), data yang *unreliable* tersebut menyebabkan ketidakmenentuan dan dapat mengurangi kualitas hasil akhir. Penelitian ini membahas pendekatan alternatif PRL menggunakan Markov Logic Networks (MLN) untuk inferensi kesesuaian pasangan rekord dalam dataset yang *unreliable*, terutama untuk dataset dengan tingkat data kosongnya (*missing rate*) tinggi. Pendekatan yang diusulkan terinspirasi oleh model *matching dependencies* (MDS) yang secara formal telah diperkenalkan untuk mengatasi dataset yang *unreliable*. Eksperimentasi pada dataset dunia nyata yang diambil dari Universitas Islam Negeri Maulana Malik Ibrahim Malang dilakukan dengan hasil tingkat akurasi 0,977 mendekati 0,986 pada metode sebelumnya yang dapat berguna untuk integrasi data bertahap.

Kata kunci: entity resolution, probabilistic record linkage, matching dependencies, markov logic networks, data integration

Halaman ini sengaja dikosongkan

ACADEMIC DATA MERGING BASED ON ENTITY RESOLUTION USING MARKOV LOGIC NETWORKS

By : M. Lukluk
Student Identity Number : 07111650067001
Supervisor(s) : 1. Dr. Ir. Achmad Affandi, DEA
2. Mochamad Hariadi, ST., M.Sc., Ph.D

ABSTRACT

Entity resolution (ER) is the problem of identifying objects referring to the same real-world entity into a single representation. In the context of the database, ER is also known as record linkage to determine records that refer to the same entities, the statistical probabilistic approach of this type of ER is called probabilistic record linkage (PRL), and PRL has been used for variety ER problems, including derivatives that use machine learning as an improvement. However, this probabilistic approach has one problem in ER for dealing with missing data that commonly occur in unreliable datasets, such unreliable data can lead to more uncertainty and can reduce the quality of the final result. This paper discusses an alternative approach of PRL using a Markov logic networks (MLN) to infer the matching of record pairs in unreliable datasets, especially for datasets with a high rate of missing data. The proposed approach is inspired by a model of matching dependencies (MDS) that has been formally introduced to address unreliable datasets. Experimentation on real-world datasets taken from State Islamic University of Maulana Malik Ibrahim Malang is done with the result of accuracy of 0.977 approaching 0.986 on the previous method which can be useful for gradual data integration.

Key words: entity resolution, probabilistic record linkage, matching dependencies, markov logic networks, data integration

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Alhamdulillahirobbil'alamin, puji syukur atas segala limpahan nikmat dan karunia Allah SWT, Tuhan yang Maha Kuasa. Hanya dengan petunjuk, rahmat dan ridho-Nya, sehingga penulis dapat menyelesaikan tesis ini.

Penulis mengucapkan terima kasih kepada Bapak Dr. Ir. Achmad Affandi, DEA selaku pembimbing pertama dan Bapak Mochamad Hariadi, ST., M.Sc., Ph.D selaku pembimbing kedua yang telah meluangkan waktu dan memberikan masukan sehingga tesis ini dapat selesai.

Penulis juga menyampaikan terima kasih yang tak terhingga kepada:

1. Universitas Islam Negeri Maulana Malik Ibrahim, Kementrian Agama, dan Kementerian Komunikasi dan Informasi yang telah memberikan kesempatan mendapatkan beasiswa Program Magister (S2) Telematika/ Pengelola Teknologi Informasi dan Komunikasi pada Institut Teknologi Sepuluh Nopember Surabaya.
2. Prof. Ir. Joni Hermana, M.Sc.Es, Ph.D., selaku Rektor Institut Teknologi Sepuluh Nopember Surabaya.
3. Dr. Tri Arief Sardjono, S.T., M.T., selaku Dekan Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember Surabaya.
4. Dr. Ir. Wirawan, DEA, selaku Kepala Program Studi Pascasarjana Fakultas Teknologi Elektro.
5. Dr. Adhi Dharma Wibawa, ST, MT, selaku Koordinator Bidang Keahlian Telematika/Pengelola Teknologi Informasi dan Komunikasi (PETIK), beserta Bapak Eko Setijadi, ST., MT., PhD selaku Dosen Wali Akademik PETIK 2016 Jurusan Teknik Elektro, atas kesabaran, arahan, dan bimbingan kepada kami semua.
6. Seluruh Pengajar dan staf Program Studi Magister (S2) Departemen Teknik Elektro, Bidang Keahlian Telematika/PETIK, atas jasa dan pengabdiannya dalam mendidik dan mendewasakan kami.
7. Orang tua tercinta *Buya* Mursyid Alifi (almarhum) dan *Ummi* Hamimah, serta *Bapak* Abdul Muhyi dan *Emak* Nihayah, terimakasih tak terhingga atas segala

do’a dan kasih sayang serta dukungan sehingga penulis dapat menyelesaikan tesis ini tepat waktu.

8. Istri tercinta Nurun Nayiroh, serta anak-anak tersayang Firda Najwa Meutia dan Muhammad Iqbal Habibie, atas segala dukungan, kesabaran, cinta, kasih-sayang, dan do’a yang selalu mengiringi sampai selesainya studi ini.
9. Rekan-Rekan PETIK 2016 (Mas Alfin, Mas A4, Mas Allan, Mas Aset, Mbak Ajah, Mbak Ika, Mas Gio, Mas Lutfi, Mbak Alfi, dan Mbak Putri) atas semua “hal” yang kita lalui bersama.

Semoga Allah SWT membalas kebaikan semua pihak yang telah memberi kesempatan, dukungan dan bantuan dalam menyelesaikan tesis ini. Penulis menyadari bahwa tesis ini masih jauh dari sempurna, oleh karena itu saran dan kritik yang membangun sangat diharapkan demi kesempurnaan tulisan ini, sehingga tesis ini memberikan manfaat yang baik bagi agama, bangsa, dan negara.

Surabaya, 6 Juli 2018

Penulis

DAFTAR ISI

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1 Latar Belakang

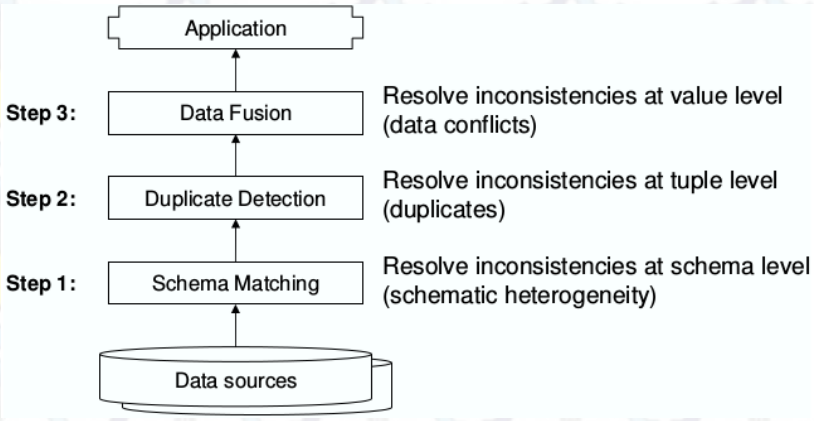
Pengembangan Sistem Informasi (SI) yang terintegrasi membutuhkan perencanaan awal yang kompleks dan komprehensif, disertai dengan kebutuhan biaya dan sumber daya yang memadai mengikuti kompleksitas model bisnis yang tercakup di dalamnya. Untuk memperkecil kompleksitas, biasanya roadmap SI terintegrasi dipecah menjadi beberapa SI kecil yang saling independen untuk dikembangkan secara bertahap maupun parallel namun secara independen, model pengembangan ini disebut sebagai pengembangan evolusif.

Pengembangan SI secara evolusif ini sering diterapkan oleh tim teknis yang berbeda, dan terkadang tidak saling terkoordinasi dalam tahapannya, baik karena beda divisi, beda waktu pengerjaan, maupun beda vendor atau pengembang, sehingga dalam jangka panjang ketika kumpulan SI ini telah membesar dan model bisnisnya telah terikat secara operasional, maka mulai muncul permasalahan sinkronisasi data, bukan hanya karena database yang diterapkan berbeda dan terpisah, namun bisa karena aplikasi RDBMS yang digunakan juga berbeda, khususnya untuk SI yang dikembangkan oleh pihak ketiga yang biasanya memiliki keterbatasan akses langsung ke database, misalkan hanya dapat akses ke tabel pelaporan atau ke tabel hasil *export* ke dokumen tertentu (keterbatasan akses data ini disebut bersifat *privacy confidential*).

Pada institusi kecil-menengah khususnya di institusi Akademik, permasalahan sinkronisasi data sangat mungkin terjadi, khususnya ketika pihak pemegang kebijakan mulai membutuhkan pelaporan yang bersifat komprehensif untuk mendukung keputusan-keputusan strategis. Mengingat keterbatasan tenaga ahli atau *expertise* teknologi informasi di institusi, maka pengembangan SI cenderung dilakukan secara evolusif, yang tidak jarang menggunakan SI pihak ketiga yang sudah hampir pasti tidak sepenuhnya kompatibel dengan terapan SI yang sudah ada sehingga menambah sifat heterogensi data yang dihasilkan.

Sinkronisasi data pada SI yang tidak terintegrasi merupakan proses yang rumit dan butuh ketelitian tinggi, baik dengan menggunakan pendekatan ETL (*Extract Transform Load*) untuk integrasi sebuah data *warehouse*, SOA (*Service Oriented Architecture*) yang berupa integrasi sisi modul aplikasi berbasis komponen, maupun dengan *web service* berupa antarmuka lintas SI melalui modul tertentu, hampir semua pendekatan integrasi membutuhkan perencanaan dan pendampingan intensif dengan disertai kualifikasi tenaga ahli yang baik serta proses yang memakan waktu. Sehingga dikembangkan sebuah topik khusus mengulas tentang aspek-aspek integrasi data, terutama di era *cloud computing* dan Big Data yang menawarkan kemudahan akses data namun dihadapkan dengan tantangan muatan (*volume*), pertumbuhan (*velocity*), keberagaman (*variety*), keterperincian/keutuhan (*veracity*), dan keberhargaan (*value*) data.

Salah satu sub-topik dari konsep integrasi data adalah penggabungan data (*data fusion*) sebagaiman ilustrasi pada Gambar 1.1, sebuah sub tahapan akhir integrasi yang fokus untuk memastikan kualitas data hasil keluaran yang bersih (*data cleansing*), yang mencerminkan kondisi riil dan ideal dari data di dunia nyata (*real-word entity trustworthy*), seperti tidak adanya entitas data yang berulang, tidak ada data yang konflik/ambigu, serta relasi data tetap utuh dan dapat dirunut dengan benar. Di dalam proses penggabungan data ini terdapat konsep *entity resolution* (ER) yang dikenal memiliki banyak sebutan sesuai penekanan konteks pada domain kajiannya, seperti *conflict resolution*, *entity deduplication*, *record linkage*, *data cleansing*, atau *object identification-consolidation*.



Gambar 1.1: Tiga Tahapan Penggabungan Data pada Integrasi Data [1]

Pada kenyataannya ER ini mirip dengan proses audit yang sangat rumit, memakan waktu, membutuhkan keahlian khusus (*expertise*), dan bersifat padat karya (*labour intensive*), sehingga untuk membantu mengatasi permasalahan ini pendekatan untuk mengotomatisasi proses banyak diajukan, baik dengan menggunakan pendekatan deterministik/eksak maupun probabilistik [2], serta menggunakan estimasi statistik maupun *machine learning* untuk setidaknya membantu porsi *expertise* di dalam proses penggabungan data.

Dalam konteks *data cleansing* pada database, pembahasan permasalahan ER pada domain kajian ini disebut sebagai *record linkage*, yang tujuannya untuk menghubungkan dua *record* pada tabel di database yang diyakini mengarah ke entitas yang sama, sehingga duplikasi dan redundansi data dapat dikurangi untuk menghasilkan kualitas data yang baik. *Record linkage* memiliki dua strategi utama, yaitu deterministik yang merupakan model eksak berdasarkan penyesuaian kunci (*matching key*), dan model probabilistik yang menggunakan *log-likelihood* untuk memutuskan kesesuaian pembandingan *record* yang dikenal dengan konsep *Probabilistic Record Linkage* (PRL). Pendekatan probabilistik ini sebenarnya mirip dengan model klasifikasi biner (yaitu sesuai dan tak-sesuai), sehingga pada pengembangan lanjutan telah dilakukan menggunakan *machine learning* dengan *naïveBayes classifier* [3] untuk proses inferensi *record linkage*. Namun pengembangan model inferensi ini masih memiliki permasalahan utama bahwa PRL masih menggunakan asumsi bahwa tiap field pada record adalah saling independen [4], yang dapat menurunkan performa jika tingkat unreliabilitas data cukup tinggi dikarenakan banyaknya data kosong pada dataset.

Salah satu metode *machine learning* yang dapat digunakan di dalam permasalahan ER adalah Markov Logic Networks (MLN) yang merupakan penyederhanaan dari konsep Markov Networks atau Markov Random Fields. MLN adalah sebuah pendekatan sederhana hasil penggabungan antara model *first-order-logic* dengan model *graphical probabilistic* (*statistical artificial intelligent*) dalam satu penyajian, metode ini dikembangkan oleh Matthew Richardson dan Pedro Domingos pada tahun 2006 di universitas Washington Seattle USA [5]. Kelebihan dari MLN ini adalah kemampuannya untuk melakukan kombinasi klausa-klausa pada proses penyimpulan, melalui bobot pembelajaran yang dikaitkan dengan

klausa tersebut, sehingga MLN dapat menangani ketakmenentuan (*uncertainty*) dan memungkinkan kondisi tak sempurna (*imperfect*) serta pengetahuan kontradiktif (*contradictory knowledge*).

Penggunaan MLN untuk proses inferensi pada permasalahan ER telah digunakan untuk permasalahan ER, salah satunya adalah metode pembagian dua tahap (*2-stages* MLN) *training* dan inferensi yang dicetuskan oleh Qing-zhong [6] dengan capaian akurasi berkisar antara 93% sampai dengan 95%, yang cukup membuktikan bahwa MLN adalah pendekatan probabilistik yang layak digunakan pada permasalahan ER.

Sedangkan untuk pendekatan deterministik pada permasalahan ER, *matching dependencies* (MDS) adalah metode yang fokus pada performa kecepatan dan skalabilitas, walaupun metode ini lebih ditekankan untuk *data cleansing* pada dataset yang *unrliable* (dari pada *linking* sebagaimana pada *record-linkage* [7]), namun berada dalam konteks domain permasalahan ER pada tabel di database. MDS mengajukan aturan formal identifikasi kesesuaian dua pasang *record* (*record pair*) melalui alternatif pemeriksaan kesamaan maupun kemiripan dari field-field yang ada di *record pair* yang disebut sebagai *relative candidate keys* (RCKs), serta menentukan aturan eksak untuk proses inferensi.

Pengembangan algoritma PRL untuk dataset yang *unreliable* sebenarnya sudah diajukan dengan algoritma terbaik disebut *Full Linkage Expansion* (FLE) [4] yang pada prinsipnya sama dengan RCKs pada MDS, perbedaanya adalah bahwa FLE hanya berisi kumpulan field cadangan yang diurut (secara manual) berdasarkan prioritas yang dapat dijadikan sebagai alternatif field kunci (*quasi-identifier* -- QID) pengidentifikasi *record* yang disebut dengan *backup fields* (BK-*fields*). Model BK-*fields* pada FLE ini lebih sederhana dan tidak sefleksibel model RCKs yang diajukan pada metode MDS yang berupa aturan filter beberapa kemungkinan kombinasi field-field.

Pada penelitian ini diajukan pendekatan baru untuk memperluas model BK-*fields* pada FLE menggunakan RCKs dari metode MDS untuk kemudian dilakukan proses inferensi *machine learning* dengan Markov Logic Networks (MLN). Kekuatan inferensi logikal probabilistik dari MLN digabungkan dengan fleksibilitas RCKs sehingga dapat menampung kondisi tambahan semisal meninjau

konsistensi *record* sebagai pertimbangan keputusan kesesuaian pembandingan *record*. Penggabungan data lingkungan akademik UIN Malang sangat sesuai dengan situasi tingginya unreliabilitas pada dataset di dunia nyata, yang dapat digunakan untuk pengujian performansi dari pendekatan yang diajukan pada penelitian ini. Tingginya tingkat unreliabilitas ini mengakibatkan pula meningkatnya kondisi *uncertainty* dikarenakan banyaknya data kosong hasil dari penggabungan data, sehingga penggunaan *machine learning* dapat menjadi solusi untuk membantu atau bahkan dapat menggantikan *expertize* dalam hal proes pengolahan hasil penggabungan, terutama pembersihan data (*data cleansing*) dan pengkaitan data rekaman (*record linkage*) melalui konsep ER. Pertimbangan lanjutan penggunaan *machine learning* ini adalah skalabilitas kuadratik O(n^2) pada pembandingan rekord (*record pairing*) pada permasalahan ER yang cukup memakan waktu jika dilakukan secara manual (misalkan dengan query database saja) walaupun jika menggunakan keterampilan tenaga ahli [8] [9].

1.2 Rumusan Masalah

Entity Resolution dengan pendekatan deterministik untuk *data cleansing* sudah mengantisipasi permasalahan ketidaklengkapan data (*data unreliability*), sebuah permasalahan yang belum menjadi fokus utama pada pendekatan probabilistik, yaitu *probabilistic record linkage* (PRL), baik PRL klasik maupun model terbaru yang menggunakan *neural network*, terutama untuk tingkat unrealibilitas data yang tinggi (tingkat unrealibilitas > 25%).

Pendekatan baru dengan menggunakan metode inferensi Markov Logic Networks diajukan pada penelitian ini dengan memperluas parameter tambahan sebagai tinjauan kondisional pada dataset, yang diharapkan dapat meningkatkan performa ER atau setidaknya tetap mempertahankannya, khususnya jika pendekatan ini diuji pada dataset dunia nyata, yaitu penggabungan database yang cenderung memiliki tingkat unreliabilitas data yang tinggi.

1.3 Tujuan

Mendapatkan hasil pengujian ER dengan bantuan *machine learning* yaitu inferensi dengan metode MLN pada penggabungan data lingkungan Akademik di UIN Malang yang dataset hasil penggabungannya bersifat *unreliable*, berdasarkan pendekatan pada model penelitian sebelumnya, yaitu MDS, PRL, dan FLE.

Penelitian ini juga ditujukan untuk memberikan alternatif ER yang dapat memperluas fitur inferensi, seperti mempertimbangkan fitur pendukung dan kecenderungan inambiguitas/konsistensi *record*.

1.4 Batasan Masalah

Penelitian ini dilakukan untuk menguji performa ER sesuai dengan metode yang diajukan, dengan mengecualikan hal-hal terkait yang membutuhkan kajian lanjutan agar konsistensi pembahasan tetap terfokus. Hal-hal terkait tersebut semisal pemilihan *threshold* similaritas yang optimal, pengujian performa untuk skalabilitas, dan optimasi *record pairing* (*indexing*).

Data sampel diambil dari data terkait kegiatan Akademik di lingkungan Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang, yang meliputi data-data Sistem Informasi saling independen baik data untuk kegiatan akademik kampus maupun data operasional terkait seperti data akademik mahasiswa, data pengabdian masyarakat, data penelitian dosen, dan lain-lain.

1.5 Kontribusi

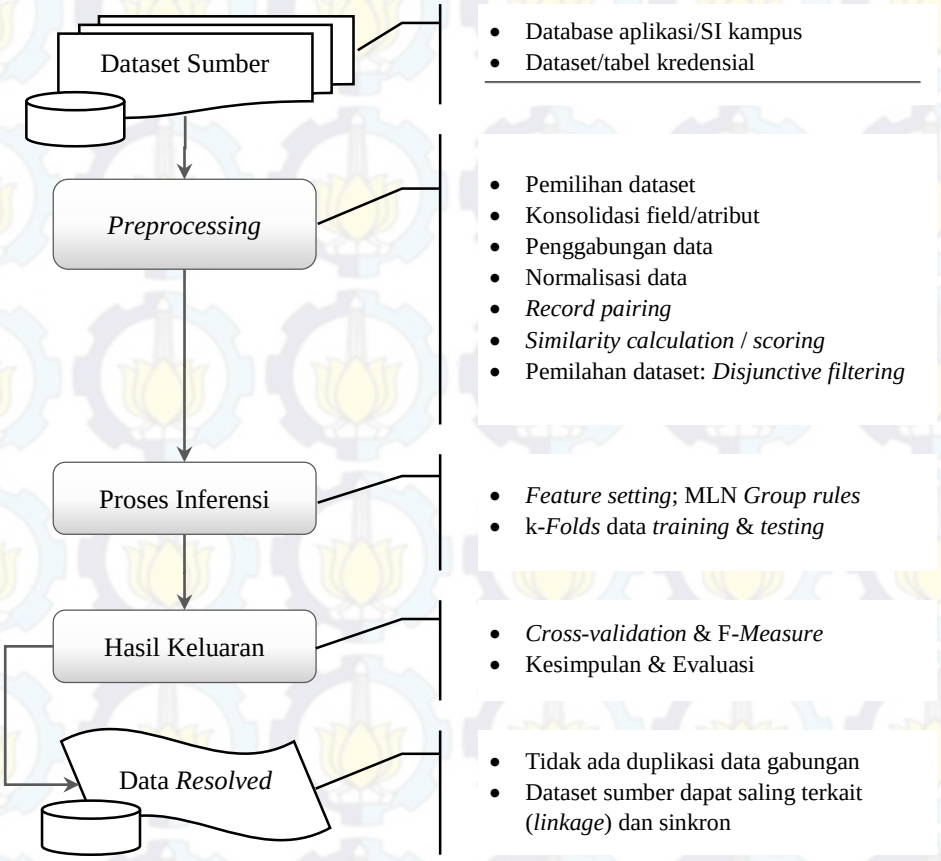
Hasil dari penelitian ini dapat digunakan sebagai solusi parsial pada permasalahan sinkronisasi dan integrasi data pada instansi akademik yang menerapkan Sistem Informasi secara evolusif, yaitu pada pembersihan data (*data cleansing*) terutama untuk institusi kecil-menengah yang dapat membantu penyajian data pelaporan untuk pendukung keputusan strategis akademik kampus, khususnya di lingkungan UIN Maulana Malik Ibrahim Malang.

Pendekatan yang digunakan juga dapat menjadi alternatif kebutuhan fleksibilitas pada klausa aturan inferensi pada permasalahan *record linkage*, mengikuti kekuatan penyajian aturan pada metode probabilistik yang digunakan yaitu metode Markov Logic Networks.

1.6 Metodologi Penelitian

Penelitian ini menggunakan metode MLN untuk data dengan ureliabilitas tinggi pada penggabungan database, melalui pemeriksaan relevansi kesesuaian pasangan rekord (*record pair*) pada tabel gabungan. Variabel bebas atau fitur diambil pecahan kelompok field berdasarkan karakteristik keunikan nilai field sebelum dan sesudah penggabungan, yang pemecahannya dilakukan mengikuti hasil pengukuran nilai kemiripan (*similarity*) tiap field di *record pair*, pecahan tiap level membentuk filter RCKs. Hasil bentukan fitur kemudian dibangun aturan formula logika *first-order-logic* untuk dilakukan proses pelatihan (*training*) dan penyimpulan (*inference*) menggunakan metode MLN.

Diagram alur pada Gambar 1.2 adalah ringkasan metodologi pada penelitian ini yang dijelaskan secara lebih terinci di Bab 3.



Gambar 1.2: Alur Ringkasan Metodologi Penelitian

Halaman ini sengaja dikosongkan

BAB 2
KAJIAN PUSTAKA

Sebelum melangkah ke tahapan pembahasan penelitian, terdapat beberapa kajian terkait yang menjadi acuan pada penelitian ini, kajian tersebut meliputi penelitian terkait serta teori dasar. Sub-sub bab berikut dipaparkan beberapa paper yang membahas tentang *entity resolution* khususnya di domain bahasan *record linkage* baik pendekatan deterministik maupun probabilistik, juga dijelaskan beberapa kajian teroretik yang mendasari pembahasan pada bab-bab selanjutnya.

2.1 Kajian Penelitian Terkait

Entity resolution (ER) merupakan domain kajian yang luas yang berada dalam sub kajian penggabungan data (*data fusion*) dalam ruang pembahasan Integrasi data. *Record linkage* yang merupakan sub spesifik kajian ER lebih menekankan konteks ER pada bentuk data terstruktur seperti rekord-rekord pada table di database relasional, mengingat pembahasan tentang *record linkage* ini muncul dari persoalan penghubungan data (*data linking*) pada skematik database relasional.

Record linkage memiliki dua pendekatan, yaitu pendekatan deterministik dan pendekatan probabilistik, di mana kedua pendekatan tersebut memiliki area bahasan penelitian tersendiri, walaupun secara umum masih berhubungan. Pendekatan yang digunakan pada penelitian ini adalah pendekatan probabilistik, karena merupakan turunan dari model pendekatan yang disebut dengan *probabilistic record linkage* (PRL).

Daftar penelitian pada Tabel 2.1 memuat dua pertama merupakan penelitian dengan pendekatan deterministik yang terkait dengan penelitian ini, sedangkan tiga berikutnya adalah pendekatan probabilistik, khususnya untuk nomor 4 dan 5 yang merupakan penelitian pendahulu, sedangkan nomor 3 adalah penelitian ER yang menggunakan metode yang sama, yaitu Markov Logic Networks.

Tabel 2.1: Daftar Penelitian Pendukung Terkait

No.	Penelitian Terkait	
1.	Judul:	Efficient discovery of similarity constraints for matching dependencies [10]
	#Sampel:	-
	Variable:	-
	Metode:	*Pruning* dan *Approximation* untuk *Matching Dependencies* (MDS)
	Validasi:	-
	Hasil:	*Support-Confidence threshold* untuk efisiensi pada MDS
2.	Judul:	ERBlox: Combining matching dependencies with machine learning for entity resolution [11]
	#Sampel:	250 ribu author dan 2,5 juta paper
	Variable:	-
	Metode:	SVM, MDS, Answer Set Program (ASP: ERBlox)
	Validasi:	-
	Hasil:	*Collective Blocking* menggunakan SVM memiliki kinerja ER untuk data relasional dengan presisi-*recall* terbaik
3.	Judul:	Data Conflict Resolution with Markov Logic Network [6]
	#Sampel:	1248 buku dan 250 film
	Variable:	*Basic, implication, inter-dependency*
	Metode:	Discriminative Markov Logic Networks (MLN)
	Validasi:	*Holdout*
	Hasil:	Perbaikan akurasi-presisi dengan metode MLN
4.	Judul:	Beyond Probabilistic Record Linkage: Using Neural Networks and Complex Features to Improve Genealogical Record Linkage [3]
	#Sampel:	80000 pasang rekord genealogi
	Variable:	*Full featured sliced (dis)agreement value*
	Metode:	Neural Network (NN) naïve Bayes classifier
	Validasi:	*Holdout*
	Hasil:	Kenaikan Signifikan akurasi-presisi dengan NN
5.	Judul:	Improving record linkage performance in the presence of missing linkage data [4]
	#Sampel:	Simulasi data dengan 5000 rekord
	Variable:	*(dis)agreement value*
	Metode:	Weight Redistrubution, Distance Inputation, Full Linkage Expansion (FLE), Compact Linkage Expansion
	Validasi:	-
	Hasil:	FLE memiliki kinerja terbaik

Penelitian ini menawarkan alternatif dari paper nomor 4, untuk melakukan ER pada data dengan tingkat *missing rate* yang tinggi sebagaimana hasil kajian pada paper nomor 5 bahwa FLE memiliki kinerja terbaik untuk data sampel dengan *missing rate* dibawah 25%, sedangkan penelitian ini menggunakan data sampel dengan tingkat *missing rate* di atas 25%.

2.2 Teori Dasar

2.2.1 Data Lingkungan Akademik UIN Malang

Sarana teknologi informasi sudah menjadi kebutuhan mendasar bagi instansi pemerintahan, khususnya kampus, instansi pendidikan yang membidangi *core* bisnis Akademik. Bagi instansi pendidikan, ketersedian dan pemanfaatan infrastruktur Teknologi Informasi dan Komunikasi (TIK) sudah menjadi tolok ukur kemajuan sebuah intansi [12].

Namun pada kenyataannya, penerapan TIK khususnya Sistem Informasi (SI), memerlukan sumber daya biaya, waktu, maupun tenaga yang tidak sedikit, sehingga perencaanaan biasanya memecah terapan menjadi tahapan-tahapan sesuai prioritas dan ketersediaan sumber daya, penerapan bertahap ini disebut sebagai penerapan evolusif. Pengembangan bertahap ini mengakibatkan dataset yang dihasilkan pada tahap operasional pada tiap SI yang dibuat saling terpisah dalam database-database tersendiri yang tidak terkait secara langsung, yang disebut dengan dataset otonom.

Dataset otonom dari terapan SI evolusif, biasanya akan mengalami permasalahan sinkronisasi dataset, yaitu terjadi konflik data saat kebutuhan integrasi mulai diperlukan, seperti kebutuhan data *warehouse* untuk pelaporan-pelaporan level eksekutif, maupun integrasi aplikasi-aplikasi SI agar mengacu ke satu database besar, sehingga dibutuhkan penyelesaian tertentu terhadap permasalahan integrasi tersebut.

Meninjau terjadinya konflik dataset, khususnya dataset bidang Akademik, penerapan SI di lingkungan UIN Maulana Malik Ibrahim Malang dapat menggambarkan contoh permasalahan integrasi pada penerapan SI evolusif-otonom. Tabel pada Gambar 2.1 adalah bagan terapan SI yang sudah dalam tahap operasional dan memerlukan integrasi data untuk dikumpulkan di dalam satu data *warehouse* menjadi sebuah SI khusus pelaporan pendukung keputusan tingkat eksekutif kampus.

SI tingkat eksekutif ini direncanakan untuk mengumpulkan seluruh dataset SI bidang instansi yang ada di dalam institusi UIN Malang yaitu meliputi [13]:

- Bidang Akademik
- Bidang Kemahasiswaan

- Bidang Kepustakaan
- Bidang Penelitian dan Pengabdian Masyarakat
- Bidang Perencanaan dan Penganggaran
- Bidang Pengelolaan Keuangan
- Bidang Sumber Daya Manusia, dan
- Bidang Pendukung Sarana TIK Kampus

Gambar 2.1: Dataset pada Lingkungan Akademik/Kampus UIN Malang [13]

Termasuk bidang yang belum tercakup karena masih dalam tahap perencanaan, yaitu Sarpras (Sarana-Prasarana), Dokumen dan Kearsipan, Kerjasama Inovasi dan Korporasi.

Dari berbagai bidang tersebut, beberapa konflik data teridentifikasi dengan mengambil beberapa penamaan atribut/*field*/kolom tabel dari database yang pada Sistem Informasi SMSBox, Aktivitas, Penelitian, Kepegawaian, Akademik, dan IDM (*Identity Management*) sebagaimana yang ditunjukkan pada Tabel 2.2, pada tabel tersebut terjadi konflik dataset pada atribut entitas yaitu `Nama Pegawai` (kolom *Conflicted Attributes*), dibuktikan dengan beragamnya penyebutan sesuai dengan konteks yang tertuang di masing-masing terapan SI tersebut, walaupun pada dasarnya seluruh dataset pada atribut tersebut sebenarnya mengacu ke satu atribut entitas di dalam institusi, yaitu `Nama Pegawai` saja.

Tabel 2.2: Contoh Identifikasi Beberapa Konflik Dataset

***Data Source* (DB SI)**	***Referred Keys* (1)**	***Conflicted Attributes* (2)**	***Related Attributes* (3)**	***Affected Attributes* (4)**
SMSBox	-	fullname	phone/msisdn	-
Aktivitas	nip/nipt	nama	kode_finger	SatKer
Penelitian	nip/nipt	ketua.nama	telp; email	SatKer
	userID	anggota.nama	telp_rumah; email	SatKer
Kepegawaian	nip_lama; nip_baru	gelar_depan; nama; gelar_blkg	telp1; telp2; email1; email2	-
Akademik	nip_lama; nip	gelardepan; nama; gelarbelakang	kode	-
IDM	code; rcode	name; alias	phone; email	Basecamp

Keterangan kolom pada Tabel 2.2:

DB SI : nama database pada Sistem Informasi

(1) : acuan penghubung data gabungan

(2) : konflik yang perlu diselesaikan dengan ER

(3) : pembantu penyimpulan keterhubungan

(4) : Dataset relasional yang mungkin ikut terimbas konflik akibat proses penggabungan data

Identifikasi konflik dapat dilakukan dengan menentukan atribut kunci (kolom *Referred Keys*) yang menjadi acuan yang berisi nilai unik (walaupun memiliki nama-nama atribut berbeda), pada Tabel 2.2, atribut kunci adalah NIP/NIPT atau Nomor Induk Pegawai yang hanya memiliki satu nilai tiap pegawai instansi.

Sebuah dataset tambahan, yaitu kolom *Related Attributes*, dapat digunakan sebagai atribut pembantu penyimpulan bahwa satu dataset mengacu ke satu entitas yang sama untuk menguatkan pembuktian penyelesaian konflik data. Sedangkan kolom terakhir, yaitu kolom *Affected Attributes*, adalah konflik yang ditimbulkan akibat adanya relasi data pada tabel yang akan digabungkan, sehingga konflik ini perlu menjadi perhatian khusus karena membutuhkan penyelesaian lanjutan.

Tabel 2.3 adalah fragmen konflik isian data (data *value*), yang akan menimbulkan masalah konflik ketika dilakukan sinkronisasi ke satu data *warehouse* SI eksekutif kampus, permasalahan konflik ini membutuhkan penyelesaian yang dijabarkan dalam satu konsepsi khusus yang disebut sebagai *entity resolution* (ER), yang merupakan salah satu bidang kajian di dalam ruang lingkup topik besar pembahasan Integrasi Data.

Tabel 2.3: Contoh Fragmen Konflik Dataset

***Data Source* (DB SI)**	***Referred Keys***	***Conflicted Attributes***	***Related Attributes***	***Affected Attributes***
SMSBox	-	Siti Masitoh, Dra, M.Hum.	08xxx776874	-
Aktivitas	1xxxx020200; 3122001	Siti Masitoh	20129	Fakultas Humaniora
Kepegawaian	15xx31144; 1xxxx020200; 3122001	DRA.; Siti Masitoh; M.Hum	-	-
IDM	15xx31144; 1xxxx020200; 3122001; -	Siti Masitoh; Dra. Siti Masitoh, M.Hum	-;-	Bahasa dan Sastra Inggris

Beberapa konteks atribut yang potensial untuk mengakibatkan konflik pada saat penggabungan data dan membutuhkan ER untuk menyelesaikan konflik tersebut antara lain:

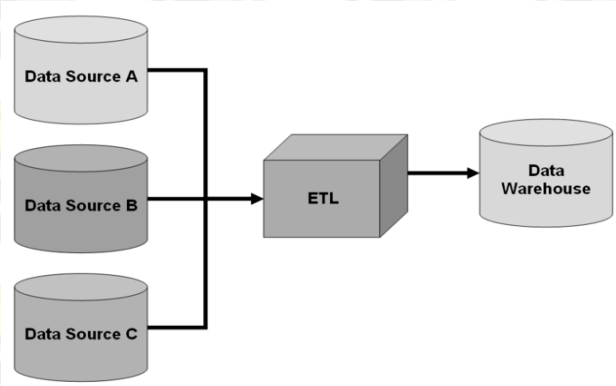
- Nama; baik nama dosen, karyawan, mahasiswa, dan nama alumni
- Kontak person
- Alamat email
- Satuan kerja di dalam instansi
- Meta data penelitian, artikel, publikasi, jurnal, dan tugas akhir

2.2.2 Integrasi Data

Integrasi data merupakan penggabungan data dari beberapa sumber berbeda yang menyajikan satu tampilan berupa hasil proses penggabungan tersebut. Tujuan integrasi data ini agar dapat melakukan *query* dan akses terhadap berbagai

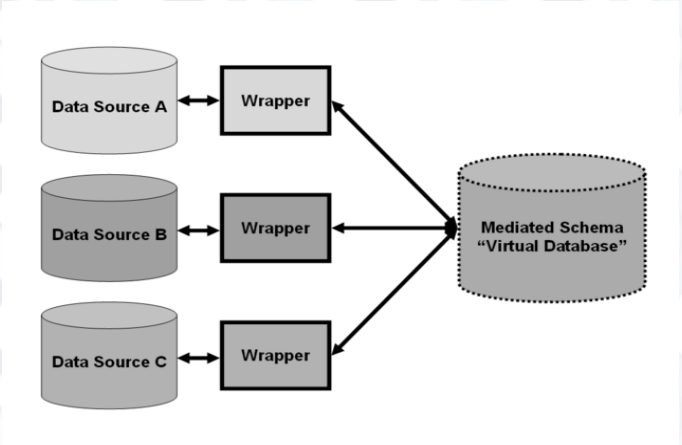
data sumber berbeda untuk menangani skalabilitas data, sifat heterogensi data, sifat otonom, serta sifat semi terstruktur antar sumber data tersebut [14].

Sistem integrasi data pertama kali dilakukan di Universitas Minnesota pada tahun 1991 dengan pendekatan *data warehouse* menggunakan konsep ETL (*Extract Transform Load*) dari beberapa sumber heterogen menjadi satu skema tunggal, pendekatan ini walaupun dapat memuat kapasitas database yang besar namun kelemahan utamanya adalah keterikatan arsitektural yang kuat (*tightly coupled*), karena membutuhkan sinkronisasi data terus-menerus secara fisik.



Gambar 2.2: *Data Warehouse* dengan ETL *(Extract Transform Load)*

Pada tahun 2009 tren integrasi data mulai fokus untuk merenggangkan keterkaitan (*loosly coupled*) arsitektural dengan menempatkan antarmuka *query* untuk mengakses data secara *real-time*, melalui skema perantara (*mediated schema*). Pendekatan ini melakukan pemetaan (*mapping*) antara *mediated schema* dengan skema sumber, dan metransformasi *query* yang sesuai dengan database sumber.



Gambar 2.3: *Virtual Database* melalui *Mediated Schema*

Schema mapping tersebut memiliki dua pendekatan, yaitu:

1. *Global As View* (GAV); pemetaan skema dari *mediated schema* ke data sumber.
2. *Local As View* (LAV); pemetaan skema dari data sumber ke *mediated schema*.

Pendekatan terbaru bahkan menggunakan teknik inferensi untuk memecahkan problem proses *query* pada *mediated schema*.

Konsep integrasi data ini diajukan untuk mengatasi kesulitan-kesulitan yang dihadapi pada *multi-datasource* [15], antara lain:

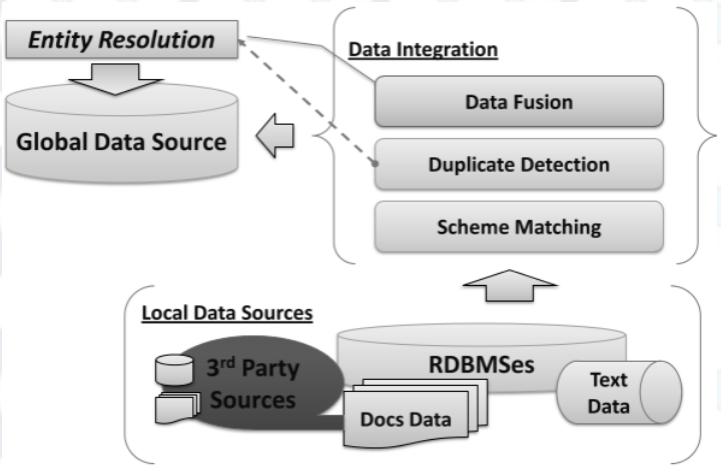
- Pengelolaan platform data yang berbeda
- Eksekusi *query* antar sumber dan sistem data yang berbeda
- *Query* yang tersebar di setiap sumber data
- Sifat heterogen skema data
- Penentuan lokasi data yang relevan
- Interoperabilitas terkait sumber data yang memiliki akses terbatas, seperti karena sebab keamanan, privasi, dan implikasi performa sumber data

Target yang menjadi harapan bahwa konsep integrasi data dapat sepenuhnya ditangani dengan menggunakan teknik AI (*Artificial Intelligent*) sehingga solusi dengan integrasi data dapat bersifat otomatis, atau setidaknya dapat mengurangi permasalahan kompleksitas konfigurasi aplikasi untuk pengintegrasian data [16].

Terdapat tiga jenis integrasi data [17], yaitu:

1. *Data consolidation*; pengumpulan dan penggabungan serta penyatuan data dari berbagai sumber menjadi satu penampung data (*data store*)
2. *Data propagation*; pemindahan data dari satu atau beberapa sumber data ke lokasi yang lebih mudah untuk diakses berdasarkan aturan pemindahan.
3. *Data federation/virtualization*; pengumpulan data dari berbagai sumber atau database yang berbeda-berbeda tanpa pemindahan atau penduplikasian data sama sekali.

Secara umum integrasi data memiliki tiga tahap sebagaimana yang diilustrasikan pada Gambar 2.4. Tahap di mana proses *entity resolution* dilakukan adalah pada tahap *daat fusion* atau penggabungan data.



Gambar 2.4: Tiga Tahap Umum pada Integrasi Data

Deteksi duplikasi data (*conflict – uncertainties & contradictions*) dilakukan sebelum proses *data fusion*, sehingga proses resolusi konflik atau *entity resolution* adalah proses terakhir yang menentukan hasil akhir data keluaran sebagai global *datasource* yang akan diakses oleh aplikasi [1].

Proses penggabungan data sebagai bagian akhir tahap integrasi data merupakan proses yang harus didampingi oleh tenaga ahli (*expert user*), mengingat kompleksitas proses yang harus dilakukan. Pada tahap ini terdapat beberapa strategi resolusi konflik yang bisa dilakukan [18], yaitu:

- *Consider all possibilities*; seluruh konflik diabaikan dan tiap kombinasi data yang dihasilkan diteruskan ke pengguna untuk diproses secara manual.
- *Trust your friends*; konflik dihindari dengan memilih data dari *datasource* tertentu dan mengabaikan dari *datasource* yang lain.
- *Cry with the wolves*; mengambil nilai data yang sering muncul pada data yang konflik, dengan asumsi bahwa data yang benar adalah data yang sering digunakan.
- *Meet int the middle*; kemungkinan terakhir adalah dengan menyelesaikan konflik dengan membuat nilai data baru dari

seluruh data yang konflik, misalkan mengambil nilai rata-rata untuk konflik data numerik

Terlepas dari berbagai strategi penangan konflik pada penggabungan data, proses resolusi konflik tetap tidak terlepas dari aspek berikut [6]:

- bahwa penentuan prosedur resolusi konflik sangat tergantung dengan domain keahlian dan pemahaman pengguna terhadap kondisi data serta proses yang sangat memakan waktu (*labour-intensive* & *time-consuming*)
- setiap penyatuan *datasource* baru dilakukan, prosedur penggabungan sebelumnya sangat mungkin harus diperbarui, atau bahkan didefinisikan ulang, sehingga resousi konflik cenderung sulit beradaptasi, sedangkan integrasi data bersifat dinamis.
- dari beberapa strategi resolusi konflik, strategi "*Trust your friends*" dan "*Cry with the* wolves" adalah strategi yang paling banyak digunakan, walaupun secara praktis tetap membutuhkan perhatian khusus, terutama untuk menemukan data valid dari data yang terduplikasi (*trustworthy*) yang berasal dari integrasi data dari pihak ketiga, seperti dari data web.

2.2.3 *Entity Resolution*

Entity Resolution (ER) adalah sebuah permasalahan untuk menentukan *record* yang mana di dalam database yang mengacu ke entitas yang sama, serta merupakan langkah yang krusial dan berat di dalam proses *data mining*. ER juga disebut sebagai permasalahan untuk mensarikan (*extracting*), memadankan (*matching*), dan menyelesaikan (*resolving*) keberagaman penyebutan entitas di dalam data yang bersifat terstruktur maupaun yang tidak terstruktur [9].

ER merupakan tugas untuk mengatasi terjadinya ambiguitas pada manifestasi entitas-entitas dunia nyata dalam berbagai record dan berbagai penyebutan berbeda dengan cara penghubungan (*linking*) dan pengelompokan (*grouping*). ER ini merupakan permasalahan yang sudah lama di lingkup pengelolaan database (DBMS), pemanggilan informasi (*information retrieval*),

machine learning, pemrosesan bahasa alami (*natural language processing*), dan statistik.

ER memiliki banyak sekali penyebutan, karena ER belum merupkan konsep yang telah matang dengan teori-teori yang sudah mapan, sehingga ER juga dikenal sebagai: *record linkage*, *duplicate detection*, *conference resolution*, *reference reconciliation*, *fuzzy match*, *object identification*, *object consolidation*, *deduplication*, *approximate match*, *entity clustering*, *dentity uncertainty*, *merge/purge*, *household matching*, *hardening soft databases*, *householding*, *reference matching*, *conflict resolution*, dan *data association* [19]. Terlepas dari semua penyebutan tersebut, ER memiliki potensi aplikatif yang jelas di berbagai bidang, seperti pengelolaan identitas tunggal di pemerintahan, data kesehatan publik, pencarian di web, pembandingan untuk pembelanjaan, penegakan hukum, dan sebagainya.

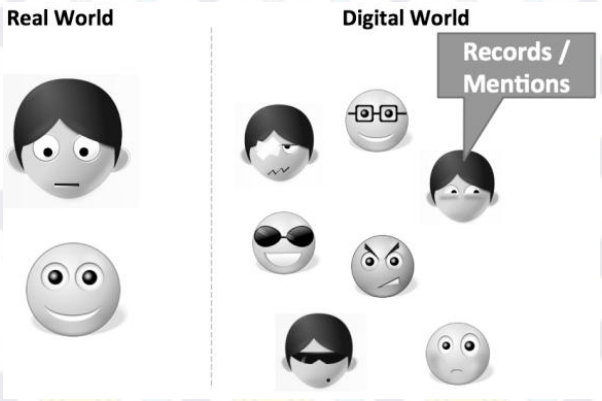
2.2.3.1 Tugas-tugas pada Entity Resolution

Gambar 2.5 memberikan ilustrasi bagaimana pada penyajian digital satu entitas di dunia nyata dapt terduplikasi menjadi berbagai bentuk rekord maupun penyebutan, sehingga diperlukan proses tertentu untuk menyelesaikan permasalahan tersebut, yaitu melalui konsep ER.

ER memiliki beberapa tugas berhubungan dengan penyelesaian persoalan duplikasi penyajian entitas, yaitu:

- *Deduplication*; tugas ini adalah untuk melakukan komputasi penggugusan (*clustering*) pada penyajian-penyajian digital yang mengacu ke entitas yang sama.
- *Record linkage*; sedikit berbeda dengan deduplikasi, tugas ini adalah untuk memadankan rekord-rekord dari satu penyimpanan data yang akan dideduplikasi ke data yang lain. Tugas ini lebih sesuai untuk data yang sudah ternormalisasi, seperti pada database relasional.
- *Reference matching*; tugas ini adalah menentukan dan membersihkan rekord-rekord pengganggu/berlebihan (*noisy records*) di dalam tabel acuan yang akan dideduplikasi, tujuannya

untuk menghindari ambiguitas referensi setelah proses penggabungan data.



Gambar 2.5: Ilustrasi Duplikasi Data yang Memerlukan ER

Jika keterhubungan antara entitas ditetapkan/ditentukan pada tiap tugas-tugas tersebut, maka selanjutnya adalah menangani keterhubungan antara rekord/penyebutan (*mention*). Teknik-teknik ER selayaknya dapat menangani dan persoalan keterhubungan entitas baik di dunia nyata maupun di penyajian digital, mengingat hal itu dapat menentukan signifikansi pengurangan keambiguan pada entitas.

2.2.3.2 Indexing pada Entity Resolution

Salah satu kegiatan yang dilakukan di dalam konsep ER adalah pencocokan atau pembandingan data. Secara potensial, setiap rekord dari satu basis data perlu dibandingkan atau dicocokkan dengan semua rekord yang ada di tabel lain untuk memungkinkan perhitungan kesamaan antara kedu rekord (*record pair*). Kegiatan ini menghasilkan kalkulasi jumlah total pembandingan pasangan rekord yang bersifat kuadratik mengikuti ukuran tabel atau dataset yang akan dibandingkan. Misalnya untuk basis data dengan ukuran hanya 7 rekord saja dapat menghasilkan total 7×7 = 49 pembandingan, yaitu 49 pasangan rekord (*record pair*).

Pembandingan naif pada semua pasangan rekord ini mungkin masih tidak bermasalah untuk database yang kecil, namun untuk database skala menengah-besar (baik database sektor organisasi sektor publik atau swasta) misalkan dengan satu juta rekord, maka akan menghasilkan 1.000.000×1.000.000 =

1.000.000.000.000, yaitu satu triliun *record pair*. Sebagai gambaran, jika dengan sekitar 100.000 *record pair* proses pembandingannya dapat dilakukan selama satu detik (10 µs atau 0,01 ms per-*record pair*), dibutuhkan sekitar 2.777,78 jam, atau hampir 116 hari, untuk proses pembandingan data saja [15].

Untuk mengatasi permasalahan *record pairing* ini, sub topik tersendiri telah menjadi pembahasan penelitian yang memunculkan beberapa konsep yaitu *indexing*, *blocking*, dan *windowing* [9]. Dari ketiga konsep tersebut secara umum memiliki kesamaan yaitu untuk mendapatkan *record pair* yang paling signifikan kemiripannya, sehingga dapat mengurangi biaya komputasi pembandingan kuadratik tersebut.

2.2.3.3 Algoritma pada Entity Resolution

Tinjauan singkat mengenai algoritma yang menjadi dasar *state of the art* pada tahapan ER adalah sebagai berikut [19]:

a. Persiapan Data

Tahapan pertama ini adalah normalisasi skematik dan data, pada bagian ini skematik atribut dipadankan seperti nama atribut `nomor_telepon` dengan `notelp`, termasuk atribut-atribut gabungan juga disesuaikan, misal `kode_wilayah+telp_rumah`.

Tujuan pada persiapan data ini adalah untuk mendapatkan pasangan rekord, sebuah "vektor pembanding" dari skor kesesuaian (*similarity scores*) untuk tiap komponen atribut. Skor kesesuaian ini dapat berupa nilai *boolean* berupa sesuai atau tidak sesuai (*match* atau *not-match*) atau berupa nilai riil sebagai nilai fungsi kedekatan (*distance function*).

b. *Pairwise Matching*

Setelah terbentuk vektor yang dapat diproses dengan pemadanan rekord, maka dikalkulasi kemungkinan kesesuaian rekord-rekord. Terdapat banyak metode untuk menentukan kemungkinan kesesuaian, dua yang termudah adalah dengan menggunakan bobot jumlahan atau rerata pada skore kesesuaian, untuk menentukan ambang (*threshold*).

Metode *Active Learning* dan teknik-teknik *unsupervised/semi-supervised* telah digunakan untuk menangani kesulitan menentukan *training set*, salah satu

yang menggunakan metode *Active Learning* adalah pendekatan *Committee of Classifier* untuk membangun *training set*.

c. *Constraints*

Terdapat beberapa konstrain yang relevan untuk proses ER, sebagai gambaran untuk penyebutan (*mention*) M_i:

1) *Transitivity*; jika M_1 dan M_2 sepadan, M_2 dan M_3 sepadan, maka M_1 dan M_3 seharusnya juga sepadan.
2) *Exclusivity*; jika M_1 sepadan dengan M_2, maka M_3 tidak dapat sepadan dengan M_2.
3) *Functional dependency*; jika M_1 dan M_2 sepadan, maka M_3 dan M_4 haruslah sepadan.

Pada klasifikasi konstrain tersebut juga berlaku penyangkalan, yaitu bukti kebalikan untuk konstrain tertentu.

Berdasarkan konstrain-konstrain tersebut, dapat disimpulkan bahwa *transitivity* adalah kunci untuk proses deduplikasi, *exclusivity* untuk *record linkage*, dan *functional dependency* untuk pembersihan data (*data cleaning*).

Lebih jauh tentang konstrain, seperti *aggregate*, *subsumption*, *neighborhood*, dan lain-lain dapat digunakan untuk konteks domain tertentu.

d. Pendekatan Kolektif

Jika keputusan keanggotaan gugusan tergantung pada gugus yang lain, maka dapat digunakan pendekatan kolektif, pendekatan ini termasuk pendekatan non-probabilistik, seperti sebaran kesamaan (*similarity propagation*), atau model probabilistik termasuk kerangka generatif yang disebut juga pendekatan hibrid.

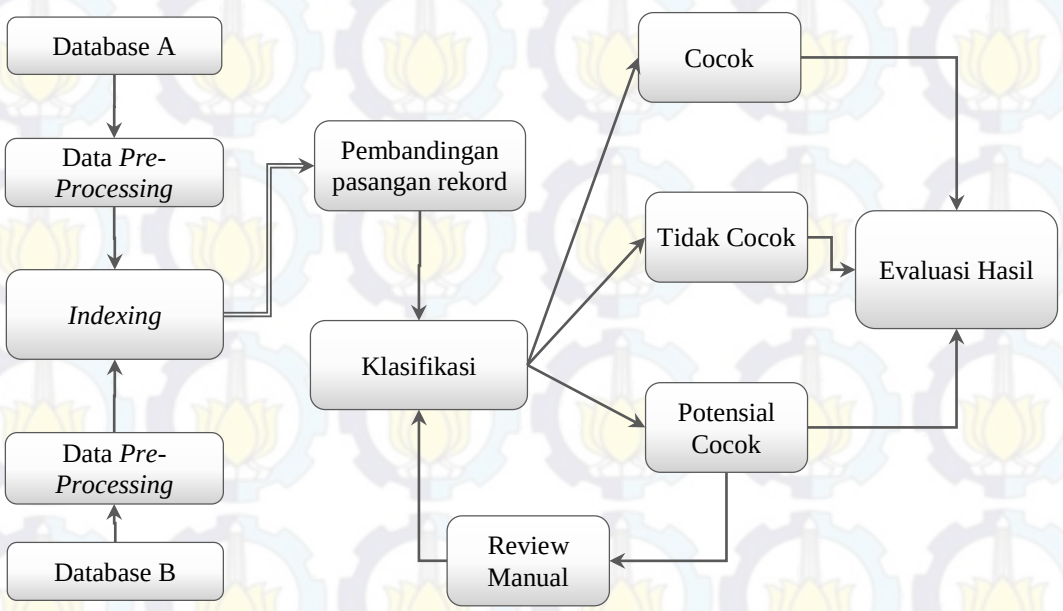
Pendekatan probablistik generatif berdasarkan pada model yang dipandu (*directed*), di mana kebergantungan berpadanan dengan keputusan-keputusan dalam hal generatif, pendekatan ini memliki beberapa variasi yang diantaranya adalah penggunaan Bayesian Networks.

Pendekatan probablistik yang tidak dipandu (*undirected*) menggunakan semantik berdasarkan Markov Networks, dengan kelebihan sintaksis deklaratif menggunakan formula *first-order-logic* untuk membuat konstrain-konstrain, keuntungan yang didapatkan adalah peningkatan kemampuan skalabilitas inferensi

data. Beberapa pendekatan diajukan untuk model ini yaitu Conditional Random Fields, Markov Logic Networks (MLN), dan Probabilistic Soft Logic.

2.2.4 *Data Matching*

Pada tahun 1946, Dunn menggunakan istilah *record linkage* untuk menggambarkan gagasan perakitan sebuah buku catatan kehidupan (*book of life*) untuk setiap individu di dunia. Masing-masing buku ini akan mulai dengan catatan kelahiran dan diakhiri dengan catatan kematian, dan di antaranya akan terdiri dari catatan tentang kontak individu dengan sistem keamanan sosial dan sosial, dan juga termasuk catatan pernikahan dan perceraian. Dunn menyadari bahwa memiliki buku kehidupan seperti itu untuk semua individu dalam suatu populasi akan memberikan banyak informasi yang memungkinkan pemerintah untuk meningkatkan statistik nasional, layanan rencana yang lebih baik dan juga meningkatkan identifikasi individu. Dunn juga mengakui kesulitan menangani data masalah kualitas, seperti nama umum, kesalahan, dan variasi data.



Gambar 2.6: Diagram Umum Alur Proses *Data Matching* [16]

Pada 1950-an dan awal 1960-an, Howard Newcombe dkk. lalu diusulkan penggunaan komputer untuk mengotomatisasi proses pencocokan data. Ia juga sukses mengembangkan ide dasar dari pendekatan *probabilistic record linkage*

(PRL). Dalam pendekatannya, pengodean Soundex fonetik diterapkan ke atribut seperti nama belakang untuk mengatasi variasi nama. Berdasarkan distribusi nilai atribut, bobot kecocokan (*match/non-match* yang juga disebut juga dengan bobot *agreement* dan *disagreement*) dihitung dan digunakan untuk memutuskan apakah dua catatan sesuai dengan kecocokan atau tidak. Berdasarkan gagasan Newcombe, pada tahun 1969 dua ahli statistik Ivan Fellegi dan Alan Sunter menerbitkan makalah seminar mereka tentang PRL. Teori mereka membuktikan bahwa aturan keputusan probabilistik yang optimal dapat ditemukan di bawah asumsi bahwa atribut yang digunakan dalam perbandingan catatan tidak bergantung pada satu sama lain. Karya pionir ini telah menjadi dasar bagi banyak sistem pencocokan data dan produk perangkat lunak, dan masih banyak digunakan saat ini [16].

2.2.5 *Probabilistic Record Linkage* (PRL)

Sebagaimana disebutkan sebelumnya bahwa Fellegi dan Sunter [20] telah menerbitkan formaliasi dari pendekatan *probabilistic record linkage* (PRL) yang sampai sekarang telah banyak digunakan pada perangkat lunak yang khusus untuk menangani ER pada dataset terstruktur. Aturan formal tersebut dipaparkan sebagai berikut:

Jika *M* adalah himpunan pasangan rekord (*record pair*) yang merepresentasikan satu entitas yang sama, dan *U* adalah himpunan pasangan rekord yang mengacu ke entitas berbeda, serta jika terdapat *n* field, maka dua field probabilitas persetujuan (*agreement*) yang disebut dengan m-*probability* dan u-*probability* dapat didefinisikan untuk tiap field ke–*i* dengan $i = 1 \ldots n$ sebagai berikut [3]:

m_i = *P(*field *i* setuju bahwa pasangan rekord adalah sama) = $a_{m,i} / c_{m,i}$

u_i = *P(*field *i* setuju bahwa pasangan rekord tidak sama) = $a_{u,i} / c_{u,i}$

Demikian halnya untuk probabilitas ketidaksetujuan (*disagreement*) dapat didefinisikan sebagai:

m'_i= *P(*field *i* tidak setuju pasangan rekord adalah sama) = $d_{m,i} / c_{m,i}$

u'_i = *P(*field *i* tidak setuju pasangan rekord tidak sama) = $d_{u,i} / c_{u,i}$

dengan diketahui:

$a_{m,i}$ = jumlah pasangan yang sama yang setuju pada field ke-*i*

$d_{m,i}$ = jumlah pasangan yang sama yang tidak setuju pada field ke-*i*

$c_{m,i} = a_{m,i} + d_{m,i}$

$a_{u,i}$ = jumlah pasangan yang tidak sama yang setuju pada field ke-*i*

$d_{u,i}$ = jumlah pasangan yang tidak sama yang tidak setuju pada field ke-*i*

$c_{u,i} = a_{u,i} + d_{u,i}$

sehingga dari literature dapat disimpulkan bahwa $m'_i = (1 - m_i)$ dan $u'_i = (1 - u_i)$. dan nilai $c_{m,i}$ dan $c_{u,i}$ masing-masing bisa kurang dari nilai $|M|$ atau $|U|$ karena sering terdapat rekord yang memiliki nilai field data yang kosong, yang berarti bahwa field ke-*i* yang berisis nilai kosong tersebut tidak menyatakan persetujuan sama sekali. Keadaaan data kosong tersebut biasanya diisikan dengan nilai (*dis*)*agreement* terkecil sebagai nilai bawaan.

Kalkulasi skor pada pasangan rekord dilakukan dengan menambahkan bobot pada tiap field, jika dua rekord setuju bahwa field menyatakan kesamaan, maka bobot persetujuan/*agreement* ditambahkan, dan sebaliknya jika tidak setuju maka boot ketidaksetujuan/*disagreement* ditambahkan. Namun jika satu field bersesuaian dari pasangan rekord tersebut berisi nilai kosong, maka tidak ada penambahan pada nilai bobot (*dis*)*agreement* tersebut.

Kalkulasi bobot dilakukan dengan menggunakan log-*likelihood* yang merupakan nilai proporsi kejadian/kemunculan kondisi bersangkutan, yaitu bobot persetujuan/*agreement* pada field ke-*i* adalah:

$$w_{a,i} = \ln\left(\frac{m_i}{u_i}\right) = \ln(m_i) - \ln(u_i) \quad (2.1)$$

sedangkan bobot ketidaksetujuan/*disagreement* pada field ke-*i* adalah:

$$w_{d,i} = \ln\left(\frac{m'_i}{u'_i}\right) = \ln(m'_i) - \ln(u'_i) \quad (2.2)$$

Keputusan untuk menentukan kesamaan dua pasang rekord adalah dengan membandingkan skore bobot tersebut dengan sebuah nilai ambang (*threshold*) untuk menentukan apakah pasang rekord tersebut sama atau tidak.

2.2.6 *Levenshtein Edit Distance*

Metode *Levenshtein edit distance* adalah metode untuk mendapatkan satuan jarak perbedaan pada pembandingan string dengan melakukan pengukuran

berdasarkan urutan rentetan karakter pada dua string [21]. Secara informal, jarak Levensthein antara dua kata adalah jumlah minimum dari satu perubahan karakter (baik berupa penambahan, penghapusan, dan penggantian) yang dibutuhkan untuk merubah satu kata menjadi kata yang lainnya. Metode ini diberi nama sesuai dengan nama penemunya, yaitu seorang ahli matematika Vladimir Levenshtein pada tahun 1965.

Secara matematis, jarak Levenshtein Antara dua string *a* dan *b*, dengan |*a*| dan |*b*| adalah panjang masing string, diasjikan dalam bentuk $\text{lev}_{a,b}(|a|, |b|)$ adalah:

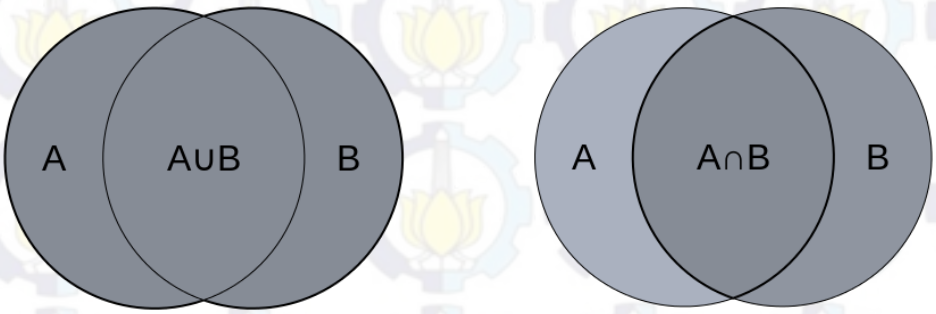
$$\text{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{Jika } \min(i,j) = 0; \\ \min \begin{cases} \text{lev}_{a,b}(i-1,j)+1 \\ \text{lev}_{a,b}(i,j-1)+1 \\ \text{lev}_{a,b}(i-1,j-1)+1_{(a_i \neq b_j)} \end{cases} & \text{Jika tidak;} \end{cases} \quad (2.3)$$

Dimana $1_{(ai \neq bj)}$ adalah fungsi indikasi sama dengan 0 jika $a_i = b_j$, dan sama dengan 1 jika sebaliknya, $\text{lev}_{a,b}(i, j)$ adalah jarak antara karakter ke-*i* dari string *a* dengan karakter ke-*j* pada string *b*.

Normalisasi pada nilai jarak Levenshtein ini dapat dilakukan dengan membagi hasil pengukuran dengan string terpanjang (yang merupakan nilai maksimum hasil pembandingan), sehingga dihasilkan angka antara 0 dan 1.

2.2.7 *Jaccard Coefficient*

Jaccard index dikenal juga sebagai irisan tarhadap gabungan himpunan (*intersection over union*) atau *Jaccard similarity coefficient*, adalah nilai statistic yang digunakan untuk perbandingan kemiripan dan perbedaan pada himpunan sampel [22].



Gambar 2.7: Ilustrasi Irisan dan Gabungan pada Himpunan Sampel A dan B

Metode ini mengukur kemiripan antara himpunan sampel terhingga, yang didefinisikan dengan besaran irisan dibagi dengan besaran gabungan dari himpunan sampel:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}. \tag{2.4}$$

Jika himpunan *A* dan *B* keduanya kosong, maka *J*(*A*,*B*) = 1. Nilai keluaran dari pengukuran ini adalah $0 \leq J(A,B) \leq 1$.

Sedangkan nilai *Jaccard distance* yang digunakan ntuk mengukur perbedaan antara dua himpunan sampel adalah kebalikan dari nilai *Jaccard coefficient,* nilai ini dapat diperoleh dengan mengurangi hasil nilai *Jaccard coefficient* dari 1, atau dengan membagi perbedaan besaran gabungan dan irisan dari dua himpunan dengan besaran gabungannya:

$$d_J(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}. \tag{2.5}$$

2.2.8 Matching Dependencies

Matching dependencies (MDs) di definisikan sebagai aturan-aturan untuk pemeriksaan kesesuaian rekord-rekord. MDs merupakan formalisasi aturan yang ditujukan untuk *entity resolustion* khususnya untuk *data cleansing* pada dataset terstruktur yang menggunakan pendekatan deterministik [7].

MDs ini melakukan formalisasi terhadap empat hal, yaitu operator kemiripan (*similarity operator*), daftar atribut yang bisa dibandingkan (*comparable list*), kebergantungan pemeriksaan kesesuaian (*mathing dependency*), dan kandidat kunci relatif (*relative candidate keys* – RCKs*).*

Similarity operator lebih menekankan aturan dan sifat dari operator kemiripan yang digunakan pada operasi pemeriksaan kesesuaian, seperti sifat refleksif, simetris, *subsum equality*, dan sifat transifitas parsial.

Sedangkan *comparable list*, adalah penentuan batasan/*constraint* pada atribut/field pada dua skematik tabel yang dapat dibandingkan, terutama pada domain data dan panjang data.

Matching dependency (MD) adalah formalisasi sintaksis implikatif dari kebergantungan kesesuaian dari atribut, MD ini dinotasikan dengan φ dari dua relasi skematik (R_1, R_2) sebagai:

$$\bigvee_{j\in[1,k]}\left(R_1[X_1[j]] \approx_j R_2[X_2[j]]\right) \rightarrow R_1[Z_1] \rightleftarrows R_2[Z_2] \quad (2.6)$$

dimana (X_1, X_2) (sedemikian berurut (Z_1, Z_2)) adalah daftar atribut yang dapat dibandingkan pada (R_1, R_2), dan untuk tiap $j \in [1, k]$, sedangkan $\approx_j$ adalah operator kemiripan, dan $k = |X_1|$. Bagian kiri dari notasi $\rightarrow$ disebut sebagai *left-hand-side* (LHS), dan sebelah kanannya adalah *right-hand-side* (RHS) pada MD φ.

MD φ ini menyatakan bahwa jika $R_1[X_1]$ dan $R_2[X_2]$ dinyatakan mirip dengan mempertimbangkan nilai metrik kemiripan tertentu, maka dapat didentifikasi $R_1[Z_1]$ dan $R_2[Z_2]$.

Relative candidate keys (RCKs) di definisikan untuk memutuskan apakah dua representasi rekord mengarah ke satu entitas di dunia nyata. Sebuah kunci Ψ relative pada daftar atriut (Y_1, Y_2) pada (R_1, R_2) adalah sebuah MD dimana RHS ditetapkan sebagai (Y_1, Y_2) yang mana $k = |X_1| = |X_2|$.kunci Ψ ini ditulis dalam bentuk notasi $(X_1, X_2 \parallel C)$ dimana (Y_1, Y_2) adalah bebas dari konteks, dan C adalah operator kemiripan $[\approx_1, \ldots, \approx_k]$, sedangkan k merujuk pada panjang dari Ψdengan C sebagai vektor pembandingnya.

2.2.9 Markov Random Field

Markov Networks atau Markov Random Fields (MRF) dikenal juga dengan sebutan *undirected graphical model* (UGM), adalah himpunan variabel yang memiliki property Markov yang dideskripsikan dalam bentuk *undirected graph*, atau dapat disebutkan bahwa sebuah *random field* disebut sebagai MRF jika memenuhi property-property Markov.

MRF mirip dengan Bayesian Network yang sama membahas penyajian dari keterhubungan/kebergantungan (*dependencies*), perbedaan utama MRF dengan Bayesian adalah bahwa Bayesian bersifat *directed* dan *acyclic*, sedangkan MRF bersifat *undirected* dan bersifat *cyclic*. Sehingga MRF dapat menyajikan keterhubungan yang lebih spesifik yang tidak dapat dilakukan oleh Bayesian, seperti kebergantungan sirkular (*cyclic dependencies*).

Jika densitas probabilistik *join* pada variabel random itu cenderung positif, maka dapat disebut dengan Gibbs Random Fields (GRF), karena menurut teorema Hammersley-Clifford, hal tersebut dapat disajikan menggunakan pengukuran Gibbs dengan menggunakan fungsi energi yang sesuai [23].

Pada domain *Artificial Intelligent* (AI), MRF digunakan untuk memodelkan berbagai tugas-tugas level bawah dan menengah di dalam topik pembahasan *image processing* dan *computer vision*.

2.2.9.1 Definisi Markov Networks

Jika diberikan sebuah *undirected graph G = (V,E),* dan sebuah himpunan variabel random $X = (X_v)_{v \in V}$ yang terindeks dengan *V* dari MRF yang diketahui sebagai *G* jika memenuhi properti lokal Markov sebagai berikut:

- *Pairwise Markov*; adalah tiap dua variabel berdekatan yang secara kondisional saling bebas diantara semua variabel:

$$X_u \perp\!\!\!\perp X_v \mid X_{V \setminus \{u,v\}} \quad \text{dengan} \quad \{u, v\} \notin E \tag{2.7}$$

- *Local Markov*; sebuah variabel yang secara kondisional bebas di antara variabel-variabel yang lain yang di antara tetangga-tetangganya:

$$X_v \perp\!\!\!\perp X_{V \setminus \mathrm{N}[v]} \mid X_{\mathrm{N}(v)} \tag{2.8}$$

  dengan N(*v*) adalah himpunan tetangga dari *v*,

  dan N[*v*] = *v* ∪ N(*v*) dan tetangga terdekat dari *v*.

- *Global Markov*; tiap subset dari variabel yang secara kondisional bebas yang ditentukan dari subset terpisah:

$$X_A \perp\!\!\!\perp X_B \mid X_S \tag{2.9}$$

  di mana tiap path dari node di dalam *A* ke node di dalam *B* melewati *S*.

Ketiga property Markov tersebut tidak setara, dalam arti bahwa *Global Markov* berpengaruh lebih kuat dari pada *Local Markov*, dan selanjutnya lebih kuat dari property *Pairwise Markov*.

2.2.9.2 Faktorisasi Clique

Oleh karena property Markov pada distribusi probabilitas tak tentu (*arbitrary*) cukup sulit untuk dilakukan, maka kelompok yang MRF yang dapat difaktoriasi adalah dengan melihat "*clique*" dari sebuah graf.

Diberikan himpunan variabel random $X = (X_v)_{v \in V}$, dan $P(X=x)$ sebagai probabilitas dari bagian *field* konfigurasi x di dalam X. Di mana $P(X=x)$ adalah probabilitas untuk menemukan variabel random X yang diambil dari bagian nilai x. oleh karena X adalah sebuah himpunan, probabilitas dari x harus dapat dilakukan dengan mengambil *joint distribution* pada X_v.

Jika densitas *joint* ini dapat difaktorisasi pada *clique G*, maka:

$$P(X = x) = \prod_{C \in \mathrm{cl}(G)} \phi_C(x_C) \tag{2.10}$$

dengan X adalah bentuk MRF pada G, dengan $cl(G)$ adalah himpunan pada *clique G*. ϕ_C terkadang disebut sebagai potensial faktor atau potensial *clique*.

2.2.9.3 Model Logistik

MRF dapat disajikan dalam bentuk model *log-linear* dengan fitur fungsi f_k yang distribusi *full-joint*-nya dapat ditulis sebagai:

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_k w_k^\top f_k(x_{\{k\}}) \right) \tag{2.11}$$

di mana notasi:

$$w_k^\top f_k(x_{\{k\}}) = \sum_{i=1}^{N_k} w_{k,i} \cdot f_{k,i}(x_{\{k\}}) \tag{2.12}$$

adalah merupakan hasil operasi *product* tiap konfigurasi *field*, dan Z adalah fungsi partisi:

$$Z = \sum_{x \in \mathcal{X}} \exp\left(\sum_k w_k^\top f_k(x_{\{k\}}) \right) \tag{2.13}$$

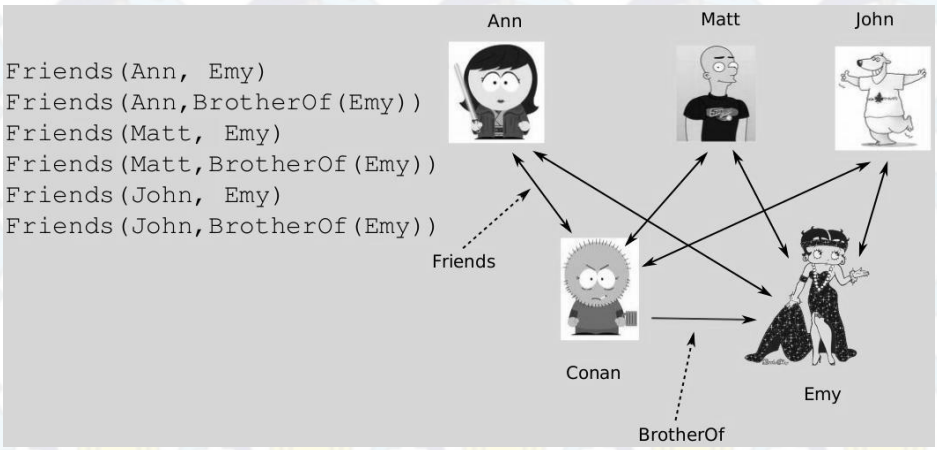
X adalah himpunan tiap seluruh kemungkinan nilai pada *network* variabel random. Biasanya fungsi fitur $f_{k,i}$ didefinisikan karena menjadi indikator dari konfigurasi *clique*, misal $f_{k,i}(x_{\{k\}}) = 1$ jika $x_{\{k\}}$ berkorespondensi ke kemungkinan konfigurasi ke-k dari *clique* ke-i atau $f_{k,i}(x_{\{k\}}) = 0$ jika sebaliknya.

2.2.10 Markov Logic Networks

Markov Logic Networks (MLN) adalah pendekatan sederhana yang menyatukan penyajian kombinasi antara metode *first-order-logic* dengan model

grafikal probabilistik, dan merupakan perluasan probabilistik pada *first-order-logic* untuk memodelkan data relasional [24].

Dalam MLN setiap formula memiliki keterkaitan bobot yang menunjukkan seberapa kuat batasan sebuah aturan, semakin tinggi bobot, maka semakin tinggi pula log probabilitas antara suatu pembahasan yang memenuhi formula tersebut dari pada yang tidak memenuhi, dan sebaliknya. Dengan mengikuti pengertian ini, MLN memperhalus batasan pada *first-order-logic*. Sehingga dapat dipaparkan bahwa jika suatu pembahasan menyalahi formula tersebut, maka probabilitasnya rendah, akan tetapi tidak sampai ke taraf mustahil.



Gambar 2.8: Ilustrasi Model *Undirected* Grafik dalam Penyajian MLN

Definisi MLN, yang jika dinotasikan sebagai L, adalah sebuah pasangan himpunan $\{(F_i, w_i)\}_{i=1}^{m}$, di mana F_i adalah formula dalam *first-logic*, dan bilangan real w_i adalah bobot dari formula tersebut. MLN L beserta himpunan konstanta terbatas $C = \{C_1, C_2, \cdots, C_{|C|}\}$ membangun sebuah Markov Random Field (MRF) $M_{L,C}$ sebagai berikut:

1) $M_{L,C}$ memuat satu *node* biner untuk tiap kemungkinan *grounding* (landasan) untuk tiap predikat yang ada di dalam L.
2) $M_{L,C}$ memuat satu fitur untuk tiap kemungkinan *grounding* pada tiap formla F_i di dalam L. Nilai dari fitur adalah 1 jika *ground* formula bernilai *true*, dan 0 jika sebaliknya. Nilai bobot dari fitur yaitu w_i dikaitkan dengan F_i di dalam L.

Sehingga MLN dapat digambarkan sebagai sebuah *template* untuk membangun MRF. Probabilitas pada *state x* di dalam sebuah MLN didapatkan dari:

$$P(X=x)=\frac{1}{Z}\exp\left(\sum_i w_i n_i(x)\right)=\frac{1}{Z}\prod_i \phi_i\left(x_{\{i\}}\right)^{n_i(x)} \quad (2.14)$$

dengan Z adalah faktor normalisasi untuk membantu penskalaan nilai dari $P(X=x)$ untuk berada di interval [0,1], $n_i(x)$ adalah jumlahan *grounding* yang bernilai *true* pada F_i pada x, dan $x_{\{i\}}$ adalah *state* dari *atom* yang terdapat pada F_i, dan $\phi_i(x_{\{i\}}) = e^{wi}$, dengan w_i adalah bobot dari formulai ke-i.

Persamaan (8) mendefinisikan model generatif pada MLN, yaitu mendefinisikan probabilitas *joint* untuk keseluruhan predikat. Pada *entity resolution* menggunakan pendekatan *discrimative* MLN yang memiliki kelebihan untuk menggabungkan beberapa fitur tak tentu, dan menunjukkan potensi yang baik sebagaimana perbandingan pada kasus model generatif.

Pada *entity resolution*, predikat-predikat dibagi menjadi dua himpunan, yaitu bukti predikat X dan *query* predikat Q. Ambil satu instan x, maka *discrimative* MLN mendefinisikan kondisi distribusi sebagai berikut:

$$P(q \mid x)=\frac{1}{Z_x(w)}\exp\left(\sum_{i\in F_Q}\sum_{j\in G_i} w_i g_j(q,x)\right) \quad (2.15)$$

dengan $Z_x(w)$ adalah faktor normalisasi, F_Q adalah himpunan formula dengan setidaknya terdapat satu *grounding* yang ikut dalam *query* predikat, dan G_i adalah himpunan *ground* pada formula ke-i pada formula *first-order*. $g_i(q,x)$ adalah fungsi biner yang sama dengan 1 jika formula *ground* ke-j bernilai *true*, dan 0 jika sebaliknya.

Pada permasalahan *conflict resolution*, data digunakan untuk memeriksa kebenaran pada fakta yang konflik, dan mengidentifkasi nilai yang benar sesuai dengan kondisi dunia nyata. Sehingga pada model MLN, hanya dibutuhkan satu prediktor, misalkan *IsAccurate(fact)*, yang menjelaskan akurasi dari fakta.

2.2.11 Uji Validasi

Uji validasi bertujuan untuk menemukan parameter terbaik dari suatu *rule*/model yang dilakukan dengan cara menguji besarnya *error* pada data *testing*. Terdapat beberapa metode validasi yang dapat digunakan untuk menemukan *rule*/model terbaik pada proses klasifikasi.

2.2.11.1 Metode Holdout

Metode *holdout* merupakan salah satu metode validasi yang digunakan untuk memilih *rule*/model terbaik, yang akan digunakan untuk proses klasifikasi pada data *testing*. Dalam metode ini, data yang diberikan secara acak (*random*) dibagi menjadi dua set (bagian) secara independen, satu set digunakan sebagai data pelatihan (data *training*) dan satu set digunakan sebagai data pengujian (data *testing*). Biasanya, dua-pertiga dari data dialokasikan untuk data *training*, dan sisanya sepertiga dialokasikan untuk data testing. *Training set* digunakan untuk memperoleh *rule* / model dengan tingkat akurasi yang terbaik, sehingga *rule*/model kemudian diperkirakan menggunakan data *testing* untuk memperoleh hasil klasifikasi terbaik.

2.2.11.2 Metode k-Fold Cross Validation

Metode k-*Fold Cross Validation*, merupakan salah satu metode validasi yang bertujuan untuk menemukan *rule*/model terbaik dengan cara menguji besarnya *error* pada data *testing*. Pada metode ini, data dibagi menjadi k sampel dengan ukuran yang sama, kemudian k-1 sampel digunakan sebagai data *training*, sedangkan satu sampel sisanya digunakan sebagai data *testing*. Sebagai contoh, jika ada 10 set data, akan digunakan 10-*fold*, maka 10 set data tersebut akan dibagi menjadi dua bagian, yaitu 9 set digunakan sebagai data *training*, dan 1 set data digunakan sebagai data *testing*, demikian seterusnya sampai kesepuluh set mendapat bagian menjadi data *testing*. Dari hasil percobaan tersebut maka dapat dihitung rata-rata *error*. *Rule* / model yang terbaik akan memiliki rata-rata *error* terkecil.

2.2.12 Uji Evaluasi

Pada *penelitian* klasifikasi, salah satu cara untuk mengukur performansi dari metode klasifikasi yang telah digunakan pada proses klasifikasi, maka hasil

klasifikasi dilakukan uji evaluasi yaitu dengan menggunakan Metode Pendekatan *Confusion Matrix* dan *Receiver Operating Curve* (ROC) [25].

2.2.12.1 Confusion Matrix

Confusion Matrix adalah perangkat untuk mengevaluasi kehandalan metode yang digunakan untuk proses klasifikasi dengan cara mengenali kelas yang diprediksi dan kelas aktual klasifikasi saat ini. TP (*True Positive*) dan TN (*True Negative*) menunjukkan bahwa hasil klasifikasi (data yang dikenali/diprediksi) adalah benar, sedangkan FP (*False Positive*) dan FN (*False Negative*) menunjukkan hasil klasifikasi (data yang dikenali/diprediksi/disimpulkan/inferensi) yang tidak benar, dan N adalah jumlah total klasifikasi yang dilakukan. *Confusion Matrix* direpresentasikan dalam satu tabel dengan ukuran *m*, dimana m ≥ 2.

Nilai *Confusion Matrix* ini pada bagian baris menunjukkan kelas aktual klasifikasi saat ini, sedangkan nilai pada bagian kolom menunjukkan kelas prediksi hasil klasifikasi sebagaimana pada tabel Tabel 2.4. Berdasarkan tabel tersebut dapat diperoleh beberapa pengukuran kualitas hasil klasifikasi, yaitu:

Akurasi / *Accuracy* (ACC) = $\frac{TP + TP}{TP + TN + FP + FN} = \frac{TP + TN}{P + N}$ (2.16)

Presisi / *Precision* / *Positive Predictive Value* (PPV) = $\frac{TP}{TP + FP}$ (2.17)

Sensifitas / *Recall* / *True Positive Rate* (TPR) = $\frac{TP}{TP + FN}$ (2.18)

Spesifitas / *Specifity* / *True Negative Rate* (TNR) = $\frac{TN}{TN + FP}$ (2.19)

Fallaout / *False Positie Rate* (FPR) = $\frac{TN}{TN + FP} = 1 - TNR$ (2.20)

Tabel 2.4: *Confusion Matrix*

Confusion Matrix		Prediksi / Inferensi	
		Positif	Negatif
Aktual	Benar	TP	FN
	Salah	FP	TN

Keterangan:

N = Jumlah Total Klasifikasi

TP (*True Positive*) = Data sampel (kelas aktual) bernilai benar yang mempunyai hasil prediksi klasifikasi bernilai benar

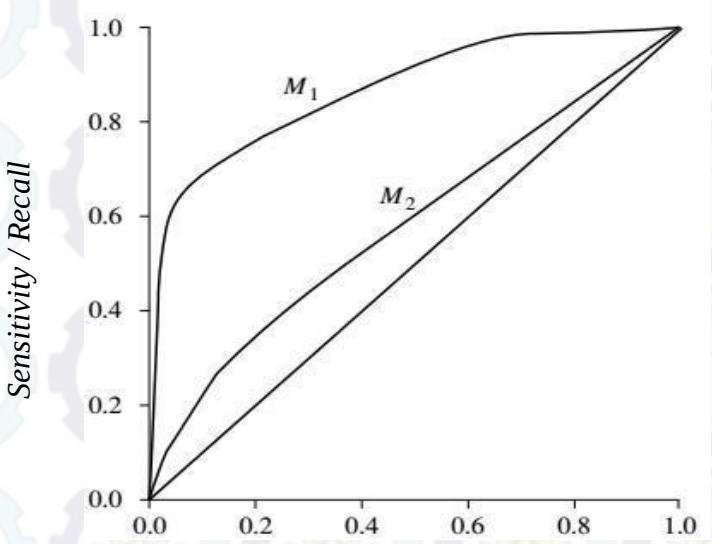
TN (True *Negative*) = Data sampel (kelas aktual) bernilai benar yang mempunyai hasil prediksi klasifikasi bernilai salah

FP (*False Positive*) = Data sampel (kelas aktual) bernilai salah yang mempunyai hasil prediksi klasifikasi bernilai benarra

FN (*True Negative*) = Data sampel (kelas aktual) bernilai salah yang mempunyai hasil prediksi klasifikasi bernilai salah

2.2.12.2 Kurva ROC

Kurva *Receiver Operating Characteristic* (ROC) adalah salah satu teknik visualisasi yang berguna untuk menggambarkan kualitas hasil metode klasifikasi atau merupakan perbandingan dari dua hasil metode klasifikasi. Kurva ROC berasal dari teori deteksi sinyal yang dikembangkan selama Perang Dunia II untuk menganalisis gambar radar. Kurva ROC untuk model tertentu menunjukkan *trade-off* antara rata-rata hasil klasifikasi yang benar atau positif (*True Posotif Rate*/TPR) dan rata-rata hasil klasifikasi yang bernilai salah (*False Positive Rate*/FPR). Grafik ROC menunjukkan fungsi sensitivitas vs (1-spesifisitas) yang berarti bahwa jika nilai sensitivitas rendah, maka spesifisitas yang ditunjukkan oleh pasangannya adalah tinggi, begitu pula sebaliknya. Berdasarkan kurva ROC maka dapat diukur luas daerah di bawah kurva (*Area Under Curve*/ AUC) yang menunjukkan tingkat akurasi model atau metode klasifikasi yang digunakan. Berikut ini adalah interval nilai sebagai dasar penilaian terhadap tingkat akurasi metode klasifikasi yang digunakan semakin luas daerah di bawah kurva (AUC) maka metode klasifikasi yang digunakan semakin baik [25]:



Gambar 2.9: Kurva *Receiver Operating Curve* (ROC)

1. 0.90 - 1.00 = Klasifikasi yang memuaskan (*Excellent Classification*)
2. 0.80 - 0.90 = Klasifikasi yang baik (*Good Classification*)
3. 0.70 - 0.80 = Klasifikasi yang kurang baik (*Fair Classification*)
4. 0.60 - 0.70 = Klasifikasi yang buruk (*Poor Classification*)
5. 0.50 - 0.60 = Klasifikasi yang gagal (*Failure Classification*)

Keterangan :

M_1 = Metode Klasifikasi 1

M_2 = Metode Klasifikasi 2

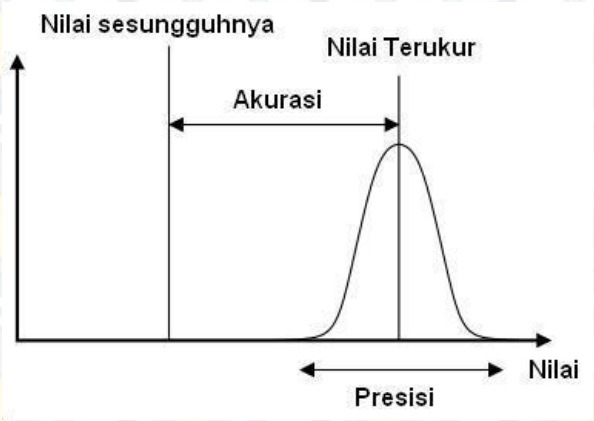
2.2.12.3 Akurasi Sistem

Akurasi dari suatu sistem pengukuran adalah tingkat kedekatan pengukuran kuantitas terhadap nilai yang sebenarnya. Akurasi digunakan pada data mining (misal klasifikasi) untuk mengukur hasil data asli yang bernilai benar (nilai prediksi *klasifikasi*) mendekati hasil data yang memang bernilai benar (nilai klasifikasi aktual / saat ini). Semakin tinggi tingkat akurasi suatu sistem, maka hasil klasifikasi dapat dikatakan semakin baik dan prediktor layak digunakan dalam proses klasifikasi.

2.2.12.4 Akurasi dan Presisi (Precision)

Sebuah sistem pengukuran dapat bernilai akurat dan tepat, atau akurat tetapi tidak tepat, atau tepat tetapi tidak akurat atau tidak tepat dan tidak akurat.

Akurasi menunjukkan kedekatan hasil pengukuran dengan nilai sesungguhnya, sedangkan presisi menunjukkan seberapa dekat perbedaan nilai pada saat dilakukan pengulangan pengukuran.



Gambar 2.10: Grafik Hubungan antara Akurasi dan Presisi

2.2.12.5 F-Measure (F1-Score)

F-measure merupakan salah satu perhitungan evaluasi dalam temu kembali informasi yang mengkombinasikan *recall* dan *precision*. Nilai *recall* dan *precision* pada suatu keadaan dapat memiliki bobot yang berbeda. Ukuran yang menampilkan timbal balik antara *recall* dan *precision* adalah *F-measure* yang merupakan bobot *harmonic mean* dari *recall* dan *precision*. *F-measure* dapat digunakan untuk mengukur kinerja dari *recommendation system* ataupun *information retrival system*.

Karena merupakan rata-rata harmonis dari *precision* dan *recall*, *F-measure* dapat memberikan penilaian kinerja yang lebih seimbang. Range dari nilai *F-Measure* adalah antara 0 dan 1.

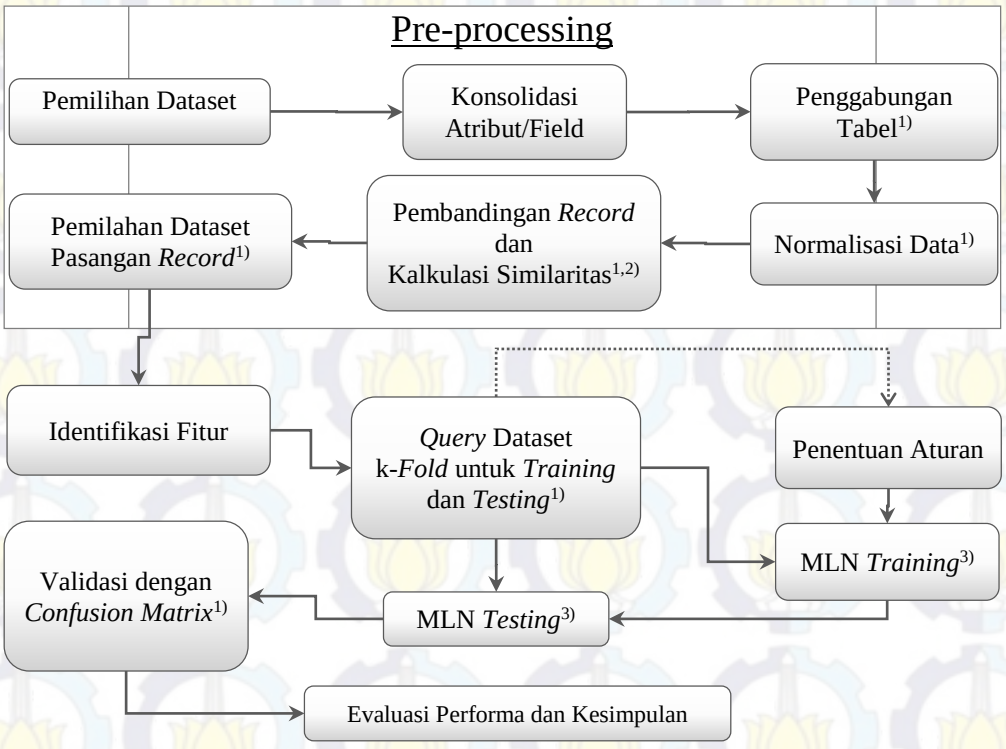
$$F_1\text{-}Score = 2.\frac{PPV.TPR}{PPV + TPR} = \frac{2.TP}{2.TP+FP+FN} \quad (2.21)$$

Halaman ini sengaja dikosongkan

BAB 3
METODOLOGI PENELITIAN

Penelitian ini terdiri dari beberapa tahapan yang menjelaskan bagaimana penelitian dilakukan untuk membukitkan apakah pendekatan yang diajukan cukup bagus untuk menjadi pendekatan alternatif permasalahan *entity resolution* (ER) pada penggabungan database yang cenderung memiliki unreliabilitas yang tinggi.

Tahapan penelitian terdiri dua kelompok, yaitu *pre-processing* meliputi tahapan persiapan data mulai dari pengambilan data yang diteruskan dengan pengolahan awal sampai data siap untuk penentuan fitur. Tahap selanjutnya adalah tahap proses *entity resolution* mulai dari pemilihan fitur-fitur sampai dengan proses training dan inferensi dengan pendekatan probabilitik menggunakan metode MLN.



(Penggunaan Tool: 1) MySQL; 2) Harry *Similarity Measure*; 1) Alchemy MLN)

Gambar 3.1: Tahapan Metodologi Penelitian ER Menggunakan MLN

3.1 *Pre-Processing*

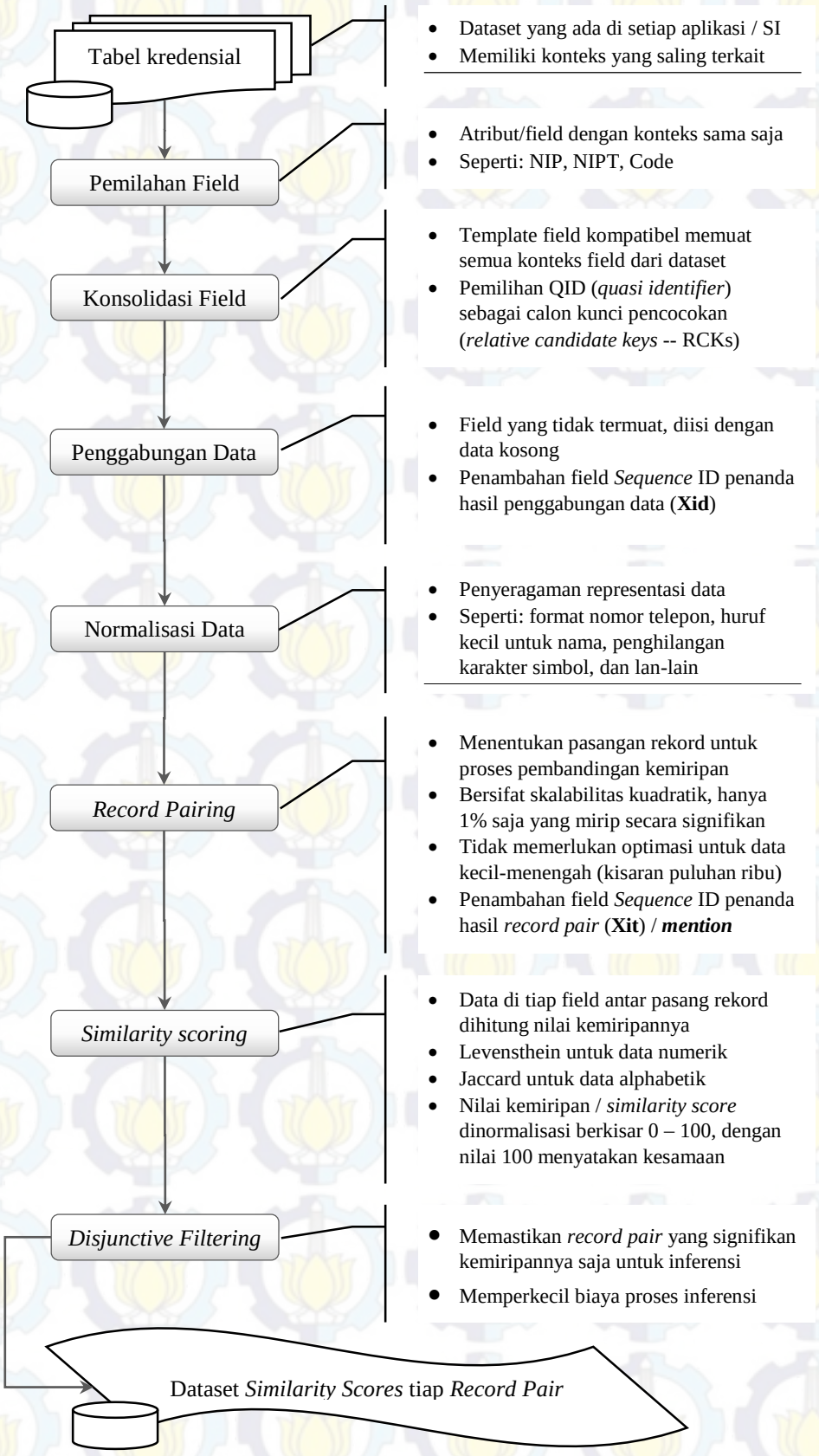
Tahap ini meliputi persiapan pemilihan database sampai dengan pemilahan *record pairs* (pasangan rekord) yang paling signifikan untuk proses lanjutan yaitu pemeriksaan relevansi pada *record pairs* tersebut. Dataset gabungan diambil dari database dari 12 Sistem Informasi sebagai tertera di Tabel 3.1.

Tabel 3.1: Pemilihan Integrasi Sistem Informasi pada UIN Malang

Database	Sistem Informasi	∑ Tabel
Pegawai	Data offline versi data Bagian Kepegawaian	1
e-Journal	Jurnal internal kampus	126
e-Theses	Sistem tugas akhir mahasiswa	182
RadiusDB	Data autentikasi internet kampus	50
Repository	Repository arsip dosen	123
SIAktif	Aplikasi kinerja pegawai	24
SIAkad	Aplikasi akademik kampus	101
SIMaPel	Aplikasi penelitian dan pemberdayaan masyarakat	23
SIMPeg	Aplikasi kepangkatan dan kepegawaian	56
SIPeMas	Aplikasi pengabdian masyarakat	13
SMSBox	Aplikasi SMS broadcast	9
Email	Aplikasi pengelolaan email dan autentikasi	10

Database Pegawai adalah data *offline* yang dimiliki oleh Sub Bagian Kepegawaian yang merupakan data yang ter-*update* (data diambil pada akhir tahun 2017), sehingga penekanan ER pada penelitian ini sebenarnya untuk menghubungkan antara Database Pegawai tersebut ke database yang lain untuk tujuan sinkronisasi data dalam rangka integrasi keseluruhan Sistem Informasi yang ada di lingkungan Akademik UIN Maulana Malik Ibrahim Malang.

Tahapan *pre-processing* ini merupakan rangkaian tahapan yang cukup panjang dan kompleks untuk mempersiapkan dataset agar dapat dilanjutkan untuk proses inferensi, untuk lebih membuka gambaran utuh tentang tahapan ini, diagram alur pada Gambar 3.2 dapat membantu memberikan sedikit penjelasan tentang tahapan proses *pre-processing* ini. Hasil dari tahapan ini adalah data sekunder berupa dataset *similarity scores* tiap *record pair*, dataset ini siap diolah lebih lanjut ke proses penyimpulan, yaitu *training*, *testing*, dan *validation* menggunakan metode inferensi Markov Logic Networks dan metode k-*folds cross-validation*.



Gambar 3.2: Ringkasan Diagram Alur Proses *Pre-Prosesing*

Tabel 3.2: Pemilihan Tabel untuk Proses *Record Linkage*

Database	**Tabel**	**Keterangan**	**∑ *Record***
Pegawai	*Pegawai*	Data *up-to-date* Kepegawaian	901
e-Journal	*Authors*	Penulis jurnal kampus	4.533
	Users	Afiliasi dan reviewer jurnal	5.757
e-Theses	*User*	Penulis dan reviewer tugas akhir	2.427
RadiusDB	*Dosen*	Autentikasi hotspot dosen	1.204
	Staf	Autentikasi hotspot staf	680
	Mahasiswa	Autentikasi hotspot mahasiswa	23.071
Repository	*User*	Pemilik arsip kegaitan akademik	1.546
SIAktif	*Pegawai*	Data pegawai	1.004
SIAkad	*Dosen*	Data akademik dosen	1.194
	Mahasiswa	Data akademik mahasiswa	43.826
SIMaPel	*Ketua*	Data pengampu atau peneliti utama	351
	Anggota	Anggota pengabdian masyarakat atau asisten peneliti	308
SIMPeg	*Pegawai*	Pegawai kampus	482
SIPeMas	*Peserta*	Pengampu dan peserta mahasiswa	3.072
SMSBox	*Kontak*	Kontak pegawai	895
Email	*Pegawai*	Data email dosen dan staf	834
	Mahasiswa	Data email mahasiswa	43.704
Total *record*			**135.789**

Tabel 3.3: Pemilahan Field untuk Konsolidasi Attribut

Database	**Tabel**	**Nama Field**
Pegawai	*Pegawai*	id, kode, nama, _gdep, _nama, _gbel, tipe
e-Journal	*Authors*	author_id, first_name, middle_name, last_name, email
	Users	user_id, first_name, middle_name, last_name, phone, gender, email
e-Theses	*User*	userid, name_given, email
RadiusDB	*Dosen*	sid, nip, gelardepan, nama, gelarbelakang
	Staf	kode, nip, nama
Repository	*User*	userid, name_given, email
SIAktif	*Pegawai*	id, nip, kode, nama
SIAkad	*Dosen*	kode, nip, gelardepan, nama, gelarbelakang
SIMaPel	*Ketua*	id, nip, nama, telp, gender, email
	Anggota	id, nip, nama, telp, gender, email
SMSBox	*Kontak*	id, name, phone
Email	*Pegawai*	id, code, rcode, name, alias, phone, gender, email, umail

3.1.1 Pemilihan Dataset

Dari database yang sudah dipilih untuk diintegrasikan, tidak semua tabel dapat diintegrasikan secara langsung, hanya tabel dengan konteks tertentu saja yang bisa digabung atau dihubungkan/dikaitkan (*linkage*). Pada penelitian ini hanya tabel dengan konteks kredensial saja yang digabungkan karena tabel ini pasti ada pada setiap database Sistem Informasi yang telah di pilih.

Adapun tabel-tabel pilihan tersebut disajikan pada Tabel 3.2 beserta jumlah *record* yang termuat masing-masing.

3.1.2 Konsolidasi Atribut

Konsolidasi atribute adalah proses pencocokan field yang ada pada tiap tabel, ditujukan untuk proses pembandingan pasangan rekord (*record pair*). Pada tahap ini juga ditentukan field yang dapat digunakan untuk membentuk RCKs (*Relative Candidate Keys*) yang menjadi *quasi-identifier* (QID) sebagai alternatif pembandingan kesamaan *record pair*.

Tabel 3.4: Skema Hasil Konsolidasi Tabel untuk Penggabungan Data

Field	Tipe	Keterangan
Xid	*Numeric*	*Sequence* ID untuk acuan *record merging*
Db	*Numeric*	Nomor identifikasi database sumber
Tb	*Numeric*	Nomor identifikasi tabel sumber
Priority*)	*Numeric*	Prioritas berdasarkan kelengkapan field
Id	*Numeric*	*Identifier* yang dimiliki oleh tabel sumber
Sid	*Text*	Kode akademik di Aplikasi Siakad
Code	*Text*	Kode pegawai, NIP
Phone	*Text*	Nomor kontak pegawai
Gender	*Boolean*	Jenis kelamin
Email	*Text*	Email personal pegawai
Umail	*Text*	Email institusional pegawai
Name	*Text*	Nama pegawai
Type	*Boolean*	Tipe pegawai, dosen atau staf

*) Prioritas dataset sebagaimana tertera di Tabel 4.1

3.1.3 Penggabungan Data

Tujuan utama dari *record linkage* adalah untuk pembersihan data (*data cleansing*), yaitu dengan identifikasi duplikasi atau konflik data pada database.

Untuk konteks integrasi database, lebih spesifik penggabungan data pada tabel, ada dua kemungkinan, yaitu bisa berupa *data cleansing* atau hanya berupa *data linking*, di mana pada *data linking* tabel tetap tidak sepenuhnya digabungkan namun hanya berupa penambahan field referensi baru pada tiap tabel yang digabungkan untuk menghubungkan tiap *record* pada tabel-tabel tersebut.

Pada penelitian ini, pengamatan lebih ditekankan ke data linking, di mana bahasan lebih difokuskan ke *record matching* saja, sehingga pada tahap penggabungan data skema ditambah dengan field untuk menyimpan informasi yang dapat digunakan sebagai perunutan ke tabel aslinya, sebagaimana pada Tabel 3.4 yang berisi daftar field skema yang siap untuk proses ke tahapan selanjutnya.

3.1.4 Normalisasi Data

Proses *entity resolution* dapat lebih optimal jika tiap field yang akan diproses diseragamkan terlebih dahulu menjadi satu format yang tertentu, proses penyeragaman ini disebut dengan normalisasi yang dalam konteks dataset pada tabel database meliputi penyeragaman data-data format numerik, tanggal, penyajian nama, dan lain-lain.

Proses normalisasi ini sepenuhnya dilakukan secara manual dengan menggunakan bantuan beberapa *scripting tool*: seperti sed, bash, awk, dan php-cli dan sql *command line*. Proses terinci dari penyeragaman ini tidak dipaparkan secara langsung di sini, mengingat data-data yang digunakan pada penelitian ini adalah data konfidensial internal institusi UIN Maulana Malik Ibrahim Malang. Namun gambaran contoh fragmen normalisasi dipaparkan untuk menggambarkan proses normalisasi yang dilakukan.

3.1.5 Kalkulasi *Similarity*

Sebelum proses kalkulasi kemiripan (*similarity*), idealnya dilakukan satu tahapan yang sebenarnya masuk kategori optimasi untuk memperkecil jumlah pembandingan saat membentuk dataset *record pairs* yang akan dibandingkan. Tahap optimasi tersebut dikenal dengan metode *blocking* dan *windowing* [9], namun karena data sampel setelah proses pemilihan dataset sebelumnya tinggal

menyisakan hanya belasan ribu *record* saja atau tepatnya 17.067, yang merupakan skala kecil-menengah dan masih mungkin untuk dilakukan kalkulasi seluruh *record pair* secara langsung, sehingga dilakukan kalkulasi kemiripan dahulu (menggunakan *tool* harry untuk mengukur kemiripan string [26]), walaupun proses ini tetap memakan waktu sekitar 2 jam di laptop dengan spesifikasi menengah (Intel Core i5 2.2 GHz CPU dan 8GB RAM).

Alasan lain tidak digunakannya metode penyederhanaan skalabilitas dengan *blocking* atau *windowing* [9] adalah karena kedua metode tersebut adalah tetap merupakan metode pendekatan, yang memiliki *error rate* pada penerapannya, sehingga untuk mendapatkan nilai yang lebih eksak, metode tersebut tidak digunakan, disamping juga untuk lebih memfokuskan pembahasan pada pengujian performa dari model yang diajukan di penelitian ini.

Pada tahap ini proses pembentukan *record pair* menghasilkan sekitar 14 juta pasang *record*, yang merupakan pembandingan antara seluruh data *offline* pegawai dengan semua tabel yang sudah dipilih, sehingga jumlah *record pair* data gabungan dengan mudah dapat dihitung sebagai berikut:

$$901\ x\ (17067 - 901) = 901\ x\ 16.166 = \ 14.565.566$$

Karena sebagaimana telah disebutkan sebelumnya, bahwa dataset yang digunakan adalah data untuk menemukan dan menghubungkan (*data linking*) data *offline* kepegawaian ke beberapa database Sistem Informasi.

Tabel 3.5: Contoh Gambaran Kalkulasi Kemiripan pada *Record Pair*

Field	Record 1	Record 2	SimV	Keterangan
Xit	***11001***			ID *record pair* (***mention***)
Xid	*6001*	*8061*		ID *record merging*
Sid	12345	-	0	Hasil kalkulasi kemiripan (*similarity value*)
Code	1234567890	-	0	
Phone	081234567	-	0	
Gender	m	m	100	
Email	danial@y.com	danial@y.com	100	
Umail	-	-	0	
Name	denial hilmi	danial	53	
Type	1	1	100	
Match	-	-	0	Status *Actual Match* pada *record pair*

Tabel 3.5: menunjukkan bagaimana hasil kalkulasi keimiripan (kolom SimV) dilakukan. Untuk field numerik, yaitu **Sid**, **Code**, dan **Phone**, digunakan metode pengukuran *Levenshtein Edit Distance*, sedangkan untuk sisanya digunakan metode *Jaccard Coefficient*. Perbedaan pemilihan metode pengukuran ini dikarenakan metode *Edit Distance* masih memperhatikan urutan sebagai perbedaan, sedangkan metode *Jaccard* yang berbasis himpunan urutan kemunculan karakter tidak dianggap. Nilai kemiripan dinormalisasi agar berada di rentang 0 sampai dengan 100, dengan 0 adalah nilai similaritas jika salah satu atau kedua field adalah kosong (Null).

3.1.6 Pemilahan Dataset

Pada proses kalkulasi similarity sebelumnya, dari 14 juta pasangan rekord (*record pair*) yang didapatkan, hanya sekitar 1% yang berpotensi untuk sama/duplikat [16], dan dari 1% yang signifikan tersebut proses ER dilakukan yang bertujuan untuk menemukan *record pair* benar-benar sesuai menggunakan pendekatan yang diajukan pada penelitian ini.

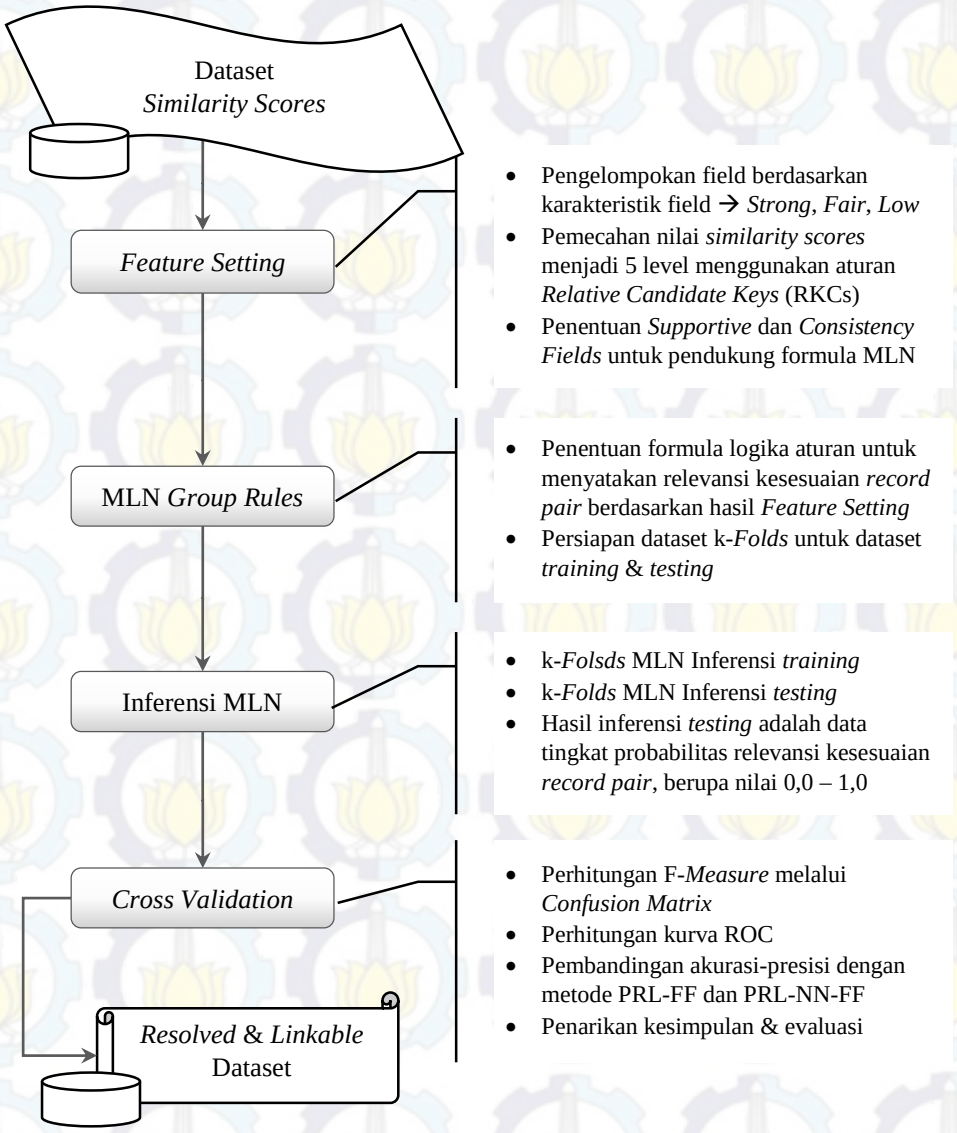
Untuk pendapatkan 1% yang signifikan tersebut, dapat dilakukan dengan filter *disjunctive clause* [27], yaitu dengan menentukan *threshold* δ_i untuk tiap field yang memenuhi $\Delta_s(field_i) \leq \delta_i$. Dengan $\delta_i = 90$, didapatkan sekitar 15 ribu *record pair* (15.735) yang diasumsikan signifikan dan diambil untuk menjadi data sampel, akurasi penggunaan metode filter sederhana ini diabaikan di sini mengingat fokus pada penelitian ini adalah pada pengukuran performa *record matching* yang diajukan.

Dari sekitar 16 ribu *record pair* yang didapatkan tersebut, hanya sekitar 3,5 ribu data benar-benar sesuai (3.505) untuk dilakukan *record linking*, proses penentuan sesuai atau tidak sesuai ini dilakukan secara manual baik dengan *query* SQL maupun dengan mencermati pengurutan hasil *query*, status kesesuaian ini disimpan di field “Match” pada Tabel 3.5:, field ini akan digunakan pada proses validasi untuk pengujian performa pendekatan pada penelitian ini.

3.2 Proses *Entity Resolution*

Proses *ER* merupakan proses yang terdiri dari empat bagian utama, yaitu persiapan dataset untuk *training* dan *testing*, setting fitur dan aturan, proses inferensi, dan validasi. Hasil pada proses ini selanjutnya dianalisa untuk melihat performa dari metode yang diajukan, sebagaimana diagram alur yang ditampilkan pada Gambar 3.3 untuk proses inferensi ini.

Lampiran 3 pada bagian akhir buku tesis ini ditampilkan contoh sampel dari proses inferensi ini, untuk sedikit memberikan gambaran teknis proses inferensi.



Gambar 3.3: Diagram Proses Inferensi, Validasi, dan Evaluasi

3.2.1 Penentuan Fitur

Mengamati PRL yang menggunakan metode machine learning *neural network* yang dicetuskan oleh Wilson [3], fitur yang dibangun adalah menggunakan perhitungan (*dis*)*agreement probabilities* yang dikaitkan ke tiap field pada tabel dengan mengikuti model PRL klasik yang diajukan oleh Fallegi-Sunter [20], yang kemudian nilai tersebut menjadi dasar untuk kalkulasi bobot *log-likelihood* untuk digunakan menjadi penentu keputusan kesesuaian *record pair* berdasarkan *threshold* tertentu.

Penentuan fitur pada PRL dengan *neural network* dikarenakan menggunakan *naïve Bayes classifier*, maka harus mengasumsikan independensi antar fitur yang menjadi variable *perceptron*, sehingga Wilson mengajukan tiap field untuk menjadi basis fitur, yang kemudian tiap fitur dipecah menjadi empat level (*dis*)*agreement* yang disebut sebagai *probabilistic record linkage with full-features* (PRL-FF) sebagai berikut [28]:

Feature f_1: field a_i *agrees very well*
Feature f_2: field a_i *agrees well*
Feature f_3: field a_i *partially agrees*
Feature f_4: field a_i *conflicts*

Pada penelitian ini, fitur tidak ditentukan langsung dari field, akan tetapi ditentukan dari pemecahan kelompok/grup field menjadi lima level dengan tetap mempertahankan sifat mutual ekslusifitasnya. Skor yang digunakan juga bukan dari nilai (*dis*)*agreement*, tapi diambil langsung dari kalkulasi *similarity* sebagaimana disebutkan di bagian 3.1.5.

Tujuan penentuan fitur dengan pengelompokan field ini adalah untuk mengakomodasi konsep relasional database yang memiliki field key berdasarkan karakteristik tertentu (seperti sifat keunikan) untuk menjadi acuan identifikasi entitas, sehingga field-field pada skema database sebenarnya tidak bisa sepenuhnya dianggap mutual eksklusif (dikenal konsep *primary key*, *foreign key*, *unique key*, *indexing*, dan field data biasa), misal secara berurut field *primary key*, *foreign key*, dan *unique key* memiliki berpengaruh berbeda terhadap proses identifikasi dan pencocokan, apa lagi terhadap field-field non-*key*.

Sehingga dengan pengelompokan field (yang diikuti dengan pemecahan level pada kelompok field tersebut) lebih akomodatif terhadap konsep *variabel* pada *machine learning* maupun konsep relasional database, serta lebih adaptif terhadap sifat keterkaitan atribut pada skema database relasional, khususnya untuk dataset dengan unreliabilitas tinggi, karena penambahan fitur pendukung (*supportive)* sebagai pendukung pembuktian pencocokan rekord (*record matching*) yang melibatkan field-field non-*key*.

Fallegi-Sunter [20] pada proses pembobotan mengabaikan *missing data* (data kosong), dan Wilson [3] menganggap *missing data* sebagai konflik, yang mengakibatkan kedua pendekatan tersebut kurang fokus terhadap sifat ureliabilitas data. Sedangkan Ong [4] mengajukan *quasi-identifier* dengan menggunakan alternatif field untuk pertimbangan pencocokan, namun membatasi pengujian hanya dengan *missing rate* pada data yang *unreliable* kurang dari 25% (data kosong dibanding dengan data terisi pada tiap field). Pada penelitian ini, *missing data* digunakan untuk mereview fitur *pendukung* melalui *rule setting* pada formula MLN, sehingga penentuan fitur dengan model ini lebih akomodatif terhadap dataset dengan *missing rate* di atas 25%, khususnya pada kasus penggabungan Data Akademik UIN Maulana Malik Ibrahim.

Tabel 3.6: Pembagian Level Grup Field dalam Aturan Filter RCKs

Fitur	**Level**	**Filter *Relative Candidate Keys* (RCKs) tiap Field**
f_1	1	{ $field_1$; $field_2$; … ‖ =; =; … }
f_2	2	{ [$field_1$; $field_2$; …], $field_1$, $field_2$, … ‖ [≈; ≈; …], !=, !=, … }
f_3	3	{ [$field_1$; $field_2$; …], $field_1$, $field_2$, … ‖ [≠~; ≠~; …], !(≈=), !(≈=), … }
f_4	4	{ [$field_1$; $field_2$; …], $field_1$, $field_2$, … ‖ [≠; ≠; …], !(~≈=), !(~≈=), … }
f_5	5	{ $field_1$, $field_2$, … ‖ Ø, Ø, … }

Simbol	**Deskripsi Notasi**
{ }	Skop filter pada RCKs
[ ]	Prioritas group
( )	Skop operasi
‖	Pemisah operasi
=	$s_f = 100$
≈	$t_m \leq s_f < 100$
~	$t_u \leq s_f < t_m$

Simbol	**Deskripsi Notasi**
≠	$0 < s_f < t_u$
Ø	$s_f = 0$
!	Logikal *NOT*
;	Logikal *OR*
,	Logikal *AND*
…	dan seterusnya

S_f : skor *similarity* dari field pada *record pair*
t_m : *threshold* menyatakan batas kemiripan
t_u : *threshold* menyatakan batas ketidakkemiripan

Pengelompokan field dilakukan berdasarkan karakteristik field sebelum dan sesudah penggabungan data, kelompok field ini dibagi menjadi tiga grup utama, ditambah dua grup pendukung sebagai berikut:

- "*Strong*", untuk field yang bisa menjadi *unique key* pada tabel sebelum penggabungan dilakukan, seperti field NIP dan telepon
- "*Fair*", untuk field yang sangat mungkin untuk *unique*, tapi tidak bisa menjadi *unique key*, karena field tersebut mungkin kosong pada dataset yang tidak handal (*unreliable*), seperti email dan tanggal lahir
- "*Weak*", untuk field yang mungkin sama pada entitas atau konteks/domain yang berbeda, misal nama pegawai dan judul buku
- "*Supportive*", untuk field yang nilainya sangat mungkin untuk sama di tiap record, seperti jenis kelamin, tipe pegawai, tempat lahir, kode pos, dan sebagainya
- "*Consistent*", adalah kondisi khusus digunakan jika grup "*Strong*" kosong atau tidak terisi data (*missing value*), dengan cara mereview *record pair* lain dengan grup "*Weak*" yang mirip dengan *record pair* yang sedang dievaluasi, jika ada kecenderungan konsisten, maka grup "*Consistent*" ini dapat mendukung relevansi kemiripan *record pair*. Konsistensi ini dilihat dari kondisi konflik atau ambigu, misal nama sama tapi NIP berbeda

Threshold t_m dan t_u ditentukan dengan mempertimbangkan rerata nilai dari dataset *training*, pada tiap field pada masing-masing group, dan didapatkan angka perkiraan $t_m = 80$ dan $t_u = 50$.

Tabel 3.6 menampilkan pembagaian level pada grup field "Strong", "Fair", dan "Weak" namun tidak untuk "*Supportive*" dan "*Consistent*" karena grup ini lebih ditekankan sebagai pendukung bukti (*evidence*) pada *rule settings* untuk inferensi. Pembagian level pada grup ini terinspirasi dari RCKs [7] yang dicetuskan

pada model *matching dependencies* (MDs) yang menggunakan aturan filter dari *quasi-identifier* atau *candidate key* yang teridentifikasi.

3.2.2 Penentuan Aturan

Pada PRL yang menggunakan Bayesian sebagai basis algoritma, kelemahan utamanya adalah asumsi bahwa setiap field itu saling tidak berhubungan (*independent*) [3], padahal secara konseptual skematik tabel pada database relasional tidaklah demikian [16]. Pengelompokan field sebagaimana dijelaskan berguna untuk menutupi kelemahan ini, walaupun secara elementer tiap level saling independen, namun grup dan tiap levelnya memiliki pengaruh inferensi berbeda mengikuti aturan logic yang ditetapkan.

Penyajian aturan pada MLN dilakukan dengan menyiapkan logika deklaratif untuk menyatakan relevansi kesesuaian *record pair* dari penentuan fitur menjadi tiga kondisi:

1. *Accurate* (A); aturan dengan klausa tepat/eksak
    a) Grup "Strong" tepat sama (=), maka dinyataka relevan
    b) Grup "Strong" mirip (≈) dan "Weak" sama, juga relevan
    c) Grup "Strong" tidak sama (≠) atau kosong (Ø) sedangkan "Weak" mirip saja, maka dinyatakan tidak relevan
2. *Confident* (C); aturan dengan klausa meyakinkan
    a) Grup "Strong" mirip dengan dukungan bukti *Supportive* atau *Consistent*, maka dinyatakan relevan
    b) Grup "Fair" dan "Weak" tepat sama dan terbukti *Consistent*, juga relevan
3. *Indecisive* (I); aturan dengan klausa bimbang/meragukan
    a) Grup "Strong" kosong, dengan "Weak" tepat sama atau mirip, maka dipertimbangkan
    b) Grup "Fair" tepat sama, dengan "Weak" mirip atau hampir mirip (~), juga dipertimbangkan
    c) Kondisi dipertimbangkan namun didukung *Supportive* atau *Consistent*, maka dianyatakan relevan

3.2.3 Persiapan Data, Training, dan Testing

Terdapat perbedaan penyajian data yang digunakan pada saat *pre-processing* dengan proses inferensi ER, pada *pre-processing* dataset yang digunakan masih berada di dalam lingkup data relasional yang dapat dimanipulasi menggunakan RDBMS biasa, yang pada penelitian ini digunakan database MySQL baik mulai dari proses pemilihan dataset sampai dengan pemilahan dataset, kecuali pada saat kalkulasi *similarity* dimana data dikonversi terlebih dahulu dalam bentuk format teks masukan yang diperlukan oleh *tool* Harry [26], dan data teks keluaran dikonversikan balik ke database untuk proses lanjutan yaitu pemilihan dataset.

Persiapan data ini juga dilakukan *query* dan konversi dari database ke format teks serta sintaksis formula Markov Logic Network yang diperlukan oleh *tool* Alchemy [29], untuk kemudian data teks hasil dikonversi balik juga ke database untuk proses pengujian.

3.2.4 Tahap Uji Validasi dan Evaluasi

Penelitian ini menggunakan validasi hasil ER dengan menggunakan metode *cross validation*, dengan menentukan *k-Fold* menjadi 5 *folding* dan secara acak *record pair* dipilih untuk tetap menjadi bagian dari masing-masing 5 *folding* tersebut, dari 16 ribu data didapatkan sekitar 3 ribuan untuk masing-masing *folding*. Pada proses training, 4 *folding* dijadikan data *training* dan satu sisanya untuk data *testing*, demikian dilakukan lima kali sampai ke tiap *folding* mendapat bagian sebagai data *training* dan *testing*.

Dari proses *cross-validation*, akan didapatkan lima hasil proses *training* dan *testing* yang memuat:

- Jumlah *record pair* yang sesuai/sama (*actual match*) dan tidak sama (*actual unmatch*) menurut pemeriksaan secara manual
- Jumlah prediksi sesuai/*match* dan prediksi tidak sesuai/*unmatch* menurut hasil proses inferensi

Keempat hasil tersebut untuk setiap *folding* proses trainig disajikan dalam *confusion matrix* untuk mendapatkan perhitungan *precision* (konsistensi asumsi), *recall* (kesalahan asumsi), dan *accuracy* (ketepatan asumsi) untuk menilai

performansi dari pendekatan yang diajukan pada penelitian ini. Hasil akhir diambil dengan melakukan rata-rata pada masing-masing *folding* untuk selanjutnya ke tahap analisa dan penyimpulan.

3.2.5 Penarikan Kesimpulan

Probabilisic record linkage merupakan metode yang bersifat perkiraan (*approximation*) sehingga relevansi kesesuaian sebuah *record pair* dinyatakan dalam rentang nilai antara 0.0 sampai dengan 1. Dengan hasil yang demikian, maka sebuah ambang batas (*threshold*) Θ_r ditentukan untuk menjadi acuan dalam memutuskan apakah *record pair* dinyatakan sesuai/sama/duplikat atau tidak. Ambang batas ini juga berada di rentang antara 0 sampai dengan 1, sehingga hasil akhir seperti nilai *F-score* juga dapat berubah sesuai nilai Θ_r [30], sehingga gambaran performa disandarkan dengan rentang hasil *precision*, terutama untuk pembandingan dengan pendekatan sebelumnya.

Halaman ini sengaja dikosongkan

BAB 4
HASIL DAN PEMBAHASAN

Pada Bab ini dijelaskan mengenai proses dan hasil penelitian Penggabungan Database Akademik Berbasis *Entity Resolution* Menggunakan Markov Logic Networks. Dataset sampel yang menjadi obyek penelitian dalam penelitian ini adalah 12 Database Sistem Informasi pada Universitas Islam Negeri Maulana Malik Ibrahim Malang yang dikembangkan secara saling independen sehingga masing-masing Database dikelola secara terpisah.

4.1 *Pre-Processing*

Tahapan ini ditujukan untuk menghasilkan dataset yang siap untuk proses inferensi, yang meliputi pemilihan dataset, konsolidasi skema, penggabungan data pada tabel, normalisasi data, kalkulasi similaritas, dan pemilahan dataset.

4.1.1 Pemilihan Dataset

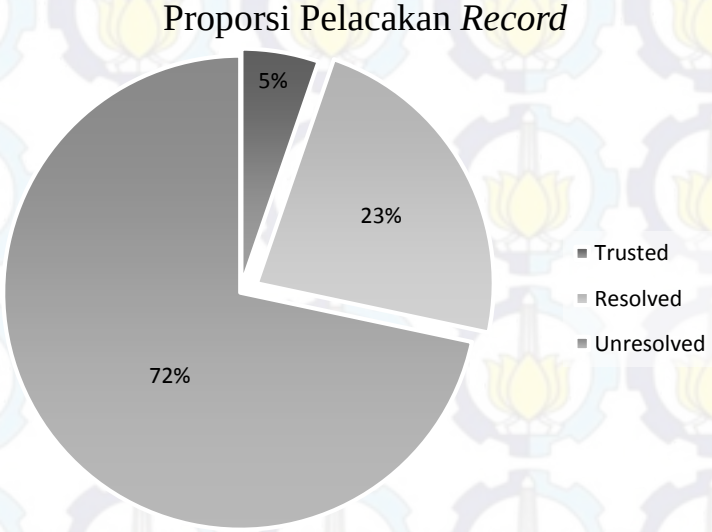
Pada penelitian ini hanya tabel dengan konteks kredensial saja yang digabungkan karena setiap Sistem Informasi dapat dipastikan memiliki informasi pengguna, namun dari keseluruhan tabel dengan konteks kredensial, perlu dipilih lagi tabel-tabel yang akan digabungkan dan kemudian dilakukan proses ER, sebagaimana tertera pada Tabel 4.1 di mana hanya sekitar 17 ribu *record* yang tidak terlacak (*resolve* tidak sampai 100%) dari 135 ribu *record* yang ada, sedangkan tabel yang sudah terlacak 100% penuh dieliminasi karena tidak perlu inferensi untuk *linking*, kecuali tabel pertama yaitu tabel Pegawai yang akan dijadikan sebagai data acuan. Pelacakan *record* ini baik dilakukan melalui *query* menggunakan *primary key* maupun *quasi-identifier* (QID) dari field-field yang ada pada tabel bersangkutan.

Kolom Prioritas pada Tabel 4.1 adalah penomoran untuk digunakan di bagian penentuan fitur (Subbab 3.2.1 Penentuan Fitur), prioritas 3, 2, 1, dan 0 secara berurutan berdasarkan prosentase *resolve* adalah 1-10%, 11-50%, 50-99%, dan prioritas 0 khusus untuk data acuan utama, yaitu data kepegawaian.

Tabel 4.1: Pemilahan Dataset untuk Proses *Record Linkage*

Database	**Tabel**	**Prioritas**	**%*Resolve***	**∑ *Record***
Pegawai	*Pegawai*	0	100	901
e-Journal	*Authors*	3	11	4.035
	Users	3	1	5.700
e-Theses	*User*	3	1	2.403
RadiusDB	*Dosen*	1	57	518
	Staf	2	12	599
	Mahasiswa	-	~~*100*~~	0
Repository	*User*	3	1	1.531
SIAktif	*Pegawai*	1	85	151
SIAkad	*Dosen*	2	30	136
	Mahasiswa	-	~~*100*~~	0
SIMaPel	*Ketua*	1	82	63
	Anggota	2	44	173
SIMPeg	*Pegawai*	-	~~*100*~~	0
SIPeMas	*Peserta*	-	~~*100*~~	0
SMSBox	*Kontak*	2	35	582
Email	*Pegawai*	1	67	275
	Mahasiswa	-	~~*100*~~	0
Total *Record*				**17.067**

Dari sekitar 17 ribu *record* yang didapatkan, tidak semua dapat dilacak *linkage* atau keterhubungannya terhadap data Pegawai (*Trusted* dataset), hanya sekitar 23 % atau sekitar 3,9 ribu saja yang bisa dirunut (*Resolved*) menggunakan SQL dengan menggunakan *field-field primary key* maupun *alternate key*.



Gambar 4.1: Grafik Proporsi Pelacakan *Record* Dataset

Perbandingan proporsi *record* yang bisa terlacak jika menggunakan *query* SQL terangkum pada grafik Gambar 4.1, pada grafik tersebut 72% atau sekitar 12,2 ribu *record* (*Unresolved*) harus dirunut baik secara manual (membandingkan satu-persatu dengan *query* atau *tool* pembantu seperti Microsoft Excel), maupun menggunakan metode *query* SQL tingkat lanjut (seperti *indexing*, *blocking*, *windowing*, dan sebagainya), atau menggunakan konsep *entity resolution* baik pendekatan deterministik, maupun pendekatan probabilistik. Pada penelitian ini, sesuai dengan motivasi yang telah di paparkan pada BAB 1, pendekatan probabilistik digunakan dengan sedikit modifikasi penentuan fitur dari metode sebelumnya, yaitu metode PRL-NN-FF (*probabilistic record linkage, full-featured*, dengan *neural network* menggunakan naïve Bayes classifier [3]).

4.1.2 Konsolidasi Atribut dan Penggabungan Data

Tabel 4.2: Pemilahan Field untuk Konsolidasi Atribut

@No	Database	Tabel	QID	Nama Field Sebenarnya
1	Pegawai	*Pegawai*	NIP, Nama, Tipe	id, kode, nama, _gdep, _nama, _gbel, tipe
2	e-Journal	*Authors*	Email, Nama	author_id, first_name, middle_name, last_name, email
3		*Users*	Email, Nama, Phone, Gender	user_id, first_name, middle_name, last_name, phone, gender, email
4	e-Theses	*User*	Email, Nama	userid, name_given, email
5	RadiusDB	*Dosen*	SID, NIP, Nama	sid, nip, gelardepan, nama, gelarbelakang
6		*Staf*	NIP, Nama	kode, nip, nama
7	Repository	*User*	Email, Nama	userid, name_given, email
8	SIAktif	*Pegawai*	NIP, Nama	id, nip, kode, nama
9	SIAkad	*Dosen*	SID, NIP, Nama	kode, nip, gelardepan, nama, gelarbelakang
10	SIMaPel	*Ketua*	NIP, Email, Nama, HP, Gender	id, nip, nama, telp, gender, email
11		*Anggota*	NIP, Email, Nama, HP, Gender	id, nip, nama, telp, gender, email
12	SMSBox	*Kontak*	Nama, HP	id, name, phone
13	Email	*Pegawai*	NIP, Nama, SID, Email, HP, Gender	id, code, rcode, name, alias, phone, gender, email, umail

Proses konsolidasi sebenarnya dilalukan sepenuhnya secara manual, dengan memeriksa isi field setiap tabel yang digabungkan, setiap field yang memiliki konteks isi yang sama kemudian disatukan dalam satu field bersama.

Misalkan, sebagaiana ditunjukkan pada Tabel 4.2, tabel Pegawai terdapat field "kode" yang menyimpan nomor pegawai (NIP), sedangkan di tabel RadiusDB terdapat field "nip" yang menyimpan nilai yang sama yaitu NIP, maka kedua field tersebut disatukan dalam satu field bernama "Code", sehingga pada saat rekord pada tabel Pegawai dan tabel RadiusDB digabungkan, maka tidak terjadi duplikasi konteks field pada tabel gabungan.

Pada Tabel 4.2 terdapat kolom @No yang untuk acuan yang bersesuaian dengan skema hasil konsolidasi pada Tabel 4.3. Nilai kosong (null) diberikan ke konteks field yang tidak di miliki oleh tabel yang digabungkan, misalkan pada database SMSBox yang tidak memiliki data "Sid".

Tabel 4.3: Nama-nama Field Skema Konsolidasi Atribut yang Bersesuaian

@No	Konsolidasi Atribut yang Bersesuaian							
	Sid	**Code**	**Phone**	**Gender**	**Email**	**Umail**	**_Name**	**Type**
1	id	kode	-	-	-	-	nama	tipe
2	-	-	-	-	email [1)]	email [1)]	*_name [2)]	-
3	-	-	phone	gender	email [1)]	email [1)]	*_name [2)]	-
4	-	-	-	-	email [1)]	email [1)]	name_given	-
5	sid	nip	-	-	-	-	nama	1
6	kode	nip	-	-	-	-	nama	0
7	-	-	-	-	email [1)]	email [1)]	name_given	-
8	-	-	-	-	email [1)]	email [1)]	nama	-
9	kode	nip	-	-	-	-	nama	-
10	-	nip	telp	gender	email [1)]	email [1)]	nama	1
11	-	nip	telp	gender	email [1)]	email [1)]	nama	0
12	-	-	phone	-	-	-	name	-
13	rcode	code	phone	gender	email	umail	name	type

[1)] Untuk email dengan domain institusi diarahkan ke field **Umail**
[2)] Gabungan field-field first_name, middle_name, dan last_name

"Sid" pada adalah Id pada Sistem Akademik, "Code" adalah nomor induk pegawai (NIP), "Email" adalah email pribadi, "Umail" adalah email institusi, dan "Type" adalah tipe pegawai, yang bernilai 1 untuk dosen.

Setelah skema konsolidatif sudah dibuat, maka proses penggabungan data tiap tabel pilihan bisa dilakukan, dengan melakukan *query* INSERT-INTO-

SELECT sederhana [31]. Penggabungan data dengan model skema konsolidatif ini menghasilkan satu tabel dengan tingkat data kosong *missing rate* yang tinggi, sebagaimana ditunjukkan pada grafik Gambar 4.3, yaitu pada grafik balok Sebelum pemilahan dataset dilakukan.

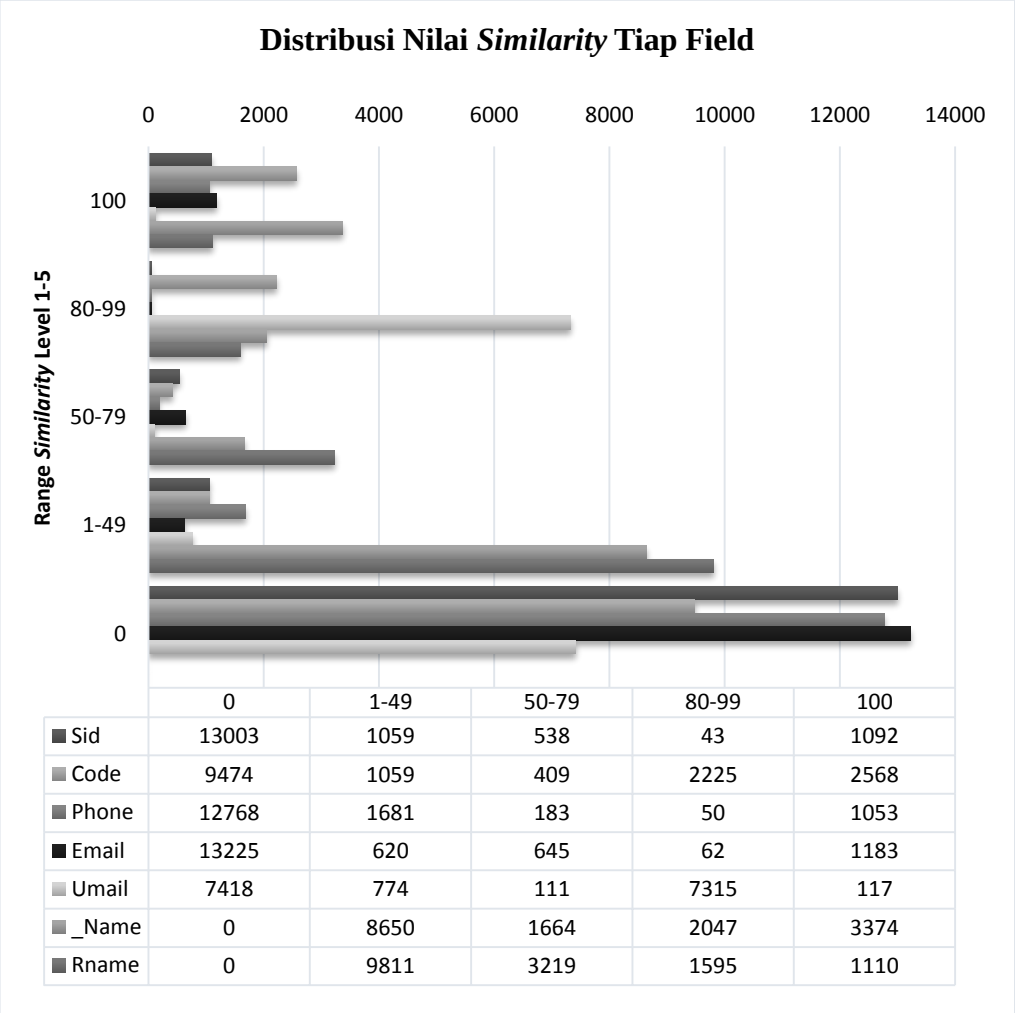
4.1.3 Proses Normalisasi Data

Penyeragaman data sebenarnya adalah satu langkah awal pembersihan data (*data cleansing*) yang memiliki banyak pilihan, diantaranya adalah segmentasi [16] untuk data yang memiliki sub format seperti alamat, namun untuk penelitian ini beberapa langkah ini cukup untuk *pre-processing* pada dataset yang digunakan:

- "Sid" dan "Code" berformat numeric, maka karakter alphabet dan simbol dihapus termasuk spasi, seperti kode NIP 1997-01-111 2233 menjadi 1997011112233
- Field "Phone" juga diperlakukan sama, karena berformat numerik juga, namun ditambah dengan mengganti kode Negara menjadi 0, seperti +621234 menjadi 01234
- Field "Gender" dikonversi menjadi karakter f (*female*) dan m (*male*) atau Null jika kosong, sedangkan "Type" dikonversi menjadi 1 (dosen), 0 (staf), atau Null
- Field "Email" dan "Umail" dibuat menjadi huruf kecil dengan penegasan format email, dan tambahan untuk "Umail**"** yaitu penghilangan nama domain utama, yaitu **.uin-malang.ac.id**
- Dan terakhir adalah memisahkan nama tanpa gelar ke field "Name" dengan nama penuh dengan gelar di field "Rname". Pada field "Name" huruf dikonversi menjadi huruf kecil dan seluruh gelar dihilangkan sehingga menyisakan Nama saja (*common name*), penghilangan simbol selain titik dan spasi, pemisah antar kata nama hanya menjadi satu spasi, dan menghapus spasi di awal dan akhir (*trimming*). Contoh: "Dr. H. Buya Hamka " menjadi "buya hamka" saja

4.1.4 Kalkulasi *Similarity* pada *Record Pair*

Penghitungan jarak kemiripan pada *record pair* dilakukan dengan menggunakan dua metode, yaitu menggunakan Levenshtein Edit Distance untuk field dengan format numerik seperti ”Sid”, ”Code”, dan ”Phone” serta ”Gender” dan ”Type”. Bagi metode pengukuran jarak *Edit Distance*, urutan kemunculan karakter dianggap berbeda, misal nilai ”123” dengan ”321” dianggap berbeda oleh metode pengukuran tersebut, sehingga cocok untuk field-field yang berformat numerik.



	0	1-49	50-79	80-99	100
Sid	13003	1059	538	43	1092
Code	9474	1059	409	2225	2568
Phone	12768	1681	183	50	1053
Email	13225	620	645	62	1183
Umail	7418	774	111	7315	117
_Name	0	8650	1664	2047	3374
Rname	0	9811	3219	1595	1110

Gambar 4.2: Grafik Distribusi Nilai Kemiripan Field

Sedangkan metode kedua yaitu *Jaccard Coefficient*, yang menggunakan prinsip perbandingan himpunan irisan dan gabungan pada konsepsi pengukurannya, yang berakibat pengabaian pada urutan kemunculan, sehingga cocok untuk pengukuran jarak pada field ”Name”, ”Rname”, ”Email”, dan

”Umail”. Pada metode ini nilai ”123” dan ”321” dianggap sama, karena perbandingan irisan dari himpunan karakter keduanya tidak ada.

Grafik pada Gambar 4.2 adalah distribusi nilai hasil pengukuran jarak tiap Field pada *Record Pair*, di mana ditampilkan dalam lima kelompok nilai *Similarity* sesuai pembagian level kemiripan yang akan dibahas pada sub bab 4.2.1 Penentuan Fitur. Grafik tersebut hanya menampilkan *record pair* yang sudah merupakan hasil pemilahan dataset yang akan dijelaskan setelah sub bab ini.

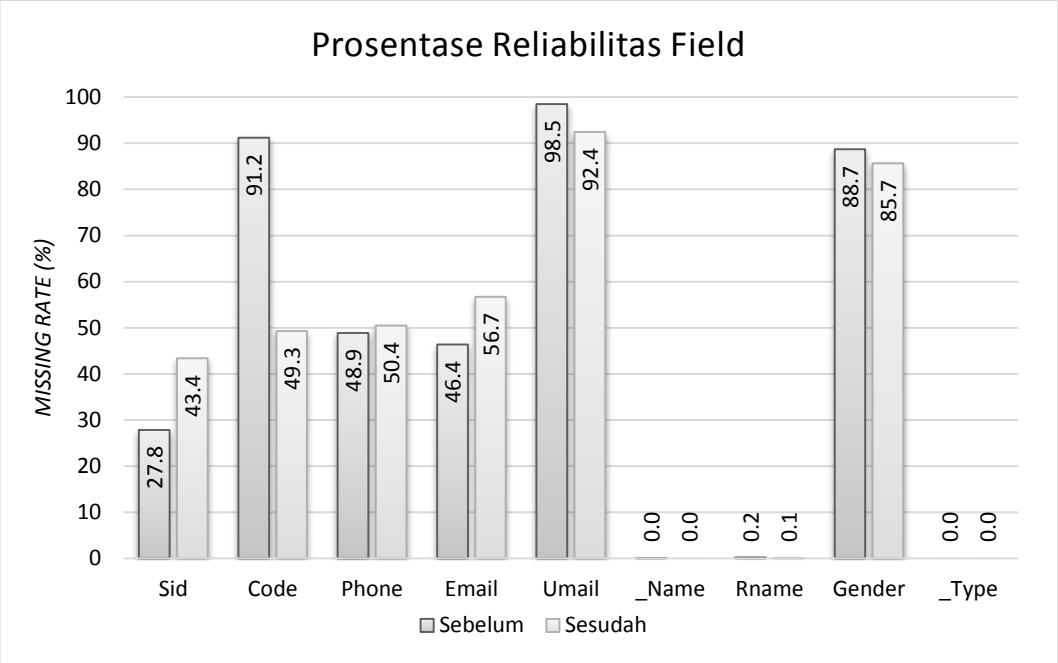
4.1.5 Pemilahan Dataset

Sebagaimana sudah disinggung pada sub bab 3.1.5 Kalkulasi *Similarity* bahwa jumlah *record pair* hasil kalkulasi *similarity* adalah sekitar 15 juta rekord, yang 1% saja dari jumlah tersebut yang benar-benar signifikan untuk diperiksa kemiripannya. Untuk mendapatkan sekira 1% dari *record pair* yang dihasilkan, dapat dilakukan dengan memilahnya melalui *quiry* filter *disjunctive clause* [31] [27], yaitu dengan menentukan satu *threshold* atau lebih sebagai batas nilai *similarity* pada field, jika ada satu saja nilai *similarity* pada field yang memenuhi batas *threshold*, maka *record pair* tersebut diambil sebagai data sampel untuk diproses selanjutnya.

Pada penelitian ini, dipilih dua *threshold* yaitu $\delta_1 = 90$ pada field “Sid”, “Code”, dan Phone”, serta $\delta_2 = 80$ untuk field-field sisanya. Dalam sintaksis *query* adalah sebagai berikut:

```
SELECT *
FROM merged_table
WHERE Sid>δ1 OR Code>δ1 OR Phone>δ1 OR Email>δ2 OR Email>δ2 OR
Umail>δ2 OR Name>δ2 OR Rname>δ2;
```

Hasil dari filter ini tergambar pada grafik Gambar 4.3 yang menampilkan *missing rate* yang berkurang akibat dari filter atau pemilahan dataset, dengan total *record pair* sekitar 15 ribu.



Gambar 4.3: Kondisi Reliabilitas pada Dataset Akademik

4.2 Proses Inferensi

Tahap ini meliputi persiapan fitur *training* mengikuti skema konsolidatif, penentuan aturan *first-order-logic* pada MLN, dan proses *training* serta *testing* pada tiap *k-Fold* untuk mendapatkan hasil berupa data keluaran tingkat *relevansi* tiap *record-pair* yang diproses.

4.2.1 Penentuan Fitur

Sebagaiman penjelasan penentuan fitur pada 3.2.1 Penentuan Fitur, bahwa fitur dikelompokkan dalam grup-grup yang kemudian dipecah menjadi lima level, yang tiap pecahan level dari group tersebut menjadi fitur yang akan digunakan pada proses inferensi.

Pada penentuan fitur, didapatkan field yang masuk kriteria dalam grup-grup tersebut adalah:

- "*Strong*"

  field "sid", "code", dan "phone" masuk ke grup ini karena pada database di Sistem Informasi asal dapat menjadi *unique key* atau *primary key*

- "*Fair*"

  field "email" dan "umail" masuk grup ini karena pada database asal (*source database*) dapat menjadi *unique key* namun sangat mungkin untuk kosong (*missing value*). Field "Phone" sebenarnya bisa masuk grup ini, namun karena di salah satu database asal ada yang menjadi primary key, maka tetap masuk ke group "*Strong*".
- "*Weak*"

  field yang termasuk dalam konteks grup ini adalah field "_name" dan "rname", karena mungkin untuk record berbeda mungkin memiliki nama yang sama
- "*Supportive*"

  field "gender" dan "_type" masuk konteks grup ini, juga jenis pegawai pada tabel *database source*
- "*Consistent*"

  grup ini hanya berupa flag yang berisi nilai dari *query* pada seluruh *record pair* dengan kondisi tiap grup "*Weak*" yang nilai *similarity* tepat sama (=) tapi grup "*Strong*" nilai *similarity* berbeda (≠ atau ~) dan tidak kosong (Ø), daftar contoh pada Tabel 4.5 memperlihatkan hasil *query* yang menampilkan beberapa *record pair* yang potensial ambigu.

Tabel 4.4 adalah fitur-fitur yang dihasilkan dari tahap pengelompokan dan pemecahan level dari daftar konsolidasi field pada tabel Tabel 3.4 di Bab 3.

Fitur *Consistency* menggunakan nilai batas/*threshold* ambiguitas 3 kali untuk menentukan bahwa *record pair* dinyatakan konsisten (tidak ambigu) atau tidak, jadi jika jumlah hasil *query* bahwa *record pair* dengan *normalized name* (Norm_Name) lebih dari *threshold*, maka *record pair* ini potensial untuk ambigu (inkonsisten) sehingga nilai fitur *Consistency* adalah 1/*true*, dan sebaliknya jika kurang dari *threshold*.

Dari *query* ambiguitas pada 15 ribu dataset *record pair*, didapatkan 18 *normalized name* yang dinyatakan potensial inkonsisten sebagaimana pada daftar Tabel 4.6, sehingga untuk tiap *record pair* yang memiliki *normalized name* yang

sama dengan salah satu pada daftar tersebut akan mendapatkan nilai fitur *Consistency* adalah *false* atau 0.

Tabel 4.4: Daftar Fitur Hasil Pengelompokan dan Pemecahan Level

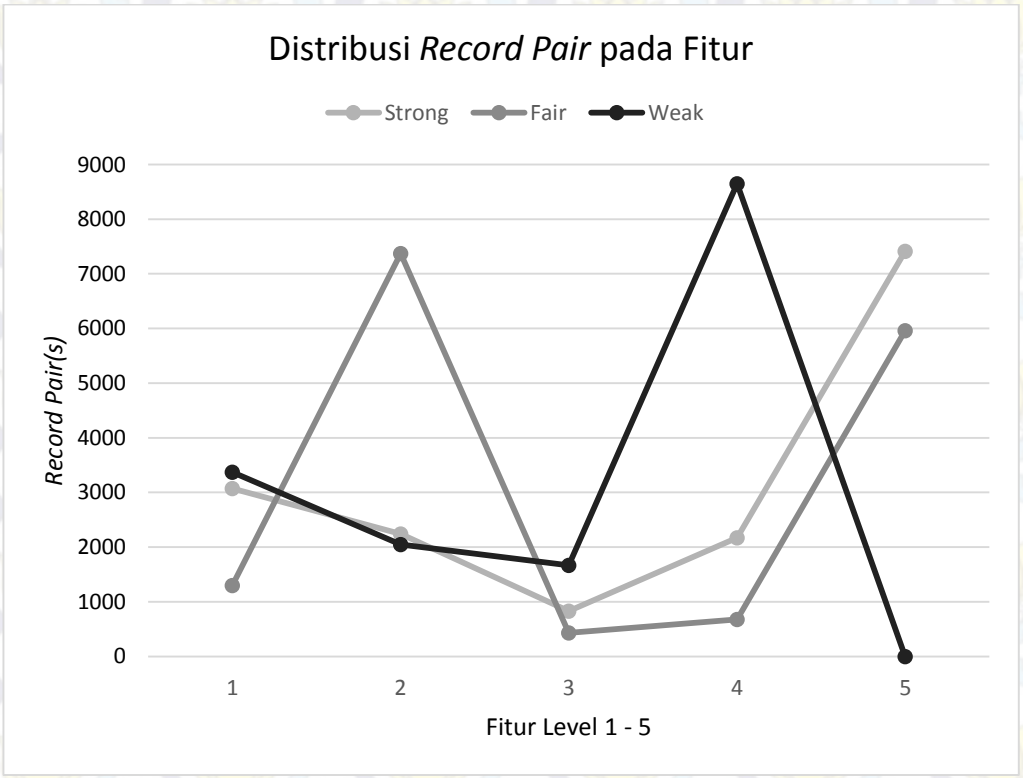
Fitur	**Atribute RCKs**
Strong$_1$	{ sid; code; phone || =; =; = }
Strong$_2$	{ [sid; code; phone], sid, code, phone || [≈; ≈; ≈], !=; !=; != }
Strong$_3$	{ [sid; code; phone], sid, code, phone || [≠~; ≠~; ≠~], !(≈=), !(≈=), !(≈=) }
Strong$_4$	{ [sid; code; phone], sid, code, phone || [≠; ≠; ≠], !(~≈=), !(~≈=), !(~≈=) }
Strong$_5$	{ sid, code, phone || Ø, Ø, Ø }
Fair$_1$	{ email; umail || =; = }
Fair$_2$	{ [email; umail], email, umail || [≈; ≈], !=, != }
Fair$_3$	{ [email; umail], email, umail || [≠~; ≠~], !(≈=), !(≈=) }
Fair$_4$	{ [email; umail], email, umail || [≠; ≠], !(~≈=), !(~≈=) }
Fair$_5$	{ email, umail || Ø, Ø }
Weak$_1$	{ _name; rname || =; = }
Weak$_2$	{ [_name; rname], _name, rname || [≈; ≈], !=, != }
Weak$_3$	{ [_name; rname], _name, rname || [≠~; ≠~], !(≈=), !(≈=) }
Weak$_4$	{ [_name; rname], _name, rname || [≠; ≠], !(~≈=), !(~≈=) }
Weak$_5$	{ _name, rname || Ø, Ø }
Supportive	{ gender || = }, { _type || = }, { _type, DB || =, is_lecturer }
Consistency	Ambiguitas grup *Weak* potensial jika kurang dari 3 kali

Tabel 4.5: Contoh Data yang Ambigu

Nama	**Norm_Name**	**#Jumlah**	**NIP / NIPT**	**Phone**
abdul aziz	abdul_aziz	45	1976********041002	0838****2622
			1972********031002	0812****606
			1969********041004	0813****5269
abdul basid	abdul_basid	11	1965********031003	0815****774
			1982********031001	0856****2844
abdul hakim	abdul_hakim	4	1976********121002	0859****0750
			2012****1211	0
abdul kadir	abdul_kadir	10	1999****1328	0813****0535
			2008****1065	0813****9080

Tabel 4.6: Daftar Nama yang Cenderung Ambigu

No.	#Jumlah	Nama	Normalized Name
1	45	abdul aziz	abdul_aziz
2	11	abdul basid	abdul_basid
3	4	abdul hakim	abdul_hakim
4	10	abdul kadir	abdul_kadir
5	5	abdul rohman	abdul_rohman
6	4	ahmad mahfudzi mafrudlo	ahmad_mahfudzi_mafrudlo
7	4	anton prasetyo	anton_prasetyo
8	4	erna herawati	erna_herawati
9	5	jumriyah	jumriyah
10	12	miftahul huda	miftahul_huda
11	4	mosubthi buchori	mosubthi_buchori
12	6	mulyono	mulyono
13	6	nur faizin	nur_faizin
14	9	sudirman	sudirman
15	4	supriyanto	supriyanto
16	8	sutikno	sutikno
17	5	suyanto	suyanto
18	7	zainul arifin	zainul_arifin



Gambar 4.4: Grafik Hasil *Query* Tiap Level Fitur pada 15 Ribu *Record Pairs*

Setelah menetapkan ketentuan fitur, maka hasil dari penentuan tersebut adalah sebagaimana grafik yang ditampilkan pada Gambar 4.4. Grafik tersebut menampilkan jumlah *record pair* yang mengikuti kriteria penentuan fitur dari Tabel 4.4, daftar pada Tabel 4.7 merupakan angka dari grafik tersebut yang menampilkan total tiap grup fitur yang konsisten dan menunjukkan bahwa tiap fitur pada grup level bersifat mutual eksklusif.

Fitur pendukung pada Tabel 4.7 adalah jumlahan *record pair* pada tiap filter RCK sesuai dengan penentuan pada Tabel 4.4.

Tabel 4.7: Distribusi *Record Pair* Pada Tiap Fitur

Grup	**Level Grup**					**Total**
	1	**2**	**3**	**4**	**5**	
Strong	3.074	2.243	832	2.172	7.414	**15.735**
Fair	1.300	7.369	429	675	5.962	**15.735**
Weak	3.374	2.047	1.664	8.650	0	**15.735**

Fitur Pendukung/*Supportive*			
Fitur	RCK_1	RCK_2	RCK_3
Supportive	3.060	1.827	40
Consistency	15.084	-	-

4.2.2 Penentuan Aturan

Pada metode MLN, inferensi dimulai dengan *training* bobot berdasarkan bukti (*evidence*) dari data *training* pada formula deklaratif logika terurut yang disebut *first-order-logic*. Hasil dari proes *training* tersebut menghasilkan nilai bobot pada formula tersebut, yang kemudian digunakan untuk melakukan inferensi pada data *testing*.

Formula pada MLN merupakan bentuk probabilitas gabungan (*join probability*) dari relasi antar fitur yang sudah ditentukan pada sub bab 4.2.1 yaitu Penentuan Fitur, formula ini dibentuk untuk inferensi relevansi kesesuaian tiap *record pair* pada dataset *testing*.

Hal pertama yang perlu dilakukan adalah menerjemahkan atribut Fitur dalam bentuk sintaksis MLN, dimulai dengan membentuk *naming* kelompok/grup Fitur dan Level serta *Supportive* dalam bentuk konstanta MLN sebagai berikut:

```
group = { Strong, Fair, Weak }
state = { Match, Alike, Close, Skimp, Empty }
flag = { Supportive, Consistent }
```

Konstanta *group* adalah grup fitur yang sudah dipaparkan di sub bab 4.2.1, dan konstanta *state* adalah level dari grup fitur secara berurutan yaitu *Match*, *Alike*, *Close*, *Skimp*, dan *Empty* adalah level 1 sampai dengan 5, sedangkan *flag* adalah konstanta untuk *naming* fitur *Supportive* dan *Consistency*.

Berikutnya adalah deklarasi predikat untuk menyatakan *evidence* yang akan menampung fakta/*mention* dari tiap *record pair* pada dataset *training* maupun *testing*. Predikat *evidence* ini diperlukan karena MLN melakukan inferensi terhadap dataset dalam sintaksis *first-order-logic*, sehingga dataset yang sudah dibentuk menjadi fitur grup dan level pada sub bab 4.2.1 tersebut akan diterjemahkan ke dalam bentuk predikat *evidence* ini untuk proses inferensi.

```
Have(mention, group, state)
With(mention, flag)
Relevant(mention)
Consider(mention)
```

Predikat *Have*() adalah bentuk pernyataan bahwa sebuah *mention* memiliki fitur grup *group* dengan level kemiripan/*similarity* bernilai level *state*. Istilah ”*mention*” adalah sebuah pengenal numerik (ID*entity*) yang mewakili satu *record pair* yang akan menjadi *evidence*, agar data *record pair* hasil inferensi MLN dapat dilacak balik ke tabel dataset gabungan

Predikat *With*() merupakan bentuk pernyataan dukungan bahwa sebuah *mention* diperkuat dengan bukti tambahan berupa fitur *Supportive* dengan konstanta *flag*. Sedangkan predikat *Relevant*() untuk menyatakan relevansi dari *mention*, serta *Consider*() yang menyatakan bahwa relevansi *mention* masih perlu dipertimbangan lebih lanjut mengikuti tersedianya dukungan *evidence* tambahan.

Predikat *Relevant*() adalah predikat yang akan di*query* pada inferensi data *testing*, dan nantiny predikat ini akan memuat bobot antara 0 sampai dengan 1 yang menyatakan tingkat kepastian dari relevansi kesesuaian *mention* pada *record pair*.

Tahap berikutnya sebelum proses inferensi, adalah menyiapkan formula yang merupakan bentuk deklaratif dari logika yang sudah ditentukan pada sub bab 3.2.2 Penentuan Aturan.

Formula pertama adalah aturan *Accurate* (A) yang memiliki tiga klausa, bentuk sintaks dalam sintaks *first-order-logic* MLN adalah:

```
a) Have(m, Strong, Match) => Relevant(m)
b) Have(m, Strong, Alike) ^ Have(m, Weak, Match)
      => Relevant(m)
c) Have(m, Strong, Skimp) v Have(m, Strong, Empty)) ^
      (Have(m, Weak, Close)v
      Have(m, Weak, Skimp)) => !Relevant(m)
```

Formula kedua adalah aturan *Confident* (C) yang memiliki dua klausa, yaitu:

```
a) Have(m, Strong, Alike) ^ (With(m, Supportive) v
      With(m, Consistent)) => Relevant(m)
b) Have(m, Fair, Match) ^ Have(m, Weak, Match) ^
      With(m, Consistent) => Relevant(m)
```

Pada aturan ini, jika fitur *Consistency* dihilangkan, maka aturan akan tinggal satu yaitu:

```
c) Have(m, Strong, Alike) ^ With(m, Supportive)
      => Relevant(m)
```

Begitu pula jika fitur *Supportive* dihilangkan, maka aturan menjadi:

```
d) Have(m, Strong, Alike) ^ With(m, Consistent)
      => Relevant(m)
e) Have(m, Fair, Match) ^ Have(m, Weak, Match) ^
      With(m, Consistent) => Relevant(m)
```

Penghilangan fitur *Consistency* dan *Supportive* ini dimaksudkan untuk melakukan pengujian efektivitas aturan *Confident* (C) terhadap proses inferensi yang akan dijelaskan pada sub bab 4.3.2 Evaluasi Proses Inferensi.

Formula ketiga adalah aturan *Indecisive* (I) yang memiliki tiga klausa:

a) `Have(m, Strong, Empty) ^ (Have(m, Weak, Match) v Have(m, Weak, Alike)) => Consider(m)`

b) `Have(m, Fair, Match) ^ (Have(m, Weak, Alike) v Have(m, Weak, Close)) => Consider(m)`

c) `Consider(m) ^ (With(m, Supportive) v With(m, Consistent)) => Relevant(m)`

Formula-formula tersebut akan di*training* untuk mendapatkan bobot menggunakan *k-fold* dataset yang kemudian digunakan pada inferensi data *testing* pada masing-masing *folding*.

4.2.3 Hasil *Training* dan *Testing*

Setelah menentukan aturan untuk inferensi *record matching*, langkah yang dilakukan sebelum proses inferensi adalah menyajikan data *record* tabel menjadi dataset predikat *evidence*.

Pada tabel Sedangkan untuk data *testing* (pada *i-Fold* di mana *mention* ini berada di *folding* untuk *testing*), predikat `Relevant(X105495401)` tidak disertakan karena justru predikat *Relevant*() ini lah yang akan disimpulkan atau dicari nilainya berdasarkan *evidence* yang tersedia di atasnya. Untuk *mention* yang dari hasil pemeriksaan manual ternyata tidak sama (*Unmatch*), maka predikat *Relevant*() ini tertulis *!*`Relevant(X105495401)` dengan prefix tanda seru (!) yang menyatakan bahwa *evidence* bersangkutan tidak relevan/irrelevant.

Tabel 4.8 ditampilkan sebuah *mention* X105495401 yang memuat data kemiripan (*SimV*) dari sepasang rekord (*record pair*), serta hasil penyajian fitur dari nilai kemiripan tersebut, jika diterjemahkan dalam sintaks predikat *evidence* untuk data *training* menjadi:

```
Have(X105495401,Strong,Alike)
Have(X105495401,Fair,Match)
Have(X105495401,Weak,Close)
With(X105495401,Supp)
With(X105495401,Cons)
Relevant(X105495401)
```

Sedangkan untuk data *testing* (pada *i-Fold* di mana *mention* ini berada di *folding* untuk *testing*), predikat `Relevant(X105495401)` tidak disertakan karena justru predikat *Relevant*() ini lah yang akan disimpulkan atau dicari nilainya berdasarkan *evidence* yang tersedia di atasnya. Untuk *mention* yang dari hasil pemeriksaan manual ternyata tidak sama (*Unmatch*), maka predikat *Relevant*() ini tertulis *!*`Relevant(X105495401)` dengan prefix tanda seru (!) yang menyatakan bahwa *evidence* bersangkutan tidak relevan/irrelevant.

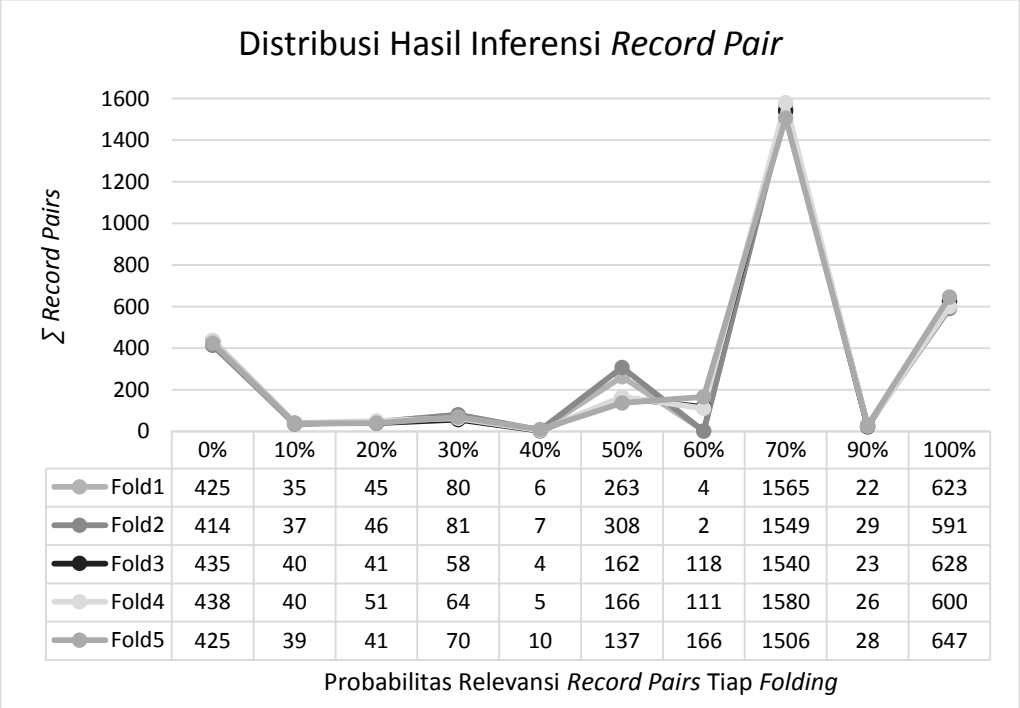
Tabel 4.8: Contoh *Mention* pada *Record Pair*, Nilai *Similarity*, dan Fitur

Field	**Tabel Sumber**		***SimV***	**Fitur**
	Data Pegawai	**Data Email**		
Xit	***105495401***			
Xid	*1458*	*107436*		
Sid	P00039	tabah	15	*Strong*$_2$
Code	1979********041001	1979********041000	95	
Phone	0813****6222	0812****4881	31	
Email	tabah79@gmail.com	tabah79@gmail.com	100	*Fair*$_1$
Umail	NULL	tabah@uin-malang.ac.id	0	
_Name	m. mujtabah	mujtabah	75	*Weak*$_3$
Gender	m	m	100	*Supp*
_Type	2	2	100	
	Didukung dua status *Supp*:			*Supportive*
	Status *Consistency*:			*Consistent*
	Status Relevansi (secara manual):			*Match*
	Mention:			***X105495401***

Hasil kalkulasi pada Tabel 4.9 menunjukkan bobot *log-likelihood* seluruh formula hasil dari proses data *training*. Bobot formula dari tiap *folding* tersebut digunakan untuk inferensi data *testing* yang dibuat atau dihasilkan dengan tanpa menyertakan predikat *Relevant*(). Pada Tabel 4.9 adalah hasil data *training* untuk satu set aturan penuh, adapun pada sub bab Evaluasi Proses Inferensi nanti setiap kelompok aturan diproses terpisah untuk menguji pengaruh tiap aturan, sedangkan nilai bobot hasil *training* secara lengkap dicantumkan terpisah pada bagian lampiran.

Tabel 4.9: Hasil Perhitungan Bobot Formula dari Proses Data *Training*

No.	*Rules*	Bobot Formula Hasil *Training*				
		***Fold* 1**	***Fold* 2**	***Fold* 3**	***Fold* 4**	***Fold* 5**
1	***Accurate* (A)**					
	a)	4.19106	4.24602	4.20829	4.22611	4.18606
	b)	5.66376	5.33179	5.53404	5.51827	5.44346
	c)	4.55928	4.46353	4.40877	4.62307	4.5179
2	***Confident* (C)**					
	a)	-3.93606	-3.86441	-3.82025	-3.90742	-3.94723
	b)	2.68982	2.66332	2.69464	2.66952	2.65122
3	***Indecisive* (I)**					
	a)	-4.94249	-4.9021	-4.75018	-4.59113	-4.75847
	b)	-5.82308	-5.92163	-5.92525	-5.92005	-5.83847
	c)	8.2093	8,38892	8,1341	8.42224	8.38602



Gambar 4.5: Grafik Nilai Relevansi *Record Pair* Hasil Inferensi Data *Testing*

Grafik Gambar 4.5 menampilkan frekuensi kemunculan prosentase relevansi kemiripan *record pair* pada data *testing* untuk setiap *folding*. Sebagaimana yang sudah dipaparkan pada sub bab 3.2.5 Penarikan Kesimpulan, bahwa hasil akhir ditentukan melalui penentuan *threshod* Θ_r yang berupa ambang batas dari prosentase relevansi pada grafik Gambar 4.5, pemilihan nilai ambang Θ_r

ini sangat menentukan pengukuran performansi dari proses inferensi dengan pendekatan pada penelitian ini.

4.3 Tahapan Validasi dan Evaluasi

Pada dasarnya, proses inferensi untuk menentukan kesamaan *record pair* dengan pendekatan probabilistik, baik itu pada model klasik Fallegi-Sunter [20] yaitu Probabilistic *Record Linkage* (PRL) beserta turunannya yang diajukan oleh Wilson [3] dengan menggunakan *Neural Network*, sebenarnya adalah merupakan bentuk klasifikasi *machine learning* yang bersifat biner (melibatkan dua kelas saja), yaitu *match* dan *unmatch*, sehingga proses validasi dapat menggunakan metode *confusion matrix* untuk mengukur performa hasil inferensi, seperti yang sudah dijelaskan sebelumnya pada sub bab 3.2.4 Tahap Uji Validasi dan Evaluasi.

4.3.1 Hasil Inferensi

Pengujian validitas pada penelitian ini adalah dengan menggunakan metode *cross validation* dengan memilih nilai *threshold* Θ_r untuk memutuskan apakah prosentase probabilitas kesesuaian/relevansi pada *record pair* dinyatakan positif sesuai/relevan atau tidak. Nilai *threshold* tersebut diberlakukan untuk validasi setiap *folding*, dan pada Tabel 4.10 didapatkan hasil inferensi dengan penentuan nilai *threshold* Θ_r = 90%, sesuai dengan grafik distribusi pada Gambar 4.5 maka diambil *record pair* yang memenuhi yaitu *record pair* dengan nilai probabilitas relevansi 90% dan 100% untuk setiap *folding*.

Pada hasil inferensi Tabel 4.10, kolom Aktual *Match* dan *Unmatch* adalah jumlah *record pair* yang dinyatakan sesuai/sama yang dilakukan secara manual, Inferensi Positif adalah jumlah *record pair* yang dinyatakan sesuai/sama menggunakan model pendekatan pada penelitian ini, sedangkan Inferensi Negatif adalah yang dinyatakan tidak sesuai.

Kolom ”Mis” menunjukkan jumlah *record pair* yang luput/terlewatkan dari proses inferensi karena set aturan tidak mengakomodasi kondisi set nilai *similarity* pada *record pair*. Pada sub bab 4.3.2 Evaluasi Proses Inferensi nanti akan diulas lebih jauh tentang pengaruh set aturan terhadap jumlah *missed record pair*

(MRP) yaitu *record pair* yang tidak terproses inferensi akibat tidak tercakup pada kombinasi set aturan yang ditentukan. Semua MRP yang dihasilkan dari proses inferensi diasumsikan sebagai kondisi Inferensi Negatif karena MRP ini tidak memenuhi kondisi relevansi yang diharapkan sesuai aturan yang ditentukan, kolom Inferensi-Negatif yang ditunjukkan pada Tabel 4.10 sudah memuat MRP ini.

Tabel 4.10: Hasil Inferensi dan Pemeriksaan Ketepatan Tiap *Folding*

Fd	***Recrd Pair***	**Mis**	**Aktual**		**Inferensi**		**Ketepatan**			
			Match	***Un-match***	**Positif**	**Negatif**	***TP***	***TN***	***FP***	***FN***
1	3.147	79	709	2.438	645	2.502	637	2.430	8	72
2	3.147	83	679	2.468	620	2.527	613	2.461	7	66
3	3.147	98	710	2.437	651	2.496	644	2.430	7	66
4	3.147	66	681	2.466	626	2.521	618	2.458	8	63
5	3.147	77	726	2.421	675	2.472	670	2.416	5	56
T	**15.735**	**403**	**3.505**	**12.230**	**3.217**	**12.518**	**3.182**	**12.195**	**35**	**323**
		Total *Record Pair* :		**15.735**		**15.735**				**15.735**

Nilai TP (*True Positive*) *–Match-Positif–* adalah jumlah hasil Inferensi Positif yang sesuai dengan Aktual *Match,* TN (*True Negative*) *–Unmatch-Negatif–* adalah jumlah hasil Inferensi Negatif yang sesuai dengan Aktual *Unmatch,* FP (*False Positive*) *–Unmatch-Positif–* adalah jumlah kesalahan menyatakan Positif padahal sebenarnya adalah Aktual *Unmatch,* dan FN (*False Negative*) *–Match-Negative–* adalah jumlah kesalahan hasil inferensi menyatakan Negatif yang sebenarnya adalah Aktual *Match*.

Dapat dilihat pada Tabel 4.10 tersebut bahwa total jumlah *Record Pair* pada kolom Aktual, Inferensi, dan pemeriksaan ketepatan sudah sesuai yaitu 15.735 *record pair* yang membuktikan bahwa proses kalkulasi pemeriksaan ketepatan antara jumlah inferensi dan aktual telah akurat.

Selanjutnya adalah mulai melakukan perhitungan *F-Measure* dengan menyajikan nilai total seluruh *folding* dari Tabel 4.10 tersebut ke tabel *confusion matrix* sebagaimana ditampilkan pada Tabel 4.11.

Tabel 4.11: *Confision Matrix* Hasil Inferensi dari Seluruh *Folding*

Confusion Matrix		**Inferensi**	
		Positif	**Negatif**
Aktual	***Match***	3.182	323
	Unmatch	35	12.195

F-Measure			
PPV	***TPR***	***ACC***	***F1-Score***
0,989	**0,907**	**0,977**	**0,947**

Nilai *PPV* (*Positive Predictive Value* atau *Precision*), *TPR* (*True Positive Rate* atau *Recall* / *Sensitivity*), *ACC* (*Accuracy*), dan *F1-Score* (*Harmonic Mean*) didapatkan dari perhitungan berikut:

$$PPV = \frac{TP}{TP + FP} = \frac{3.182}{3.182 + 35} = \frac{3.182}{3.217} = 0{,}989$$

$$TPR = \frac{TP}{TP + FP} = \frac{3.182}{3.182 + 323} = \frac{3.182}{3.505} = 0{,}907$$

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} = \frac{3.182 + 12.195}{3.182 + 12.195 + 35 + 323} = \frac{15.377}{15735} = 0{,}977$$

$$F1 - Score = \frac{2 \times TP}{2 \times TP + FP + FN} = \frac{2 \times 3.182}{2 \times 3.182 + 35 + 323} = \frac{6.364}{6.722} = 0{,}947$$

Tabel 4.12: Hasil Perhitungan *F-Measure* untuk Setiap *Folding*

Fold	***F-Measure***			
	PPV Precision	***TPR Recall***	***ACC Accuracy***	***F1-Score***
1	0,988	0.898	0,975	0.941
2	0,989	0.903	0,977	0.944
3	0,989	0.907	0,977	0.946
4	0,987	0.908	0,977	0.946
5	0,993	0.923	0,981	0.956
Rerata:	**0,989**	**0,907**	**0,977**	**0,947**

Proses perhitungan *F-Measure* sesuai Tabel 4.11, dapat dilakukan untuk setiap *folding*, sehingga hasil nilai *F-Measure* masing-masing *folding* pada *cross validation* dapat dihitung sebagaimana ditampilkan pada Tabel 4.12, dan selanjutnya rata-rata nilai *F-Measure* tiap *folding* dihitung sehingga menghasilkan nilai yang sama dengan di Tabel 4.11.

4.3.2 Evaluasi Proses Inferensi

Hasil inferensi pada sub bab sebelumnya menampilkan hasil dari satu set aturan penuh, sehingga tidak tampak bagaiamana kontribusi aturan tersebut pada hasil akhir. Pada bagian evaluasi ini, setiap kelompok aturan yaitu *Accurate* (A), *Confident* (C), dan *Indecisive* (I) dilakukan proses inferensi secara terpisah untuk membandingkan performa masing-masing terhadap hasil *F-Measure*. Fitur pendukung juga dievaluasi yaitu *Supportive* (s) dan *Consistency* (c) dengan menyertakan saling terpisah pada set aturan penuh untuk melihat pengaruhnya pada hasil inferensi.

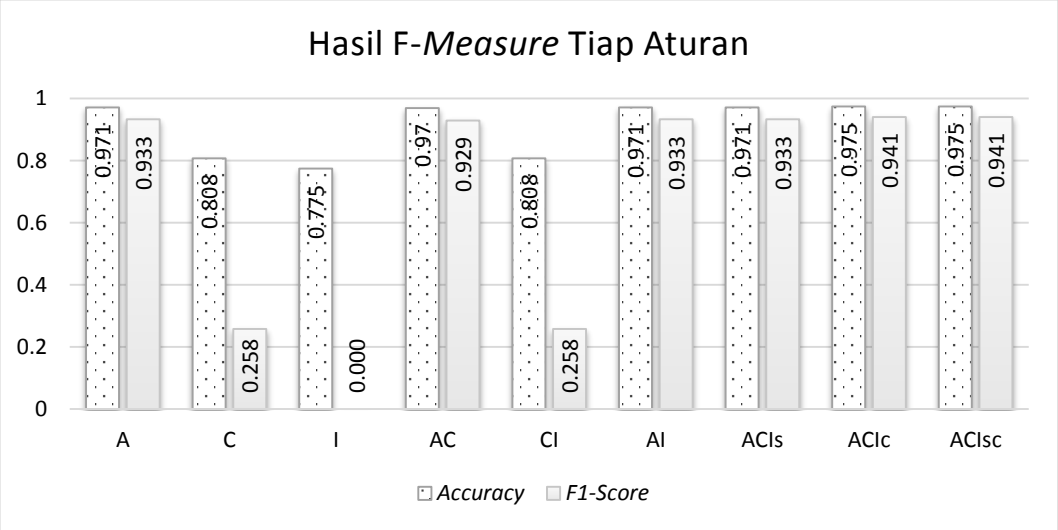
Setiap kelompok aturan dinotasikan menjadi huruf alias yaitu (A)*ccurate*, (C)*onfident*, dan (I)*ndecisive* serta (s)*upportive* dan (c)*onsistency* untuk mewakili set aturan yang diproses beserta kombinasinya. Grafik pada Gambar 4.6 menunjukkan pengukuran *F-Measure* yaitu nilai hasil *accuracy* dan *F1-Score* pada masing-masing set aturan.

Kelompok *Indecisive* (I) sebenarnya tidak layak diproses secara mandiri karena tidak menghasilkan inferensi Positif yang berakibat nilai PPV dan TPR menjadi kosong dan nilai *F1-Score* juga kosong.

Fitur *Supportive* (s) dan *Consistency* (c) hanya diproses terpisah untuk satu set kombinasi penuh ACI, karena pengujian pemilahan fitur tersebut sebenarya untuk membandingkan pengaruh kedua fitur tersebut pada satu set penuh aturan, yaitu ACIsc yang digunakan untuk proses *training* dan *testing* pada sub bab sebelumnya.

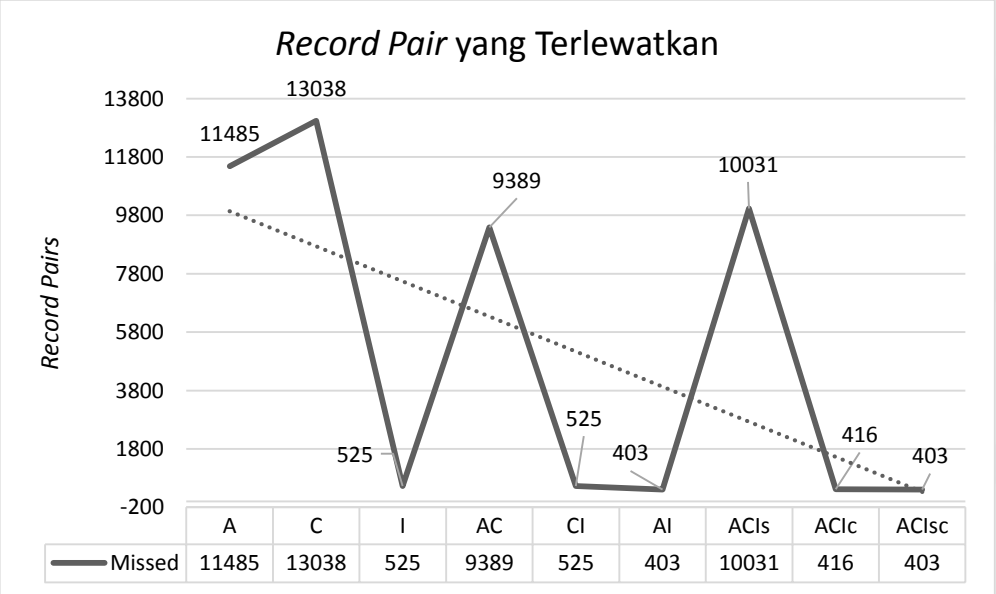
Pada grafik Gambar 4.6 tampak bahwa kombinasi A, ACIs, ACIc, dan ACIsc memiliki nilai *F1-Score* tertinggi, terutama untuk set kombinasi ACIc dan ACIsc yang memiliki nilai akurasi dan *F1-Score* yang sama, namun perbedaan

kedua set aturan tersebut ada pada pengurangan jumlah *missed record pair* (MRP), sebagaimana ditampilkan pada Gambar 4.7.



Gambar 4.6: Grafik *F-Measure* Tiap Aturan

Set aturan ACIc dan ACIsc walaupun memiliki nilai ACC dan *F1-Score* sama, namun dari Gambar 4.7 memiliki jumlah MRP yang lebih kecil, sehingga tren MRP menunjukkan penurunan sesuai set aturan yang semakin kompleks mengikuti antisipasi set aturan terhadap kondisi relevansi *record pair*.

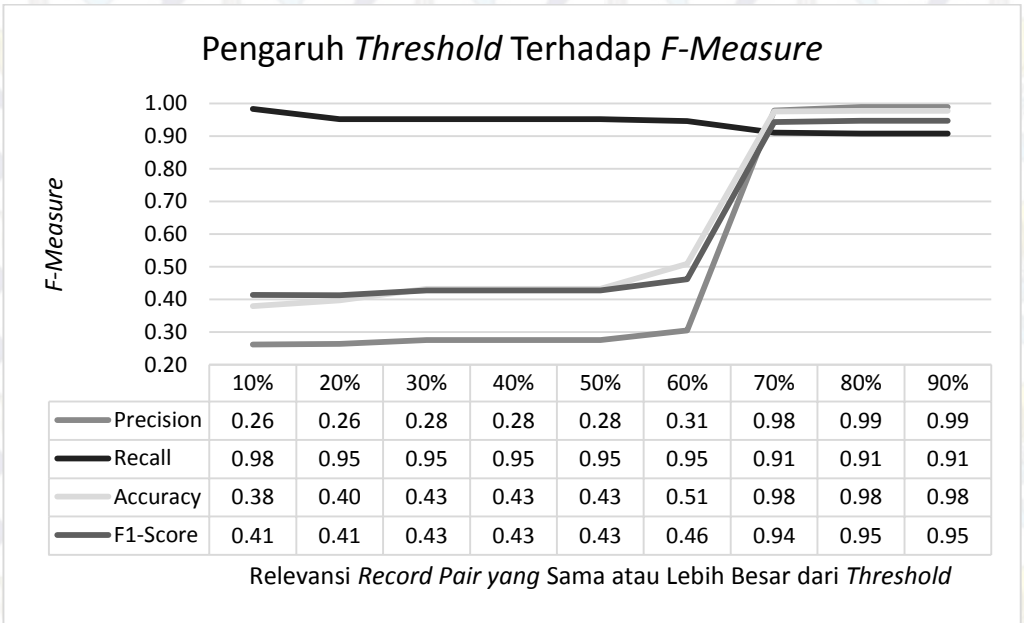


Gambar 4.7: Grafik Penurunan *Trend* pada *Record Pair* yang Terlewatkan

Pengujian ini juga menunjukkan bahwa pemilihan logika aturan yang antisipatif sangat berpengaruh pada kualitas hasil proses inferensi, dibuktikan dengan penambahan fitur *Supportive* dan *Consistency* ternyata semakin memperbaiki hasil inferensi.

Selain penentuan set aturan yang berpengaruh pada kualitas hasil inferensi, pemilihan nilai *threshold* yang sesuai juga berpengaruh signifikan, dibuktikan pada grafik Gambar 4.8, berdasarkan grafik distribusi relevansi *record pair* pada Gambar 4.5 yang menampilkan konsentrasi jumlah *record pair* pada prosentase relevansi 0%, 50%, 70%, dan 100%, sehingga pada proses validasi di sub bab sebelumnya dipilih nilai *threshold* Θ_r = 90%.

Tabel pengukuran *F-Measure* di Gambar 4.8 menampilkan bahwa nilai *F1-Score* pada Θ_r = 90% adalah yang paling optimal dengan nilai 0,95 dan tingkat akurasi 0,98. Namun terdapat kelemahan akibat tidak meratanya distribusi *record pair* pada rentang prosentase relevansi, yaitu dalam hal menilai performa model pendekatan dengan menggunakan kurva ROC (*Receiver Operating Carachteristic*) yang biasa digunakan untuk memvisualisasikan performa penggunaan metode *machine learning*, beserta perbandingan dengan metode sebelumnya sebagaimana akan dipaparkan pada sub bab selanjutnya.

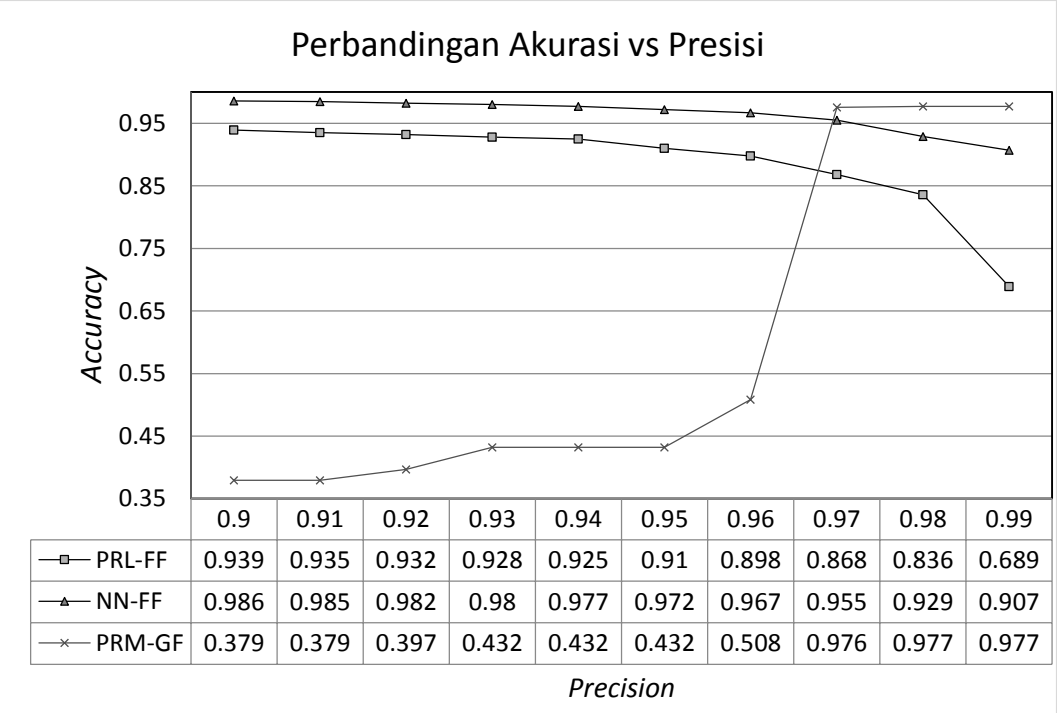


	10%	20%	30%	40%	50%	60%	70%	80%	90%
Precision	0.26	0.26	0.28	0.28	0.28	0.31	0.98	0.99	0.99
Recall	0.98	0.95	0.95	0.95	0.95	0.95	0.91	0.91	0.91
Accuracy	0.38	0.40	0.43	0.43	0.43	0.51	0.98	0.98	0.98
F1-Score	0.41	0.41	0.43	0.43	0.43	0.46	0.94	0.95	0.95

Gambar 4.8: Grafik Pengaruh *Threshold* Terhadap Hasil *F-Measure*

4.3.3 Grafik Akurasi vs Presisi

Tahap akhir dari penelitian ini adalah mereview hasil model pendekatan dengan metode pembanding, yaitu *full-featured* model klasik *Probabilistic Record Linkage* (PRL-FF) dan *Neural Network* (NN-FF) yang diajukan oleh Wilson. Grafik Gambar 4.9 menunjukkan perbandingan dengan model *grouped-feature Probabistic Record Matching* (PRM-GF) yang diajukan pada penelitian ini. Pada grafik tersebut ditampilkan bahwa metode PRM-GF cukup fluktuatif dan hanya baik saat berada di rentang presisi 0,97 sampai dengan 0,99, sedangkan metode pembanding cukup landai. Metode NN-FF memiliki tingkat akurasi sangat baik di presisi 0,94 sampai dengan 0,90 dengan tingkat akurasi tertinggi adalah 0,986 [3], sedangkan metode PRM-GF akurasi tertinggi mencapai 0,977. Sehingga grafik tersebut menampilkan bahwa metode NN-FF masih memiliki nilai akurasi lebih baik menurut pengujiannya.

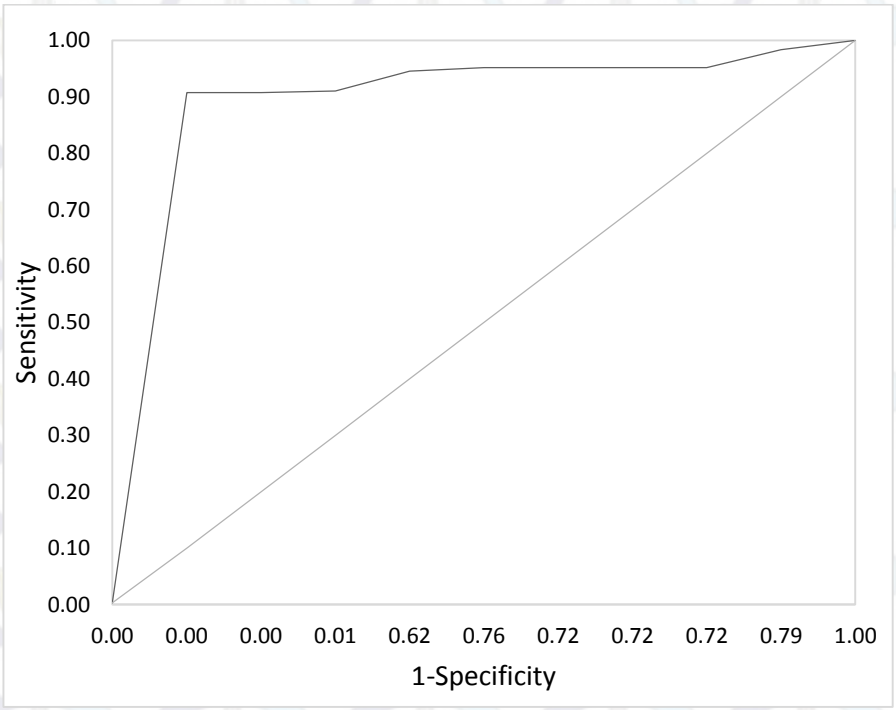


	0.9	0.91	0.92	0.93	0.94	0.95	0.96	0.97	0.98	0.99
PRL-FF	0.939	0.935	0.932	0.928	0.925	0.91	0.898	0.868	0.836	0.689
NN-FF	0.986	0.985	0.982	0.98	0.977	0.972	0.967	0.955	0.929	0.907
PRM-GF	0.379	0.379	0.397	0.432	0.432	0.432	0.508	0.976	0.977	0.977

Gambar 4.9: Grafik Akurasi vs Presisi dengan Metode Pembanding

Jika dilihat dari sisi performa klasifikasi grafik Gambar 4.10 menunjukkan kurva *Receiver Operating Carachteristic* (ROC) dari penentuan *threshold* Θ_r, pada trafik tersebut diambil rentang Θ_r 10% sampai 90% untuk nilai *sensitivity* (TPR) bersama dengan nilai 1-*specifity* (FPR), sehingga terbentuk kurva ROC dengan luas

dibawah kurva (AUC) didapatkan sebesar 0.947 yang tampak kurva tersebut berada di atas garis dasar diagonal (*baseline*).



Gambar 4.10: Kurva ROC Kehandalan Model PRM-GF

Halaman ini sengaja dikosongkan

BAB 5
PENUTUP

Tujuan dari penelitian ini adalah untuk mengajukan alternatif metode *Probabilistic Record Linkage* (PRL) untuk mengatasi kelemahan pendekatan sebelumnya yang mengasumsikan independensi tiap field pada skema tabel, yang pada kenyataannya di dunia nyata tiap field pada skematik relasional database adalah saling berhubungan. Pengujian dilakukan pada pendekatan yang digunakan pada penelitian ini, mulai hasil penentuan fitur, hasil penentuan aturan, pengaruh tiap aturan serta pemilihan *threshold* terhadap performa *F1-Score*, dan perbandingan dengan metode sebelumnya.

5.1 Kesimpulan

Dari hasil penelitian ini didapatkan beberapa poin garis besar yang dapat diambil, yaitu:

- Penentuan fitur dengan model pengelompokan berdasarkan karakteristik field yang kemudian dipecah menjadi beberapa level saling independen, dapat menjadi alternatif dari model fitur sebelumnya pada PRL. Model pengelompokan dan pemecahan level ini tetap mempertahankan sifat independensi untuk fitur inferensi, sehingga aturan *mutual exclusive* untuk variabel bebas pada *machine learning* tidak diabaikan.
- Penentuan aturan *first-order-logic* dari kombinasi tertentu level grup tersebut dapat berpengaruh pada performa inferensi menggunakan metode Markov Logic Networks, yang dibuktikan dengan penurunan tren *missed record pairs* (MRP) serta kenaikan nilai *F1-Measure*.
- Adanya fitur pendukung, yang pada penelitian ini berupa fitur *Supportive* dan *Consistency* dapat membantu meningkatkan performa pendekatan sebagaimana yang ditunjukkan pada penurunan tren MRP dan kenaikan nilai *F1-Score* tersebut.

- Menggunakan hasil pengukuran *Accuracy* vs *Precision*, perbandingan antara metode PRL dengan *neural networks* (PRL-NN), dengan metode Markov Logic Networks (MLN), menunjukkan hasil yang cukup bersaing dengan sedikit perbedaan nilai maksimum akurasi, yaitu 0,986 untuk PRL-NN dan 0,977 untuk MLN.
- Hasil pengujian menggunakan data dengan tingkat unreliabilitas yang tinggi ternyata teteap memberikan hasil yang cukup memuaskan dengan tingkat kehandalan pada skor AUC di 0,947.

5.2 Saran

Penelitian ini masih menggunakan data yang berbeda dengan metode pembanding, yaitu PRL dan PRL-NN yang diajukan oleh Wilson, yang pada metode tersebut tidak menunjukkan tingkat unreliabilitasnya, namun dari penilitian lanjutan ditegaskan bahwa tingkat unreliabilitas data sangat berpengaruh pada hasil akhir untuk inferensi menggunakan model naïve Bayes classifier, sehingga hasil akhir pembandingan akan lebih memperlihatkan performa dari metode pada penelitian ini jika menggunakan pengujian pada data yang sama.

Pemilihan *record pair* pada penelitian ini masih mengabaikan skalabilitas, dengan pertimbangan jumlah data sampel yang masih di skala kecil-menengah. Namun untuk memberikan aspek dukungan skalabilitas penggunaan metode seleksi *record pair* seperti *blocking* dan *windowing* beserta variannya akan cukup memberi nilai lebih pada hasil akhir penelitian ini.

Sebagaimana sudah disimpulkan, bahwa penambahan fitur pendukung ternyata menambah hasil performa yang signifikan, sehingga dengan dukungan logic pada MLN penambahan fitur pendukung ini menjadi penting untuk lebih meningkatkan performa metode.

DAFTAR PUSTAKA

[1] F. Naumann and J. Bleiholder, “Data Fusion in Three Steps : Resolving Inconsistencies at Schema,” *Bull. Tech. Comm. Data Eng.*, pp. 1–11, 2006.

[2] Y. Zhu, Y. Matsuyama, Y. Ohashi, and S. Setoguchi, “When to conduct probabilistic linkage vs. deterministic linkage? A simulation study,” *J. Biomed. Inform.*, vol. 56, pp. 80–86, 2015.

[3] D. R. Wilson, “Beyond probabilistic record linkage: Using neural networks and complex features to improve genealogical record linkage,” *Proc. Int. Jt. Conf. Neural Networks*, pp. 9–14, 2011.

[4] T. C. Ong, M. V. Mannino, L. M. Schilling, and M. G. Kahn, “Improving record linkage performance in the presence of missing linkage data,” *J. Biomed. Inform.*, vol. 52, pp. 43–54, 2014.

[5] M. Richardson and P. Domingos, “Markov logic networks,” no. July 2005, pp. 107–136, 2006.

[6] L. Qing-zhong, Z. Yong-xin, and C. Li-zhen, “Data Conflict Resolution with Markov Logic Networks.”

[7] W. Fan, X. Jia, J. Li, and S. Ma, “Reasoning about record matching rules,” *Proc. VLDB Endow.*, vol. 2, no. 1, pp. 407–418, 2009.

[8] B. Thalheim and Q. Wang, “Data migration: A theoretical perspective,” *Data Knowl. Eng.*, vol. 87, pp. 260–278, 2013.

[9] L. Getoor and A. Machanavajjhala, “Entity resolution: Theory, practice & open challenges,” *Proc. VLDB Endow.*, vol. 5, no. 12, pp. 2018–2019, 2012.

[10] S. Song and L. Chen, “Efficient discovery of similarity constraints for matching dependencies,” *Data Knowl. Eng.*, vol. 87, pp. 146–166, 2013.

[11] Z. Bahmani, L. Bertossi, and N. Vasiloglou, “ERBlox: Combining matching dependencies with machine learning for entity resolution,” *Int. J. Approx. Reason.*, vol. 83, pp. 118–141, 2017.

[12] “Panduan Umum Tata Kelola TIK Nasional.” Kominfo Republik Indonesia, 2007.

[13] “Bisnis Proses Sistem Informasi UIN Malang.” UIN Maulana Malik Ibrahim Malang, Malang, 2017.

[14] M. Lenzerini, “Data Integration: A Theoretical Perspective,” *Proc. 21st ACM SIGMOD-SIGACT-SIGART Symp. Princ. Database Syst.*, pp. 233–246, 2002.

[15] A. Doan, A. Halevy, M. View, and Z. Ives, *Principles of Data Integration*. Elsevier Inc., 2012.

[16] P. Christen, *Data Matching*. 2012.

[17] T. Nadu, T. Nadu, and E. Science, "Data Fusion in Ontology Based Data Integration," no. 978, 2014.

[18] F. Naumann, "Data Fusion – Resolving Data Conflicts for Integration," 2009.

[19] G. L. and M. A, "Entity Resolution for Big Data," 2013. [Online]. Available: http://www.datacommunitydc.org/blog/2013/08/entity-resolution-for-big-data.

[20] I. P. Fellegi and A. B. Sunter, "A Theory for Record Linkage," *J. Am. Stat. Assoc.*, vol. 64, no. 328, pp. 1183–1210, 1969.

[21] V. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," no. Soviet Physics Doklady, 1966.

[22] P. Jaccard, "The distribution of the flora in the alpine zone," no. New Phytologist, 1912.

[23] A. Singhal, "Modern Information Retrieval: A Brief Overview," *Bull. Ieee Comput. Soc. Tech. Comm. Data Eng.*, vol. 24, no. 4, pp. 1–9, 2001.

[24] P. Domingos, "Entity Resolution with Markov Logic," 2006.

[25] F. Gorunescu, *Data Mining: Concepts and Techniques*, vol. 12. 2011.

[26] K. Riec and W. Christian, "Harry: A Tool for Measuring String Similarity." [Online]. Available: http://www.mlsec.org/harry/. [Accessed: 01-Nov-2017].

[27] L. Jin, C. Li, and S. Mehrotra, "Efficient record linkage in large data sets," *Proc. Eighth Int. Conf. Database Syst. Adv. Appl.*, pp. 137–146, 2003.

[28] D. R. Wilson, "Genealogical Record Linkage: Features for Automated Person Matching," *RootsTech 2011*, pp. 331–340, 2011.

[29] K. Stanley, S. Parag, M. Richardson, and P. Domingos, "The Alchemy System for Statistical Relational AI," 2010. [Online]. Available: https://alchemy.cs.washington.edu/user-manual/manual.html. [Accessed: 01-Nov-2017].

[30] P. Christen and K. Goiser, "Quality and complexity measures for data linkage and deduplication," *Qual. Meas. Data Min.*, vol. 151, pp. 127–151, 2007.

[31] "MySQL 8.0 Reference Manual," *Oracle Corporation*, 2018. [Online]. Available: https://downloads.mysql.com/docs/refman-8.0-en.pdf.

LAMPIRAN

Lampiran 1

Daftar Formula Aturan FOL untuk inferensi relevansi *mention* (m)

Rules	**Formula *Fisrt Order Logic* (FOL)**
A.a)	Have(m,Strong,Match) => Relevant(m)
A.b)	Have(m,Strong,Alike) ^ Have(m,Weak,Match) => Relevant(m)
A.c)	(Have(m,Strong,Skimp) v Have(m,Strong,Empty)) ^ Have(m,Weak,Close) => !Relevant(m)
C.a)	Have(m,Strong,Alike) ^ (With(m,Supportive) v With(m,Consistent)) => Relevant(m)
Cs.a)	Have(m,Strong,Alike) ^ With(m,Supportive) => Relevant(m)
Cs.a)	Have(m,Strong,Alike) ^ With(m,Consistent) => Relevant(m)
C.b)	Have(m,Fair,Match) ^ Have(m,Weak,Match) ^ With(m,Consistent) => Relevant(m)
I.a)	Have(m,Strong,Empty) ^ (Have(m,Weak,Match) v Have(m,Weak,Alike)) => Consider(m)
I.b)	Have(m,Fair,Match) ^ (Have(m,Weak,Alike) v Have(m,Weak,Close)) => Consider(m)
I.c)	Consider(m) ^ (With(m,Supportive) v With(m,Consistent)) => Relevant(m)
Is.c)	Consider(m) ^ With(m,Supportive) => Relevant(m)
Ic.c)	Consider(m) ^ With(m,Consistent) => Relevant(m)

Catatan *Rules*: (A)*ccurate*, (C)*onfident*, (I)*ndecisive*, (s)*upportive*, (c)*onsistent*

Lampiran 2

Daftar bobot kombinasi aturan FOL tiap *folding*

No.	*Group*	*Rules*	Hasil Bobot				
			***Fold* 1**	***Fold* 2**	***Fold* 3**	***Fold* 4**	***Fold* 5**
1.	A	A.a)	4.37585	4.43222	4.37279	4.4148	4.34195
	A	A.b)	2.55362	2.18679	2.42226	2.3937	2.20486
	A	A.c)	4.56979	4.48678	4.41588	4.63793	4.54368
2.	C	C.a)	-3.01395	-2.97114	-2.90363	-3.02078	-2.96911
	C	C.b)	4.47339	4.4542	4.44575	4.45841	4.40333
3.	I	I.a)	-4.94862	-4.9183	-4.7889	-4.62237	-4.77912
	I	I.b)	-5.77122	-5.88178	-5.85633	-5.86379	-5.79895
	I	I.c)	8.28464	8.48459	8.21017	8.48343	8.42999
4.	AC	A.a)	4.19659	4.24305	4.19644	4.23435	4.15886
	AC	A.b)	5.69373	5.35352	5.5784	5.54444	5.47297
	AC	C.a)	-3.92297	-3.8492	-3.81797	-3.89071	-3.9204
	AC	C.b)	2.56442	2.51276	2.53664	2.53698	2.49108
	AC	A.c)	4.57215	4.45317	4.41989	4.64072	4.52828
5.	CI	C.a)	-3.11547	-3.12206	-2.93473	-3.12077	-3.12267
	CI	C.b)	4.47349	4.44852	4.44953	4.45947	4.4049
	CI	I.a)	-4.95587	-4.92275	-4.79705	-4.62598	-4.78346
	CI	I.b)	-5.77578	-5.8856	-5.86013	-5.86805	-5.80449
	CI	I.c)	8.28159	8.48074	8.2082	8.48363	8.42712
6.	AI	A.a)	4.38881	4.43122	4.38279	4.4276	4.33321
	AI	A.b)	2.5549	2.19423	2.41373	2.39288	2.21875
	AI	I.a)	-4.9688	-4.93405	-4.81474	-4.61987	-4.77792
	AI	I.b)	-5.82395	-5.9339	-5.9126	-5.91382	-5.86055
	AI	I.c)	8.25149	8.45899	8.18252	8.46108	8.36898
	AI	A.c)	4.56427	4.46602	4.42389	4.63398	4.50142
7.	ACIs	A.a)	4.31289	4.36817	4.3612	4.36258	4.32309
	ACIs	A.b)	2.52253	2.15931	2.46588	2.35483	2.23145
	ACIs	I.a)	-1.25585	-1.14966	-0.31452	-1.24567	-1.19344
	ACIs	I.a)	-9.02238	-8.98094	-9.2231	-9.01157	-9.0877
	ACIs	I.b)	-6.18126	-6.27974	-6.20934	-6.26089	-6.16976
	ACIs	Is.c)	2.57977	2.72077	2.48696	2.5585	2.71639
	ACIs	A.c)	4.57708	4.47206	4.40395	4.62782	4.4926
8.	ACIc	A.a)	4.19095	4.25551	4.20851	4.2286	4.17292
	ACIc	A.b)	5.66815	5.32402	5.53585	5.52019	5.44599
	ACIc	Cc.a)	-3.84413	-3.78813	-3.79785	-3.81489	-3.85846

No.	*Group*	*Rules*	Hasil Bobot				
			***Fold* 1**	***Fold* 2**	***Fold* 3**	***Fold* 4**	***Fold* 5**
	ACIc	C.b)	2.67687	2.65587	2.67202	2.65582	2.64
	ACIc	I.a)	-4.91338	-4.88953	-4.70651	-4.56065	-4.73257
	ACIc	I.b)	-5.79543	-5.91407	-5.87801	-5.89404	-5.82263
	ACIc	Ic.c)	6.87023	6.86841	6.86287	6.86894	6.87008
	ACIc	A.c)	4.56028	4.45869	4.40895	4.6232	4.50312
9.	ACIsc	A.a)	4.19106	4.24602	4.20829	4.22611	4.18606
	ACIsc	A.b)	5.66376	5.33179	5.53404	5.51827	5.44346
	ACIsc	C.a)	-3.93606	-3.86441	-3.82025	-3.90742	-3.94723
	ACIsc	C.b)	2.68982	2.66332	2.69464	2.66952	2.65122
	ACIsc	I.a)	-4.94249	-4.9021	-4.75018	-4.59113	-4.75847
	ACIsc	I.b)	-5.82308	-5.92163	-5.92525	-5.92005	-5.83847
	ACIsc	I.c)	8.2093	8.4379	8.0546	8.42224	8.38602
	ACIsc	A.c)	4.55928	4.46353	4.40877	4.62307	4.5179

Halaman ini sengaja dikosongkan

Lampiran 3

Contoh Ilustrasi Proses ER pada satu sampel data.

Gambar tabel berikut adalah sampel data yang merujuk ke satu entitas di dunia nyata, yang berasal dari beberapa tabel dari beberapa aplikasi Sistem Informasi:

Xid	Sid	Code	Phone	_Name	Rname	Email	Umail	Gdr	Typ
107436	tabah	1979 7041000	0813 6222	mujtabah	mujtabah	tabah79@gmail.com	tabah@uin-malang.ac.id	m	2
4857		1979 041001		m. mujtabah	m. mujtabah				2
6147		1979 1001	0812 4881	m. mujtabah	m. mujtabah, se	tabah79@gmail.com		m	2
15039	P00039	1979 041001		m. mujtabah	m. mujtabah, se				2
19149			0813 76222	m. mujtabah	m. mujtabah				2

Gambar tabel berikut adalah hasil perhitungan kemiripan/*similarity* ditunjukkan dengan tampilan nilai kemiripan pada kurung siku ([*x*]), ID *record pair* adalah ***Xit*** yang membandingkan dua rekord dengan ID ***Xid1*** dan ***Xid2***:

Xit	Xid1	Xid2	_sid	_code	_phone	_name	_email	_umail	_gdr	_typ	tag
105291897	1430	107436	mahya | tabah [34]	1986 011013 | 1979 041000 [48]	| 0813 76222 [0]	riwayatul mahya | mujtabah [38]	40	84	100	100	0
105495401	1458	107436	P00039 | tabah [15]	1979 041001 | 1979 041000 [95]	0812 34881 | 0813 76222 [31]	m. mujtabah | mujtabah [75]	100	0	100	100	1
107472297	1375	107436	| tabah [0]	1981 102001 | 1979 041000 [48]	| 0813 76222 [0]	sri hidayati | mujtabah [29]	0	81	33	100	0
1489136	1458	4857	P00039 | [0]	1979 041001 | 1979 041001 [100]	0812 34881 | [0]	m. mujtabah | m. mujtabah [100]	0	0	0	100	1
1489364	1458	6147	P00039 | [0]	1979 041001 | 1979 041001 [100]	0812 34881 | 0812 4881 [100]	m. mujtabah | m. mujtabah [100]	100	0	100	100	1
1490430	1458	15039	P00039 | P00039 [100]	1979 041001 | 1979 041001 [100]	0812 34881 | [0]	m. mujtabah | m. mujtabah [100]	0	0	0	100	1
1490999	1458	19149	P00039 | [0]	1979 041001 | [0]	0812 34881 | 0813 5222 [31]	m. mujtabah | m. mujtabah [100]	0	0	0	100	1

Gambar tabel berikut adalah konversi nilai *similarity* ke fitur RCKs untuk persiapan koversi ke formula aturan pada MLN (field **tag** adalah penanda status kemiripan/kesamaan rekord yang dilakukan secara manual, 1 berarti *record pair* menandakan sama, dan 0 berarti tidak sama):

Xit	Strong1	Strong2	Strong3	Strong4	Strong5	Fair1	Fair2	Fair3	Fair4	Fair5	Weak1	Weak2	Weak3	Weak4	Weak5	Gender	_Type	Priority	Const	Tag
1489136	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1
1489364	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	1	0	1	1
1490430	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1
1490999	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1
105291897	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	1	0	0	1	0
105495401	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	1	1
107472297	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0

Gambar tabel berikut adalah persiapan konversi ke formula aturan MLN, ID *record pair* ***Xit*** diubah menjadi kode *mention* dengan prefix "**X**":

Xit	Mention	Strong	Fair	Weak	Supportive	Consistent	Tag
1489136	X1489136	Match	Empty	Match		Consistent	1
1489364	X1489364	Match	Match	Match	Supportive	Consistent	1
1490430	X1490430	Match	Empty	Match		Consistent	1
1490999	X1490999	Skimp	Empty	Match		Consistent	1
105291897	X105291897	Skimp	Alike	Skimp		Consistent	0
105495401	X105495401	Alike	Match	Close		Consistent	1
107472297	X107472297	Skimp	Alike	Skimp		Consistent	0

Hasil konversi ke sintaksis formula aturan MLN untuk proses *training* menggunakan *tool* Alchemy:

```
Have(X105291897,High,Skimp) Have(X105291897,Fair,Alike)
Have(X105291897,Weak,Skimp) With(X105291897,Consistent)
!Relevant(X105291897)
Have(X107472297,High,Skimp) Have(X107472297,Fair,Alike)
Have(X107472297,Weak,Skimp) With(X107472297,Consistent)
!Relevant(X107472297)
Have(X1489136,High,Match) Have(X1489136,Fair,Empty)
Have(X1489136,Weak,Match) With(X1489136,Consistent)
Relevant(X1489136)
Have(X1489364,High,Match) Have(X1489364,Fair,Match)
Have(X1489364,Weak,Match) With(X1489364,Supportive)
With(X1489364,Consistent)
Relevant(X1489364)
Have(X1490430,High,Match) Have(X1490430,Fair,Empty)
Have(X1490430,Weak,Match) With(X1490430,Consistent)
Relevant(X1490430)
Have(X1490999,High,Skimp) Have(X1490999,Fair,Empty)
Have(X1490999,Weak,Match) With(X1490999,Consistent)
Relevant(X1490999)
```

Hasil konversi ke sintaksis formula aturan MLN untuk proses *testing* dengan melakukan *Query* probabilistik terhadap atom *Relevant*() berdasarkan *evidence*, misalkan pada *mention* `X105495401` untuk meng-*query* nilai relevansi atom *Relevant(*`X105495401`*)* sebagai berikut:

```
Have(X105495401,High,Alike) Have(X105495401,Fair,Match)
Have(X105495401,Weak,Close) With(X105495401,Consistent)
```

Hasil proses *testing* pada k-*Fold* 1 sampai 5 adalah nilai probabilistik pada atom *Relevant()* tiap *mention* sebagai berikut:

```
Relevant(X1489136) 0.99274
Relevant(X1489364) 0.999693
Relevant(X1490430) 0.99274
Relevant(X1490999) 0.666438
Relevant(X105291897) 0.666435
Relevant(X105495401) 0.0216432
Relevant(X107472297) 0.666435
```

dan kemudian nilai-nilai tersebut diinput balik ke database untuk mempermudah proses lanjutan seperti perhitungan F-*Measure*.

Gambar tabel berikut adalah hasil nilai probabilitas relevansi, Field **Tag** menunjukkan relevansi aktual menggunakan cara manual. Tampak pada gambar bahwa hanya dua hasil inferensi yang salah/meleset (**Tag**=1 tapi nilai relevansi kurang dari *threshold* 0,900), yaitu *mention* `X105495401` dan `X1490999`.

Xit	k_Fold	MLN_Rule	Missed	Relevance	Tag
105495401	1	9	0	21	1
1489136	4	9	0	992	1
1489364	4	9	0	999	1
1490430	4	9	0	992	1
105291897	4	9	0	666	0
107472297	4	9	0	666	0
1490999	5	9	0	666	1

Lampiran 4

Source code yang digunakan untuk proses inferensi.

File shell bash **`0BASEs`** memuat konstanta dan konfigurasi:

```
__DB_NAME=zrdata
__DB_HOST=localhost
__DB_PORT=3306
__DB_USER=apps
__DB_PSWD=

__RULE_FILE=rule9
__N_FOLD=1
__K_FOLD=5

export MYSQL_PWD=$__DB_PSWD

__TB_MERGED_ORIGDATA=a0__merged_origdata      #=> Original Data Mergings
__TB_MERGED_OVRRIDED=a1__merged_overrided     #=> Only Trusted Data + Updated fields
with Resolved data
__TB_MERGED_RESOLVED=a2__merged_resolved      #=> Only Resolved Data
__TB_MERGED_UNRSLVED=a3__merged_unresolved    #=> Only Unresolved data
__TB_MERGED_RUNSLVED=a4__merged_reunsolved    #=> Merged of Resolved and Unresolved
data

__TB_SIMS_TRAINER=b1__similarity_trainer      #=> Similarity Training Vectors #N-
trusted x #M-resolved
__TB_SIMS_VECTORS=b2__similarity_vectors      #=> Similarity Testings Vectors #N-
overrided x #M-unresolved
__TB_SIMS_TESTING=b3__similarity_testing      #=> Similarity Training & Testings
Vectors #N-overrided x (#M-resolved + #M-unresolved)

__TB_RCKS_TRAINER=c1__rcks_trainer            #=> Similarity Training of suitable
RCKs
__TB_RCKS_VECTORS=c2__rcks_vectors            #=> Similarity Testings of suitable
RCKs
__TB_RCKS_TESTING=c3__rcks_testing            #=> Similarity Training & Testings of
suitable RCKs

__TB_RES_TRAINER=d1__res_trainer              #=> MLN inference Result of Training
Vectors
__TB_RES_VECTORS=d2__res_vectors              #=> MLN inference Result of Testings
Vectors
__TB_RES_TESTING=d3__res_testing              #=> MLN inference Result of Training &
Testings Vectors

__TB_OUT_KFOLDS=e1__out_kfolds                #=> K-Folds validation MLN inference
results

__TB_DS_HELPER=z__ds_helper                   #=> Dataset Generation for inference
Helper

__CC=$HOME/Research
__DD=$__CC/ynorms
__EE=$__CC/znorms
```

File shell bash **`8alchemy-infer.sh`** adalah *script* kode untuk proses *training* dan inferensi tiap *folding* berdasarkan parameter input shell:

```
#!/bin/bash

. 0BASEs
```

```
cd $__EE

_NF=$__N_FOLD  # nFold        : K-Fold ke-i
_NN=$__K_FOLD  # Folds        : K value of K-Fold

_X=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "SELECT it FROM
$__TB_SIMS_TESTING WHERE nearly > 0 LIMIT 1"`
[ -z "$_X" ] && {
        _XN=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "SELECT
ROUND(COUNT(1)/$_NN) FROM $__TB_SIMS_TESTING"`
        for ((N=1; N<=$_NN; N++)); do
                [ $N -lt $_NN ] && _NRAND=" ORDER BY RAND() LIMIT $_XN"
                { mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NB <<EOF
                        UPDATE   $__TB_SIMS_TESTING   A   JOIN   (SELECT   X.it   FROM
$__TB_RCKS_TESTING X JOIN $__TB_SIMS_TESTING Y ON X.it = Y.it WHERE _rcks=0$_NRAND)
C ON A.it = C.it SET A.nearly = $N;
                        UPDATE $__TB_RCKS_TESTING C JOIN $__TB_SIMS_TESTING A ON C.it
= A.it SET C._rcks = $N WHERE A.nearly = $N;
EOF
                }
        done
        mysql  $__DB_NAME  -h  $__DB_HOST  -P  $__DB_PORT  -u  $__DB_USER  -NBe  "SELECT
COUNT(1), nearly FROM b3__similarity_testing B GROUP BY B.nearly"
}
[ -n "$1" ] && _NF=$1

_FT=800
[ -n "$2" ] && _FT=$2

_FIT='ACI'
[ -n "$3" ] && _FIT=$3
[               "$_FIT"                =               "A"               ]               ||
__WITH__=",IF(C.rck16+C.rck17+C.rck18>1,CONCAT('With(X',A.it,',Supp)'),''),
                        IF(C.rck19>0,CONCAT('With(X',A.it,',Cons)'),'')
"
_RULE=$__RULE_FILE
_NR_=1
[ -n "$4" ] && _NR_=$4
_RULE="rule$_NR_"

_QUERY=Relevant
[ "$_FIT" = "A" ] && _TRAIN=Relevant || {
        [ "$_FIT" = "C" ] && _TRAIN=Relevant || _TRAIN=Relevant,Consider
}

[ -f ${_RULE}-o$_NF.mln ] && {

        [ -f ${_RULE}-q$_NF.db ] || {

                _TR=0
                { mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NB <<EOF
                SELECT
                        IF(C.rck01>0,CONCAT('Have(X',A.it,',High,Match)'),''),
                        IF(C.rck02>0,CONCAT('Have(X',A.it,',High,Alike)'),''),
                        IF(C.rck03>0,CONCAT('Have(X',A.it,',High,Close)'),''),
                        IF(C.rck04>0,CONCAT('Have(X',A.it,',High,Skimp)'),''),
                        IF(C.rck05>0,CONCAT('Have(X',A.it,',High,Empty)'),''),
                        IF(C.rck06>0,CONCAT('Have(X',A.it,',Fair,Match)'),''),
                        IF(C.rck07>0,CONCAT('Have(X',A.it,',Fair,Alike)'),''),
                        IF(C.rck08>0,CONCAT('Have(X',A.it,',Fair,Close)'),''),
                        IF(C.rck09>0,CONCAT('Have(X',A.it,',Fair,Skimp)'),''),
                        IF(C.rck10>0,CONCAT('Have(X',A.it,',Fair,Empty)'),''),
                        IF(C.rck11>0,CONCAT('Have(X',A.it,',Weak,Match)'),''),
                        IF(C.rck12>0,CONCAT('Have(X',A.it,',Weak,Alike)'),''),
                        IF(C.rck13>0,CONCAT('Have(X',A.it,',Weak,Close)'),''),
                        IF(C.rck14>0,CONCAT('Have(X',A.it,',Weak,Skimp)'),''),
                        IF(C.rck15>0,CONCAT('Have(X',A.it,',Weak,Empty)'),'')
                        $__WITH__
                FROM $__TB_SIMS_TESTING A
                        JOIN $__TB_RCKS_TESTING C ON C.it = A.it
                WHERE   A.nearly = $_NF
```

```
EOF
                } | sed 's/\t/\n/g' | sed '/^$/d' > ${_RULE}-q$_NF.db
        }

        [ -f ${_RULE}-o$_NF.res ] || {
                mln-infer -bp -i ${_RULE}-o$_NF.mln -r ${_RULE}-o$_NF.res -e ${_RULE}-
q$_NF.db -q $_QUERY
                echo -e "\n============\n"
                mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"DELETE FROM $__TB_RES_TESTING WHERE kid = $_NF AND rule = $_NR_"
        }

        X=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "SELECT
it FROM $__TB_RES_TESTING WHERE kid = $_NF AND rule = $_NR_"`
        [ -z "$X" ] && {
                cat ${_RULE}-o$_NF.res | grep 'Relevant(X' | sed 's/Relevant(X//' |
sed 's/)//' > ${_RULE}-x$_NF.res
                { cat<<EOF
                        <?php
                        \$s = ' ';
                        echo "INSERT INTO $__TB_RES_TESTING (it,kid,rule,miss,relv)
VALUES";
                        foreach (new SplFileObject(__DIR__."/${_RULE}-x$_NF.res") as
\$tuple) {
                                \$a = explode(' ', trim(\$tuple)); if (empty(\$a[0]))
continue;
                                \$v = intval(\$a[1]*1000);
                                echo "\$s(\$a[0],$_NF,$_NR_,0,\$v)"; \$s = ',';
                        }
                        echo ";\\n";
EOF
                } | php | sed 's/\t*//' > ${_RULE}-y$_NF.res
                mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER < ${_RULE}-
y$_NF.res
                cat ${_RULE}-o$_NF.res | grep 'Consider(X' | sed 's/Consider(X//' |
sed 's/)//' > ${_RULE}-x$_NF.res
                { cat<<EOF
                        <?php
                        foreach (new SplFileObject(__DIR__."/${_RULE}-x$_NF.res") as
\$tuple) {
                                \$a = explode(' ', trim(\$tuple)); if (empty(\$a[0]))
continue;
                                \$v = intval(\$a[1]*1000);
                                echo       "INSERT       INTO       $__TB_RES_TESTING
(it,kid,rule,miss,cons) VALUE (\$a[0],$_NF,$_NR_,0,\$v) ON DUPLICATE KEY UPDATE cons
= \$v;\\n";
                        }
EOF
                } | php | sed 's/\t*//' > ${_RULE}-y$_NF.res
                mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER < ${_RULE}-
y$_NF.res
                rm -f ${_RULE}-x$_NF.res ${_RULE}-y$_NF.res
                mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"UPDATE $__TB_RES_TESTING C JOIN $__TB_SIMS_TESTING B ON C.it = B.it AND C.kid =
B.nearly SET C.infers = B.infers WHERE C.rule = $_NR_"
        } || {
                S=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"SELECT COUNT(1) FROM $__TB_SIMS_TESTING A JOIN $__TB_RCKS_TESTING C ON C.it = A.it
WHERE A.nearly = $_NF AND A.infers = 1 AND C.rck01 > 0"`
                T=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"SELECT COUNT(1) FROM $__TB_SIMS_TESTING WHERE nearly = $_NF AND infers = 1"`
                M=`cat ${_RULE}-q$_NF.db | grep -o 'X[0-9]*' | sort | uniq | wc -l`;
F=`php -r "echo abs($M-$T);"`
                P=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"SELECT COUNT(1) FROM $__TB_RES_TESTING WHERE kid = $_NF AND rule = $_NR_ AND relv
>= $_FT"`
                N=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"SELECT COUNT(1) FROM $__TB_RES_TESTING WHERE kid = $_NF AND rule = $_NR_ AND relv <
$_FT"`

                echo "---------------"
                echo "Mention Count: $M"
```

```
                echo "Accurate Count: $S"
                echo "--------------"
                echo "Actual Matched [T]: $T"
                echo "Actual Unmatched [F]: $F"
                echo "Inference Positive [P]: $P"
                let Q=$M-$P-$N
                mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"INSERT INTO $__TB_RES_TESTING SELECT B.it, B.nearly, $_NR_, 1, 0, 0, B.infers FROM
$__TB_RES_TESTING C RIGHT JOIN $__TB_SIMS_TESTING B ON C.it = B.it AND C.rule = $_NR_
WHERE C.it IS NULL"
                let N=$N+$Q
                echo "Inference Negative [N]: $N"
                echo "Inference Missing [!]: $Q"
                echo "--------------"
                TP=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"SELECT COUNT(1) FROM $__TB_RES_TESTING WHERE kid = $_NF AND rule = $_NR_ AND infers
= 1 AND relv >= $_FT"`; _TP=${TP#-}
                TN=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"SELECT COUNT(1) FROM $__TB_RES_TESTING WHERE kid = $_NF AND rule = $_NR_ AND infers
= 0 AND relv < $_FT"`; _TN=${TN#-}
                FP=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"SELECT COUNT(1) FROM $__TB_RES_TESTING WHERE kid = $_NF AND rule = $_NR_ AND infers
= 0 AND relv >= $_FT"`; _FP=${FP#-}
                FN=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"SELECT COUNT(1) FROM $__TB_RES_TESTING WHERE kid = $_NF AND rule = $_NR_ AND infers
= 1 AND relv < $_FT"`; _FN=${FN#-}
                #let FP=$F-$P; _FP=${FP#-}
                echo "True Positive [TP]: $_TP"
                echo "True Negative [TN]: $_TN"
                echo "False Positive [FP]: $_FP"
                echo "False Negative [FN]: $_FN"
                echo "--------------"
                echo "Treshold: $_FT"
                echo "--------------"
                _P=`php -r "if ($_TP+$_FP==0) echo 0; else echo $_TP/($_TP+$_FP);"`
                _A=`php -r "if ($_TP+$_TN+$_FP+$_FN==0) echo 0; echo
($_TP+$_TN)/($_TP+$_TN+$_FP+$_FN);"`
                _R=`php -r "if ($_TP+$_FN==0) echo 0; else echo $_TP/($_TP+$_FN);"`
                echo "Precision: $_P"
                echo "Accuracy: $_A"
                echo "Recall: $_R"
                echo "--------------"
                _F=`php -r "if ($_P+$_R==0) echo 0; else echo (2*$_P*$_R/($_P+$_R));"`
                Z=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe
"SELECT kid FROM $__TB_OUT_KFOLDS WHERE kid = $_NF AND rule = $_NR_ AND pole = $_FT"`
                [ -z "$Z" ] && mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u
$__DB_USER -NBe "INSERT INTO $__TB_OUT_KFOLDS VALUES
($_NF,$_NR_,$_FT,$M,$S,$T,$F,$P,$N,$Q,$_TP,$_TN,$_FP,$_FN,$_P,$_A,$_R,$_F)"
                echo "F1-Score: $_F"
                echo "--------------"
        }
} || {

        [ -f ${_RULE}-t$_NF.mln ] || {
                { mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NB <<EOF
                SELECT
                        IF(C.rck01>0,CONCAT('Have(X',A.it,',High,Match)'),''),
                        IF(C.rck02>0,CONCAT('Have(X',A.it,',High,Alike)'),''),
                        IF(C.rck03>0,CONCAT('Have(X',A.it,',High,Close)'),''),
                        IF(C.rck04>0,CONCAT('Have(X',A.it,',High,Skimp)'),''),
                        IF(C.rck05>0,CONCAT('Have(X',A.it,',High,Empty)'),''),
                        IF(C.rck06>0,CONCAT('Have(X',A.it,',Fair,Match)'),''),
                        IF(C.rck07>0,CONCAT('Have(X',A.it,',Fair,Alike)'),''),
                        IF(C.rck08>0,CONCAT('Have(X',A.it,',Fair,Close)'),''),
                        IF(C.rck09>0,CONCAT('Have(X',A.it,',Fair,Skimp)'),''),
                        IF(C.rck10>0,CONCAT('Have(X',A.it,',Fair,Empty)'),''),
                        IF(C.rck11>0,CONCAT('Have(X',A.it,',Weak,Match)'),''),
                        IF(C.rck12>0,CONCAT('Have(X',A.it,',Weak,Alike)'),''),
                        IF(C.rck13>0,CONCAT('Have(X',A.it,',Weak,Close)'),''),
                        IF(C.rck14>0,CONCAT('Have(X',A.it,',Weak,Skimp)'),''),
                        IF(C.rck15>0,CONCAT('Have(X',A.it,',Weak,Empty)'),'')
                        $__WITH__
```

```
                        ,CONCAT(IF(A.infers>0,'','!'),'Relevant(X',A.it,')')
                FROM    $__TB_SIMS_TESTING A
                        JOIN $__TB_RCKS_TESTING C ON C.it = A.it
                WHERE   A.nearly != $_NF;
EOF
                } | sed 's/\t/\n/g' | sed '/^$/d' > ${_RULE}-t$_NF.db
                dos2unix ${_RULE}-t$_NF.db
        }

        mln-learnwts -noAddUnitClauses -ms -g -i ${_RULE}.mln -o ${_RULE}-o$_NF.mln
-t ${_RULE}-t$_NF.db -ne $_TRAIN

        echo -e "\n============\n"
        cat ${_RULE}-o$_NF.mln
}
```

File shell bash **`9alchemy-runs.sh`** adalah *script* kode untuk proses *training* dan inferensi satu iterasi k-*Fold* tiap *rule* berdasarkan parameter shell input:

```
#!/bin/bash

. 0BASEs

[ $# -eq 0 ] && {
        echo "Usage: $0 <RULE:7> [Action] [Threshold:900]"
        echo "  RULE: 1=A, 2=C, 3=I, 4=AC, 5=CI, 6=AI, 7=ACIs, 8=ACIc, 9=ACIsc"
        echo "  ACTION:"
        echo "          A => Reinit Foldings, run training, testing, and validation"
        echo "          X => Run training, testing, and validation"
        echo "          Y => Run testing and validation"
        echo "          Z => Run validation only"
        echo
        exit 0
}

_NR_=9
_FITS='ACIsc'           # X: All features, A: Accurate rules, C: Confident rules, I:
Indecisive rules, s: Supportive, c:Consistency
_RULE=$__RULE_FILE
[ -n "$1" ] && _NR_=$1
case "$_NR_" in
        1) _FITS="A";; #=ACI
        2) _FITS="C";;
        3) _FITS="I";; #XX
        4) _FITS="AC";;
        5) _FITS="CI";;
        6) _FITS="AI";;
        7) _FITS="ACIs";;
        8) _FITS="ACIc";;
        9) _FITS="ACIsc";;
        *);;
esac
_RULE="rule$_NR_"

case "$2" in
        A) _A=1;;
        X) _X=1;;
        Y) _Y=1;;
        Z) _Z=1;;
        *);;
esac

_THLD=900
[ -n "$3" ] && _THLD=$3

[ -n "$_A" ] && {
        rm -f $__EE/${_RULE}-o*.mln*
        rm -f $__EE/${_RULE}-t*.db*
```

```
        rm -f $__EE/${_RULE}-q*.db*
        rm -f $__EE/${_RULE}-o*.res*
        rm -f $__EE/${_RULE}-o.out
        mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "UPDATE
$__TB_SIMS_TESTING SET nearly = 0; UPDATE $__TB_RCKS_TESTING SET _rcks = 0; DELETE
FROM $__TB_RES_TESTING WHERE rule = $_NR_; DELETE FROM $__TB_OUT_KFOLDS WHERE rule =
$_NR_ AND pole = $_THLD;"
}
[ -n "$_X" ] && {
        rm -f $__EE/${_RULE}-o*.mln*
        rm -f $__EE/${_RULE}-t*.db*
        rm -f $__EE/${_RULE}-q*.db*
        rm -f $__EE/${_RULE}-o*.res*
        rm -f $__EE/${_RULE}-o.out
        mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "DELETE FROM
$__TB_RES_TESTING WHERE rule = $_NR_; DELETE FROM $__TB_OUT_KFOLDS WHERE rule = $_NR_
AND pole = $_THLD;"
}
[ -n "$_Y" ] && {
        rm -f $__EE/${_RULE}-o*.res*
        rm -f $__EE/${_RULE}-o.out
        mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "DELETE FROM
$__TB_RES_TESTING WHERE rule = $_NR_; DELETE FROM $__TB_OUT_KFOLDS WHERE rule = $_NR_
AND pole = $_THLD;"
}
[ -n "$_Z" ] && {
        rm -f $__EE/${_RULE}-o.out
        mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "DELETE FROM
$__TB_OUT_KFOLDS WHERE rule = $_NR_ AND pole = $_THLD"
}
[ -f $__EE/${_RULE}-o.out ] || {
        echo -e "================================\n" > $__EE/${_RULE}-o.out
        for ((n=1; n<=$__K_FOLD; n++)); do
                ./8alchemy-infer.sh $n $_THLD $_FITS $_NR_
                ./8alchemy-infer.sh $n $_THLD $_FITS $_NR_
                ./8alchemy-infer.sh $n $_THLD $_FITS $_NR_ >> $__EE/${_RULE}-o.out
                echo -e "\n================================\n" >> $__EE/${_RULE}-
o.out
        done
        echo                                                                -e
"\n\n================================\n================================\n"       >>
$__EE/${_RULE}-o.out
        _P=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "SELECT
AVG(precission) FROM $__TB_OUT_KFOLDS"`
        _A=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "SELECT
AVG(accuracy) FROM $__TB_OUT_KFOLDS"`
        _R=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "SELECT
AVG(recall) FROM $__TB_OUT_KFOLDS"`
        _F=`mysql $__DB_NAME -h $__DB_HOST -P $__DB_PORT -u $__DB_USER -NBe "SELECT
AVG(fscore) FROM $__TB_OUT_KFOLDS"`
        echo "# Precision: $_P" >> $__EE/${_RULE}-o.out
        echo "# Accuracy: $_A" >> $__EE/${_RULE}-o.out
        echo "# Recall: $_R" >> $__EE/${_RULE}-o.out
        echo "--------------" >> $__EE/${_RULE}-o.out
        echo "# F1-Score: $_F" >> $__EE/${_RULE}-o.out
        echo                                                                -e
"\n================================\n================================\n"         >>
$__EE/${_RULE}-o.out

}
cat $__EE/${_RULE}-o.out | tail -n 12
```

BIOGRAFI PENULIS

M. Lukluk, Magister di Bidang Keahlian Telematika Konsentrasi Pengelola TIK Pemerintahan (PeTIK), Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember Surabaya angkatan 2016. Merupakan salah satu penerima Beasiswa Kerjasama Kementerian Komunikasi dan Informasi Republik Indonesia dengan Universitas Islam Negeri Maulana Malik Ibrahim Malang. Lahir di Malang pada tanggal 5 Oktober 1979. Anak ketiga dari lima bersaudara dari pasangan (alm) Mursyid Alifi dan Hamimah, suami dari Nurun Nayiroh dan ayah dari Firda Najwa Meutia dan M. Iqbal Habibie. Bertugas pada Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Alamat email*: *luxmile@gmail.com

Riwayat Pendidikan:

1986 – 1992 : MI Raudlatul Ulum Putra Gondanglegi Malang, Jawa Timur

1992 – 1995 : MTs Raudlatul Ulum Putra Gondanglegi Malang, Jawa Timur

1995 – 1998 : MAN I Jember, Jawa Timur

1998 – 2004 : Ilmu Komputer, Universitas Gadjah Mada, Jogjakarta

Riwayat Pekerjaan:

2004 – 2005 : CV. CGSIndonesia, Jogjakarta

2005 – 2006 : PT. Mutiara Digital, Jakarta

2006 – 2009 : Jatis Solution Ecom, Jakarta

2011 – sekarang : Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Halaman ini sengaja dikosongkan