

**PENCARIAN PASAL PADA
KITAB UNDANG-UNDANG HUKUM PIDANA (KUHP)
BERDASARKAN KASUS MENGGUNAKAN METODE
COSINE SIMILARITY DAN LATENT SEMANTIC INDEXING
(LSI)**

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
SETYOKO YUDHO BASKORO
NIM. 115060807113048



PROGRAM STUDI TEKNIK INFORMATIKA / ILMU KOMPUTER
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

PENCARIAN PASAL PADA KITAB UNDANG-UNDANG HUKUM PIDANA (KUHP)
BERDASARKAN KASUS MENGGUNAKAN METODE
COSINE SIMILARITY DAN LATENT SEMANTIC INDEXING (LSI)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Setyoko Yudho Baskoro
NIM: 115060807113048

Skripsi ini telah diuji dan dinyatakan lulus pada
8 Januari 2016

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Dosen Pembimbing II

Drs. Achmad Ridok, M.Kom
NIP: 19680825 199403 1 002

M. Tanzil Furgon, S.Kom, M.CompSc
NIP: 19820930 200801 1 004

Mengetahui
Ketua Program Studi Informatika/Ilmu Komputer

Drs. Marji, M.T.
NIP: 19670801 199203 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 15 Januari 2016



Setyoko Yudho Baskoro
NIM: 115060807113048

KATA PENGANTAR

Puji syukur Alhamdulillah penulis panjatkan kehadirat ALLAH SWT yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul “Pencarian Pasal pada Kitab Undang-Undang Hukum Pidana (KUHP) Berdasarkan Kasus Menggunakan Metode *Cosine Similarity* dan *Latent Semantic Indexing* (LSI)”.

Dalam penulisan skripsi ini tidak lepas dari hambatan dan kesulitan, namun berkat bimbingan, bantuan, nasihat dan saran serta kerjasama dari berbagai pihak, khususnya pembimbing, segala hambatan tersebut akhirnya dapat diatasi dengan baik. Oleh karena itu, pada kesempatan ini penulis ingin mengucapkan terima kasih kepada:

1. Bapak Drs. Achmad Ridok, M.Kom, selaku Dosen Pembimbing I penulis yang telah membimbing penulis selama mengerjakan skripsi ini.
2. Bapak M. Tanzil Furqon, S.Kom, M.CompSc, selaku Dosen Pembimbing II penulis yang telah membimbing penulis selama mengerjakan skripsi ini.
3. Bapak Drs. Marji, M.T, selaku Ketua Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya.
4. Dengan yang tidak lupa teristimewa kepada Ayahanda dan Ibunda tersayang yang telah banyak berkorban, mengasuh dan membesarkan serta membina dan mendidik penulis sejak kecil dengan segala daya upayanya, selalu memberikan do’a yang tulus untuk keberhasilan penulis, selalu memberikan motivasi dan nasehat yang tiada putus dan memberikan dukungan sepenuhnya baik moral maupun material sehingga penulis dapat menyelesaikan skripsi ini.
5. Husnul Khotimah yang selalu memberi semangat, perhatian, dan dukungan serta doa demi terselesaikannya skripsi ini.
6. Staf Dosen dan Pengajar Fakultas Ilmu Komputer, Universitas Brawijaya, yang telah membantu penulis baik secara langsung maupun tidak langsung dalam mengerjakan skripsi ini.
7. Dan teman-teman penulis yang telah mendukung dalam menyelesaikan skripsi ini yang tidak dapat penulis sebutkan namanya satu persatu.

Penulis menyadari bahwa skripsi ini jauh dari kata sempurna. Akhir kata semoga penulisan skripsi ini dapat memberikan manfaat bagi semua pihak.

Malang, 15 Januari 2016

Setyoko Yudho Baskoro
setyokoyudhobaskoro@gmail.com

ABSTRAK

Negara Indonesia adalah negara hukum. Sebagai negara hukum Indonesia memiliki peraturan-peraturan yang mengatur hubungan antar masyarakat, salah satunya adalah hukum pidana. Kumpulan peraturan hukum pidana tersebut tertulis secara tegas berserta ketentuannya di dalam Kitab Undang-undang Hukum Pidana (KUHP) yang berisi ratusan pasal yang mengatur hubungan antar masyarakat berdasarkan nilai, norma, dan kaidah tertentu yang menitik beratkan kepada kepentingan publik. Pada paper ini, *information retrieval* atau sistem temu kembali informasi digunakan untuk mencari pasal pada KUHP berdasarkan suatu deskripsi kasus kejahatan, menggunakan *Latent Semantic Indexing (LSI)*. Teknik dalam LSI mengadopsi proses matematis reduksi dimensi *Singular Value Decomposition (SVD)*. Sistem ini menggunakan 60 pasal sebagai data latih, dan 6 *query* atau deskripsi kejahatan sebagai data uji. Pada masing-masing data pasal KUHP terdapat data berupa bab, pasal, dan isi pasal tersebut. Sistem akan menghitung dan menentukan pasal yang terkait berdasarkan *query* atau deskripsi kasus kejahatan yang dimasukkan. Metode *cosine similarity* digunakan untuk menghitung kesamaan atau kedekatan dokumen pasal dengan *query*. Kinerja dari sistem ditunjukkan dengan hasil pengujian berupa nilai *Mean Average Precision (MAP)* pada masing-masing *k-rank* yaitu 5, 10, 20, 30, 40, 50, dan 59, dengan kinerja tertinggi sebesar 0.8944 yang terdapat pada nilai *k-rank* 40.

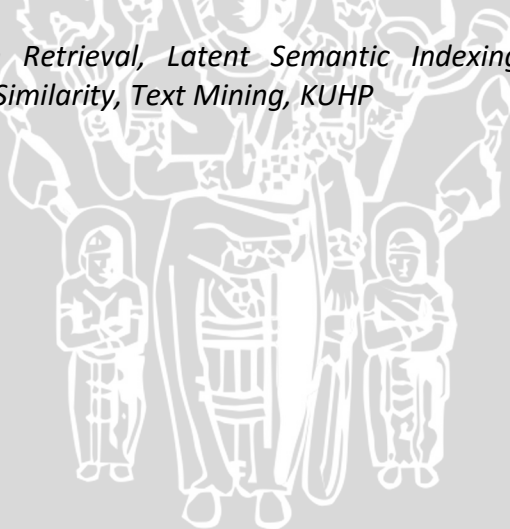
Kata Kunci : *Information Retrieval, Latent Semantic Indexing, Singular Value Decomposition, Cosine Similarity, Text Mining, KUHP*



ABSTRACT

Indonesia is a country of law. As law states, Indonesian have regulations that govern the relationship between the community, one of them is criminal law. Set of rules of criminal law is written in the Kitab Undang-undang Hukum Pidana (KUHP), which contains hundreds of clause which regulate the relationship between the community based on values, norms, and specific rules that focuses on the interests of the public. In this paper, information retrieval used to search the clause of the KUHP based on a description of the crime, using Latent Semantic Indexing (LSI). LSI adopts techniques in mathematical dimension reduction process Singular Value Decomposition (SVD). This system use 60 clause as training data, and 6 query or crime description as test data. In each of the data clause of the KUHP contained data such as clause number, clause, and the clause contents. The system will calculate and determine the relevant clause is based on query or description of the crimes that has been entered. Cosine similarity used to calculate the similarity or proximity clause KUHP with query. The performance of the system is shown by the test results of Mean Average Precision (MAP) value at each k-rank is 5, 10, 20, 30, 40, 50, and 59, with the highest performance is in k-rank 40 with MAP 0.8944.

Keywords: Information Retrieval, Latent Semantic Indexing, Singular Value Decomposition, Cosine Similarity, Text Mining, KUHP



DAFTAR ISI

PENGESAHAN	ii
PERNYATAAN ORISINALITAS	iii
KATA PENGANTAR.....	iv
ABSTRAK.....	v
ABSTRACT.....	vi
DAFTAR ISI.....	vii
DAFTAR TABEL.....	x
DAFTAR GAMBAR.....	xii
DAFTAR LAMPIRAN	xiv
BAB 1 PENDAHULUAN.....	1
1.1 Latar belakang.....	1
1.2 Rumusan masalah.....	2
1.3 Tujuan	3
1.4 Manfaat.....	3
1.5 Batasan masalah	3
1.6 Sistematika penulisan	4
BAB 2 LANDASAN KEPUSTAKAAN	5
2.1 Kajian Pustaka	5
2.2 Hukum Pidana.....	5
2.2.1 Tujuan Hukum Pidana	6
2.2.2 KUHP dan Sistematika Hukum Pidana	6
2.2.3 Fungsi Hukum Pidana	9
2.2.4 Sistem Hukuman	9
2.2.5 Sejarah Pembentukan KUHP	10
2.3 <i>Text Mining</i>	10
2.3.1 Pengertian <i>Text Mining</i>	10
2.3.2 <i>Information Retrieval</i>	11
2.3.3 <i>Text Processing</i>	12
2.3.4 <i>Apache Lucene</i>	15

2.4 TF-IDF	15
2.5 Vector Space Model	16
2.5.1 Cosine Similarity	17
2.6 Singular Value Decomposition (SVD)	17
2.7 Eigenvector dan Eigenvalue	18
2.8 Latent Semantic Indexing (LSI)	19
2.9 Precision dan Recall	19
2.10 Mean Average Precision (MAP)	20
BAB 3 METODOLOGI DAN PERANCANGAN	21
3.1 Studi Literatur	21
3.2 Analisa Kebutuhan	22
3.2.1 Deskripsi Umum Sistem	22
3.2.2 Kebutuhan Data	22
3.3 Perancangan Sistem	22
3.3.1 Preprocessing	24
3.3.2 TF-IDF	27
3.3.3 Singular Value Decomposition	32
3.3.4 Latern Semantic Indexing (LSI)	34
3.3.5 Cosine Similarity	34
3.4 Perancangan User Interface	35
3.4.1 Perancangan Tampilan Sistem	35
3.5 Perancangan Database	38
3.5.1 Tabel KUHP	38
3.5.2 Tabel Training	39
3.5.3 Tabel Testing	39
3.6 Perhitungan Manual	40
3.6.1 Langkah 1 : Preprocessing	40
3.6.2 Langkah 2 : Perhitungan Term Frequency Weight (TF)	42
3.6.3 Langkah 3 : Perhitungan Inverse Document Frequency (IDF)	43
3.6.4 Langkah 4 : Perhitungan TF-IDF	43
3.6.5 Langkah 5 : Perhitungan Singular Value Decomposition (SVD) ..	44
3.6.6 Langkah 6 : Menentukan K-Rank	48



3.6.7 Langkah 7 : Menghitung <i>Latern Semantic Indexing (LSI)</i>	48
3.6.8 Langkah 8 : Menghitung <i>Cosine Similarity</i>	49
BAB 4 IMPLEMENTASI	51
4.1 Spesifikasi Sistem	51
4.1.1 Spesifikasi Perangkat Keras (<i>Hardware</i>)	51
4.1.2 Spesifikasi Perangkat Lunak (<i>Software</i>)	51
4.2 Implementasi Antarmuka Sistem	51
4.2.1 Antarmuka Menu Sistem	52
4.2.2 Antarmuka Kitab Undang Undang Hukum Pidana.....	52
4.2.3 Antarmuka <i>Training</i> KUHP	53
4.2.4 Antarmuka <i>Singular Value Decomposition (SVD)</i> KUHP	55
4.2.5 Antarmuka Pencarian Pasal KUHP	55
4.2.6 Antarmuka Hasil Pencarian Pasal KUHP	56
BAB 5 HASIL ANALISA DAN PEMBAHASAN	58
5.1 Skenario Pengujian	58
5.2 Hasil Pengujian.....	58
5.3 Hasil Analisa	70
BAB 6 PENUTUP	72
6.1 Kesimpulan.....	72
6.2 Saran	72
DAFTAR PUSTAKA	73
LAMPIRAN A PASAL KUHP	75
LAMPIRAN B SOURCE CODE	84

DAFTAR TABEL

Tabel 2.1 Tabel KUPH Buku Pertama	7
Tabel 2.2 Tabel KUPH Buku Kedua	7
Tabel 2.3 Tabel KUPH Buku Ketiga	9
Tabel 3.1 Tabel inverted index	28
Tabel 3.2 Struktur tabel KUHP	39
Tabel 3.3 Struktur tabel <i>training</i>	39
Tabel 3.4 Struktur tabel <i>testing</i>	40
Tabel 3.5 Tabel <i>TF</i>	42
Tabel 3.6 Tabel <i>TF Weight</i>	42
Tabel 3.7 Tabel <i>DF</i> dan <i>IDF</i>	43
Tabel 3.8 Tabel <i>TF-IDF</i>	44
Tabel 3.9 Hasil perhitungan <i>singular values (s)</i>	46
Tabel 3.10 Hasil perhitungan V^T	47
Tabel 3.11 Hasil perhitungan <i>U</i>	47
Tabel 5.1 Data Uji Deskripsi Kasus (<i>Query</i>)	58
Tabel 5.2 <i>Average Precision query 1 k-rank 5</i>	60
Tabel 5.3 <i>Average Precision query 1 k-rank 10</i>	60
Tabel 5.4 <i>Average Precision query 1 k-rank 20</i>	60
Tabel 5.5 <i>Average Precision query 1 k-rank 30</i>	60
Tabel 5.6 <i>Average Precision query 1 k-rank 40</i>	60
Tabel 5.7 <i>Average Precision query 1 k-rank 50</i>	61
Tabel 5.8 <i>Average Precision query 1 k-rank 59</i>	61
Tabel 5.9 <i>Average Precision query 2 k-rank 5</i>	61
Tabel 5.10 <i>Average Precision query 2 k-rank 10</i>	62
Tabel 5.11 <i>Average Precision query 2 k-rank 20</i>	62
Tabel 5.12 <i>Average Precision query 2 k-rank 30</i>	62
Tabel 5.13 <i>Average Precision query 2 k-rank 40</i>	62
Tabel 5.14 <i>Average Precision query 2 k-rank 50</i>	62
Tabel 5.15 <i>Average Precision query 2 k-rank 59</i>	63
Tabel 5.16 <i>Average Precision query 3 k-rank 5</i>	63

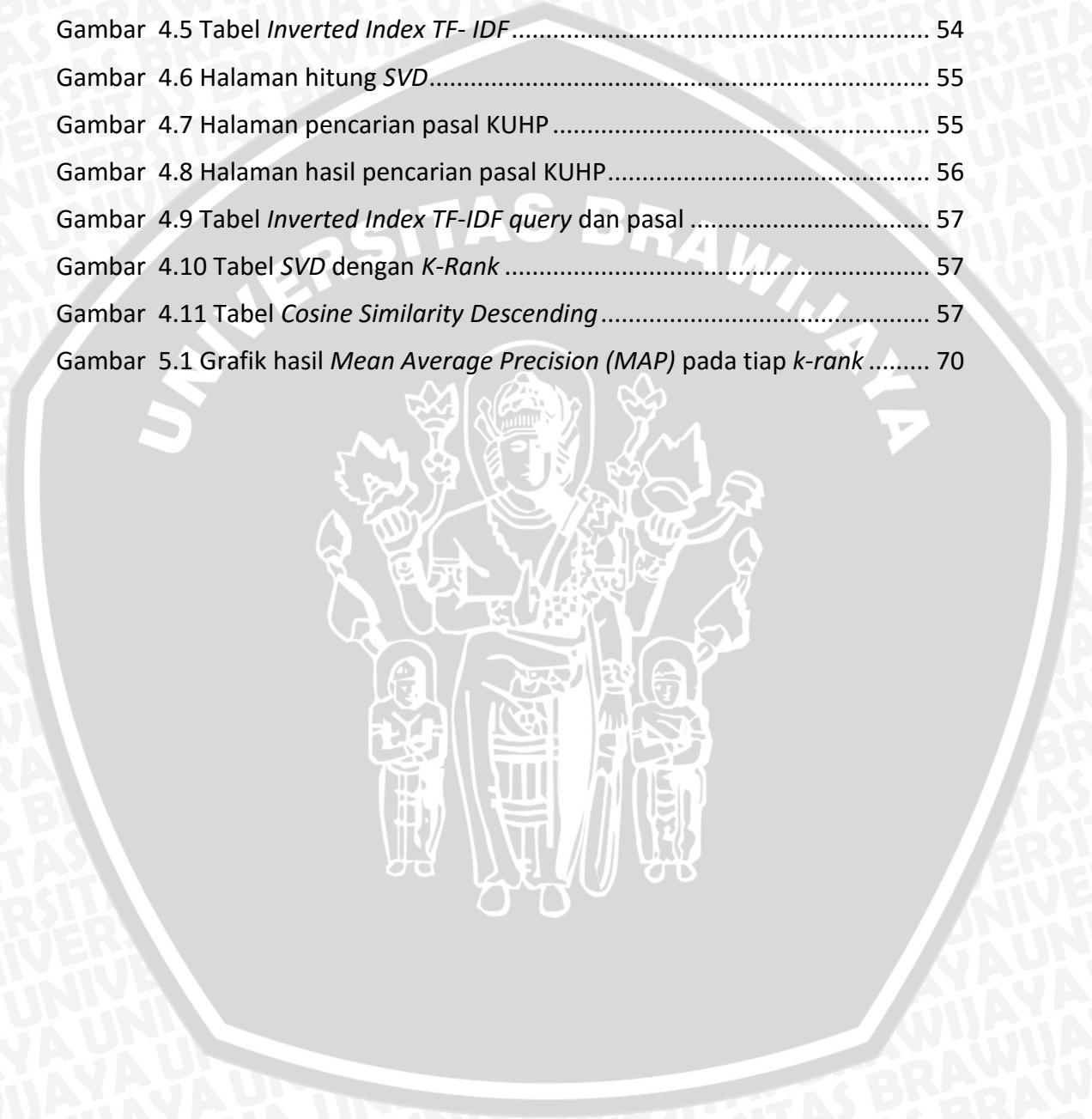
Tabel 5.17 <i>Average Precision query 3 k-rank 10</i>	63
Tabel 5.18 <i>Average Precision query 3 k-rank 20</i>	64
Tabel 5.19 <i>Average Precision query 3 k-rank 30</i>	64
Tabel 5.20 <i>Average Precision query 3 k-rank 40</i>	64
Tabel 5.21 <i>Average Precision query 3 k-rank 50</i>	64
Tabel 5.22 <i>Average Precision query 3 k-rank 59</i>	64
Tabel 5.23 <i>Average Precision query 4 k-rank 5</i>	65
Tabel 5.24 <i>Average Precision query 4 k-rank 10</i>	65
Tabel 5.25 <i>Average Precision query 4 k-rank 20</i>	65
Tabel 5.26 <i>Average Precision query 4 k-rank 30</i>	66
Tabel 5.27 <i>Average Precision query 4 k-rank 40</i>	66
Tabel 5.28 <i>Average Precision query 4 k-rank 50</i>	66
Tabel 5.29 <i>Average Precision query 4 k-rank 59</i>	66
Tabel 5.30 <i>Average Precision query 5 k-rank 5</i>	67
Tabel 5.31 <i>Average Precision query 5 k-rank 10</i>	67
Tabel 5.32 <i>Average Precision query 5 k-rank 20</i>	67
Tabel 5.33 <i>Average Precision query 5 k-rank 30</i>	67
Tabel 5.34 <i>Average Precision query 5 k-rank 40</i>	67
Tabel 5.35 <i>Average Precision query 5 k-rank 50</i>	68
Tabel 5.36 <i>Average Precision query 5 k-rank 59</i>	68
Tabel 5.37 <i>Average Precision query 6 k-rank 5</i>	68
Tabel 5.38 <i>Average Precision query 6 k-rank 10</i>	68
Tabel 5.39 <i>Average Precision query 6 k-rank 20</i>	69
Tabel 5.40 <i>Average Precision query 6 k-rank 30</i>	69
Tabel 5.41 <i>Average Precision query 6 k-rank 40</i>	69
Tabel 5.42 <i>Average Precision query 6 k-rank 50</i>	69
Tabel 5.43 <i>Average Precision query 6 k-rank 59</i>	69
Tabel 5.44 <i>Mean Average Precision</i>	70



DAFTAR GAMBAR

Gambar 2.1 Sejarah KUHP.....	10
Gambar 2.2 Tahapan <i>preprocessing</i>	12
Gambar 2.3 Tahap <i>case folding</i> dan <i>tokenizing</i>	13
Gambar 2.4 Tahap <i>filtrng</i>	14
Gambar 2.5 Proses <i>stemming</i>	15
Gambar 2.6 Ilustrasi dekomposisi matriks SVD	17
Gambar 3.1 Gambaran Umum Tahap Penelitian.....	21
Gambar 3.2 Alur kerja sistem secara umum.....	23
Gambar 3.3 Skema Aliran Data <i>Preprocessing</i>	23
Gambar 3.4 Skema aliran data perhitungan	24
Gambar 3.5 <i>Flowchart preprocessing</i>	25
Gambar 3.6 <i>Flowchart case folding</i>	25
Gambar 3.7 <i>Flowchart tokenizing</i>	26
Gambar 3.8 <i>Flowchart filtering</i>	26
Gambar 3.9 <i>Flowchart stemming</i>	27
Gambar 3.10 <i>Flowchart TF-IDF</i>	28
Gambar 3.11 <i>Flowchart</i> proses hitung <i>TF</i>	30
Gambar 3.12 <i>Flowchart</i> proses hitung <i>IDF</i>	31
Gambar 3.13 <i>Flowchart</i> proses hitung <i>TF-IDF</i>	32
Gambar 3.14 Reduksi SVD dengan <i>k-rank</i>	33
Gambar 3.15 <i>Flowchart SVD</i>	33
Gambar 3.16 <i>Flowchart LSI</i>	34
Gambar 3.17 <i>Flowchart cosine similarity</i>	35
Gambar 3.18 Rancangan <i>user interface</i> menu utama	36
Gambar 3.19 Rancangan <i>user interface</i> untuk input deskripsi kejadian kejahatan	36
Gambar 3.20 Rancangan <i>user interface</i> untuk hasil pencarian pasal.....	37
Gambar 3.21 Rancangan <i>user interface</i> untuk daftar pasal KUHP	38
Gambar 3.22 Reduksi SVD.....	47
Gambar 3.23 Diagram vektor hasil perhitungan <i>cosine similarity</i>	50

Gambar 4.1 Halaman menu utama.....	52
Gambar 4.2 Halaman KUHP	53
Gambar 4.3 Halaman <i>preprocessing</i> KUHP	54
Gambar 4.4 Tabel <i>Inverted Index TF</i> dan <i>IDF</i>	54
Gambar 4.5 Tabel <i>Inverted Index TF-IDF</i>	54
Gambar 4.6 Halaman hitung <i>SVD</i>	55
Gambar 4.7 Halaman pencarian pasal KUHP	55
Gambar 4.8 Halaman hasil pencarian pasal KUHP.....	56
Gambar 4.9 Tabel <i>Inverted Index TF-IDF query</i> dan pasal	57
Gambar 4.10 Tabel <i>SVD</i> dengan <i>K-Rank</i>	57
Gambar 4.11 Tabel <i>Cosine Similarity Descending</i>	57
Gambar 5.1 Grafik hasil <i>Mean Average Precision (MAP)</i> pada tiap <i>k-rank</i>	70



DAFTAR LAMPIRAN

LAMPIRAN A PASAL KUHP	75
A.1 Data Training Pasal KUHP Buku Kedua Tentang Kejahatan.....	75
A.2 Data Uji Deskripsi Kasus (<i>Query</i>).....	83
LAMPIRAN B <i>SOURCE CODE</i>	84
B.1 <i>Tokenizing</i>	84
B.2 <i>Stemming</i>	84
B.3 <i>Term Frequency (TF)</i>	84
B.4 <i>getUniqueKeys</i>	85
B.5 <i>IDF</i>	85
B.6 <i>TF-IDF</i>	86
B.7 <i>Singular Value Decomposition (SVD)</i>	86
B.8 <i>Matriks S</i>	87
B.9 <i>Matriks Vt</i>	87
B.10 <i>Matriks U</i>	88
B.11 <i>Latent Semantic Indexing (LSI)</i>	89
B.12 <i>Matriks S K-rank</i>	90
B.13 <i>Matriks U K-rank</i>	91
B.14 <i>Matriks VT K-rank</i>	91
B.15 <i>Cosine Similarity</i>	92

BAB 1 PENDAHULUAN

1.1 Latar belakang

Menurut Pasal 1 ayat (3) UUD 1945, negara Indonesia adalah negara hukum. Berdasarkan Undang Undang Republik Indonesia nomor 8 tahun 1981 tentang hukum acara pidana, hukum ditempatkan sebagai aturan main dalam kehidupan bermasyarakat, berbangsa, dan bernegara yang menjunjung tinggi hak asasi manusia serta menjamin seluruh warga negara diperlakukan dengan adil dimata hukum. Sebagai negara hukum Indonesia memiliki peraturan-peraturan yang mengatur hubungan antar masyarakat, salah satunya adalah hukum pidana. Hukum pidana merupakan sekumpulan peraturan hukum yang dibuat oleh negara, yang berisi larangan maupun keharusan bagi pelanggar tersebut dikenakan sanksi yang dapat dipaksakan oleh negara. Kumpulan peraturan hukum pidana tersebut tertulis secara tegas bersertakan ketentuannya di dalam Kitab Undang-undang Hukum Pidana (KUHP) yang berisi ratusan pasal yang mengatur hubungan antar masyarakat berdasarkan nilai, norma, dan kaidah tertentu yang menitik beratkan kepada kepentingan publik (Prasetyo, 2014).

Namun fakta menunjukkan, Indonesia sampai sekarang belum juga sampai ke tahap cita-cita negara hukum (Atiq, 2015). Sebagian besar masyarakat di Indonesia tidak mengerti dan memahami peraturan - peraturan dalam bernegara, terutama pada Kitab Undang-Undang Hukum Pidana (KUHP). Hukum pidana sangatlah kompleks sehingga cukup sulit bagi orang awam untuk mengerti dan memilah pasal-pasal yang mengatur suatu kasus dan permasalahan tertentu. Hal tersebut membuat masyarakat menjadi buta akan hukum. Sehingga banyak masyarakat yang melakukan tindak pidana yang menentang aturan yang tercantum dalam KUHP. Bahkan sering kali masyarakat awam menjadi korban dari penegakan hukum yang tidak adil (Republika, 2015).

Di era informasi saat ini, kemajuan teknologi komputer berkembang dengan sangat pesat. Perkembangan teknologi tidak hanya pada ilmu komputer saja, perkembangan ini juga mencakup hampir seluruh bidang di luar disiplin ilmu komputer, sehingga hal ini sangat membantu kehidupan manusia dalam menyelesaikan berbagai masalah dengan cepat dan efisien. Perkembangan teknologi memunculkan revolusi dan inovasi dalam ilmu pengetahuan, khususnya dalam teknologi *Information Retrieval* atau Sistem Temu Kembali Informasi. Dengan bantuan teknologi ini, permasalahan hukum di Indonesia dapat dikurangi dengan mengembangkan *Information Retrieval* terhadap pasal pada KUHP sehingga pencarian pasal berdasarkan suatu kasus dapat dengan mudah dan cepat dilakukan. Dengan adanya sistem tersebut dapat membantu masyarakat dalam memahami dan menemukan pasal yang berkaitan tentang suatu kejadian kejahatan yang terjadi di lingkungan sekitar, sehingga masyarakat menjadi mengerti pelanggaran dan akibat yang diterima berdasarkan KUHP, jika melakukan kegiatan yang melanggar ketentuan-ketentuan yang berlaku pada KUHP.

Information Retrieval yang dapat dikembangkan untuk pencarian pasal pada KUHP adalah dengan *Vector Space Model (VSM)*, yaitu model yang digunakan untuk menghitung besarnya derajat kemiripan diantara vektor (Kontostathis, 2007). Di dalam perhitungan VSM digunakan perhitungan *Cosine Similarity* untuk menghitung kesamaan atau kedekatan antar dokumen (Wati, 2012). Tetapi metode *Vector Space Model (VSM)* memiliki beberapa kelemahan, yaitu tidak mampu membedakan dan mengenali kata *synonym* dan *polysemy* (Kozareva, 2013). Menurut Kamus Besar Bahasa Indonesia (KBBI), sinonim adalah kata yang memiliki bentuk yang berbeda dengan kata lain namun memiliki arti, makna atau pengertian yang sama atau mirip. Sinonim bisa disebut juga dengan persamaan makna kata atau padanan kata. Sedangkan polisemi merupakan kata-kata yang memiliki makna atau arti lebih dari satu.

Untuk menutupi kekurangan dari VSM tersebut, maka dapat menggunakan metode *Latent Semantic Indexing (LSI)*, yang dapat menemukan hubungan semantik antar kata dan ketekaitan makna dengan kata. Banyaknya pasal pada KUHP juga menjadi beban dalam proses perhitungan, maka diperlukan reduksi pada dimensi data untuk mempercepat proses perhitungan (Sari, 2012). Untuk reduksi data pada pasal-pasal KUHP, menggunakan *Singular Value Decomposition (SVD)* merupakan model matematis yang digunakan untuk reduksi dimensi data (Samat, 2009).

Penelitian mengenai LSI telah dilakukan oleh Yunita, Ahcmad Ridok, dan Mardji (2012) mengenai penentuan lirik lagu berdasarkan emosi menggunakan sistem temu kembali informasi dengan metode LSI, yang menggunakan data berupa lirik lagu berbahasa. Analisis hasil yang didapatkan sistem dapat bekerja dengan baik, dibuktikan dengan nilai MAP yang rata-rata mendekati nilai 1, menunjukkan bahwa sistem dapat mendeteksi kemiripan makna antara *query* dengan *corpus* lirik lagu, dengan *k-rank* bernilai 10 (Sari, 2012).

Sehingga dengan perpaduan dua metode tersebut yaitu *Latent Semantic Indexing (LSI)* dan *Cosine Similarity* diharapkan akan menghasilkan pencarian pasal KUHP lebih cepat dan akurat.

Berdasarkan latar belakang diatas, maka penulis menggunakan judul **“Pencarian Pasal Pada Kitab Undang-Undang Hukum Pidana (KUHP) Berdasarkan Kasus Menggunakan Metode *Cosine Similarity* dan *Latent Semantic Indexing (LSI)*”**.

1.2 Rumusan masalah

Dari latar belakang masalah di atas, maka penulis merumuskan pokok permasalahan, yaitu :

1. Bagaimana menerapkan *Information Retrieval* dalam pencarian pasal pada Kitab Undang-Undang Hukum Pidana (KUHP) menggunakan metode *Latent Semantic Indexing (LSI)* dan *Cosine Similarity* ?

2. Bagaimana akurasi dari hasil pencarian pasal pada Kitab Undang-Undang Hukum Pidana (KUHP) berdasarkan suatu kasus menggunakan metode *Latent Semantic Indexing (LSI)* dan *Cosine Similarity*?

1.3 Tujuan

Tujuan yang ingin dicapai dalam penelitian ini adalah sebagai berikut :

1. Menerapkan metode *Latent Semantic Indexing (LSI)* dan *Cosine Similarity* pada *Information Retrieval* dalam pencarian pasal pada Kitab Undang-Undang Hukum Pidana (KUHP) .
2. Mengetahui akurasi dari hasil pencarian pasal pada Kitab Undang-Undang Hukum Pidana (KUHP) berdasarkan suatu kasus menggunakan metode *Latent Semantic Indexing (LSI)* dan *Cosine Similarity*.

1.4 Manfaat

Penelitian yang dilakukan diharapkan dapat memberikan manfaat bagi pihak-pihak berkepentingan, antara lain yaitu :

1. Mengetahui dan memahami pasal-pasal yang terkait dengan suatu pelanggaran atau suatu kasus secara tepat dan cepat.
2. Mempercepat pencarian pasal-pasal yang terkait dengan suatu pelanggaran atau kasus secara tepat.
3. Dapat memudahkan masyarakat awam untuk mengetahui dan mengerti serta mencari pasal-pasal KUHP yang berhubungan dengan kasus tindak pidana.

1.5 Batasan masalah

Untuk memfokuskan penelitian yang akan dilakukan permasalahan yang dibatasi adalah sebagai berikut:

1. Materi hukum untuk data training dari Pencarian Pasal Pada Kitab Undang-Undang Hukum Pidana (KUHP) Berdasarkan Kasus Menggunakan Metode *Cosine Similarity* dan *Latent Semantic Indexing (LSI)*, diadopsi dari Kitab Undang - Undang Hukum Pidana (KUHP).
2. Pasal yang digunakan dalam Kitab Undang - Undang Hukum Pidana (KUHP) adalah KUHP Buku Kedua Tentang Kejahatan, sebanyak 60 pasal.
3. Sistem ini berupa aplikasi berbasis desktop dengan bahasa *Java* dengan *MySQL Database*.
4. *Output* dari aplikasi ini berupa pasal yang berlaku berdasarkan suatu kasus yang diinputkan oleh *user*.

1.6 Sistematika penulisan

Dalam proposal skripsi ini dibagi menjadi enam bab dengan beberapa sub-bab. Adapun sistematika penulisan proposal skripsi adalah sebagai berikut:

BAB I : PENDAHULUAN

Dalam bab ini akan diuraikan tentang latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, ruang lingkup atau batasan masalah, sistematika penulisan dari penelitian yang diangkat.

BAB II : LANDASAN KEPUSTAKAAN

Dalam bab ini akan diuraikan mengenai teori-teori yang akan digunakan pada penelitian ini dan sumber-sumber teori tersebut kemudian diuraikan.

BAB III : METODOLOGI DAN PERANCANGAN

Dalam bab ini diterangkan bagaimana gambaran sistem Pencarian Pasal Pada Kitab Undang-Undang Hukum Pidana (KUHP) Berdasarkan Kasus Menggunakan Metode *Cosine Similarity* dan *Latent Semantic Indexing (LSI)*. Selain itu juga dijelaskan analisis kebutuhan dari sistem.

BAB IV : IMPLEMENTASI

Hasil perancangan sistem kemudian diimplementasikan dalam bentuk program aplikasi. Bab ini akan menjelaskan bagaimana langkah-langkah melakukan implementasi tersebut. Implementasi program menggunakan bahasa pemrograman *JAVA* dan *MySQL database*.

BAB V : HASIL ANALISA DAN PEMBAHASAN

Dalam bab ini dijelaskan mengenai langkah-langkah uji coba sistem dan analisa hasil dari uji coba yang telah dilakukan.

BAB VI : PENUTUP

Pada bab ini dijelaskan mengenai kesimpulan dan saran dari keseluruhan hasil penelitian Tugas Akhir.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Terdapat beberapa metode dalam pengembangan *Information Retrieval* yang saat ini telah dikembangkan baik untuk kebutuhan riset maupun komersial. Salah satu metode yang dapat digunakan untuk mengembangkan *Information Retrieval* terhadap pasal pada KUHP yaitu *Vector Space Model (VSM)* dengan menggunakan *Cosine Similarity*, yaitu fungsi yang digunakan untuk menghitung besarnya derajat kemiripan diantara vektor (Kontostathis, 2007), dan untuk menghitung kesamaan atau kedekatan antar dokumen. Tetapi metode *Vector Space Model (VSM)* memiliki beberapa kelemahan, salah satu kelemahan dari *Vector Space Model (VSM)* yaitu tidak mampu membedakan dan mengenali kata *synonym* dan *polysemy* (Kozareva, 2013). Banyaknya pasal pada KUHP juga menjadi beban dalam proses perhitungan, maka diperlukan reduksi pada dimensi data untuk mempercepat proses perhitungan pada *Cosine Similarity*.

Untuk menutupi kekurangan dari *VSM* tersebut, maka dapat menggunakan metode *Latent Semantic Indexing (LSI)*, yang dapat menemukan hubungan semantik antar kata dan ketekaitan makna (Sari, 2012). Untuk reduksi data pada pasal-pasal KUHP, menggunakan *Singular Value Decomposition (SVD)* untuk mengurangi proses komputasi (Samat, 2009). Sehingga dengan perpaduan dua metode tersebut yaitu *Latent Semantic Indexing (LSI)* dan *Cosine Similarity* diharapkan akan menghasilkan pencarian pasal KUHP lebih cepat dan akurat.

Penelitian mengenai *LSI* telah dilakukan oleh Yunita, Ahcmad Ridok, dan Mardji (2012) mengenai penentuan lirik lagu berdasarkan emosi menggunakan sistem temu kembali informasi dengan metode *LSI*. Sistem yang dikembangkan menggunakan data berupa lirik lagu berbahasa Indonesia dengan jumlah lirik lagu pada *corpus* 370 lirik dan pada *query* terdapat 5 lirik. Analisis hasil didapatkan sistem dapat bekerja dengan baik, dibuktikan dengan nilai *MAP* yang rata-rata mendekati nilai 1, menunjukkan bahwa sistem dapat mendeteksi kemiripan makna antara *query* dengan *corpus* lirik lagu, dengan *k-rank* bernilai 10 (Sari, 2012).

2.2 Hukum Pidana

Menurut Martiman Prodjohamidjojo, hukum pidana adalah bagian dari keseluruhan hukum yang berlaku di suatu negara, yang mengadakan dasar - dasar dan aturan – aturan untuk menentukan perbuatan – perbuatan mana yang tidak boleh dilakukan, yang dilarang, dengan disertai ancaman atau sanksi pidana tertentu bagi siapa saja yang melanggarnya. Menentukan kapan dan dalam hal apa kepada mereka yang telah melakukan larangan – larangan itu dapat dikenakan atau dijatuhi pidana sebagaimana yang telah dicantumkan. Menentukan dengan cara bagaimana pengenaan pidana itu dapat dilaksanakan apabila orang yang diduga telah melanggar ketentuan tersebut (Prasetyo, 2014).

Secara singkat hukum pidana dibagi menjadi dua bagian yakni (Prasetyo, 2014):

1. Hukum pidana materiil, yaitu hukum pidana yang berisi bahan atau materinya, ialah norma dan sanksinya termasuk di dalamnya orang yang bagaimana atau dalam keadaan bagaimana dapat dijatuhi pidana.
2. Hukum pidana formal, biasa disebut hukum acara pidana, yaitu suatu tata cara aturan yang mengatur bagaimana pidana itu dapat dilaksanakan bila ada seseorang yang melanggar hukum pidana materiil. Dengan kata lain, hukum pidana formail atau hukum acara pidana adalah hukum yang menegakkan atau mempertahankan hukum pidana materiil.

Hukum pidana merupakan hukum yang memiliki sifat khusus, yaitu dalam hal sanksinya. Didalamnya terdapat ketentuan tentang apa yang harus dilakukan dan apa yang tidak boleh dilakukan, serta akibatnya. Yang membedakan hukum pidana dengan hukum yang lainnya, diantaranya adalah bentuk sanksinya, yang bersifat negatif yang disebut dengan pidana (hukuman). Bentuk dari pidana bermacam – macam, dari dipaksa diambil hartanya karena harus membayar denda, bahkan dapat pula dirampas nyawanya, jika dijatuhi pidana mati (Prasetyo, 2014).

Atas dasar Kitab Undang - Undang Hukum Pidana Pasal 1 yang mengatakan bahwa perbuatan yang pelakunya dapat dipidana atau dihukum adalah perbuatan yang sudah disebutkan di dalam perundang-undangan. Tindak pidana merupakan perbuatan yang dapat dikenakan hukuman karena merupakan pelanggaran terhadap undang-undang tindak pidana (Prasetyo, 2014).

2.2.1 Tujuan Hukum Pidana

Dalam Rancangan KUHP Juli Tahun 2006, tujuan pemidanaan ditentukan dalam Pasal 51, yaitu Pemidanaan bertujuan:

1. Mencegah dilakukannya tindakan pidana dengan menegakkan norma hukum demi pengayoman masyarakat;
2. Memanyarakatkan terpidana dengan mengadakan pembinaan sehingga menjadi orang yang baik dan berguna;
3. Menyelesaikan konflik yang ditimbulkan oleh tindak pidana, memulihkan keseimbangan, dan mendatangkan rasa damai dalam masyarakat.

Tujuan hukum pidana mengandung makna pencegahan terhadap gejala-gejala sosial yang kurang sehat disamping melakukan rehabilitasi bagi yang sudah terlanjur tidak berbuat baik. Jadi hukum Pidana, berisi ketentuan-ketentuan yang mengatur dan membatasi tingkah laku manusia dalam mengendalikan pelanggaran terhadap kepentingan umum (Prasetyo, 2014).

2.2.2 KUHP dan Sistematika Hukum Pidana

Hukum pidana di Indonesia tertulis dalam sebuah kitab undang-undang. Hukum pidana tersebut dikodifikasikan beserta ketentuan-ketentuannya di dalam Kitab Undang-Undang Hukum Pidana (KUHP) yang berasal dari zaman pemerintah penjajahan Belanda (Prasetyo, 2014).

Kitab undang-Undang Hukum Pidana (KUHP) terdiri atas 569 pasal, secara sistematis dibagi dalam (Prasetyo, 2014):

Buku I : Memuat tentang Ketentuan-ketentuan Umum. Pasal 1-103.

Buku II : Mengatur tentang Kejahatan. Pasal 104-488.

Buku III : Mengatur tentang Pelanggaran. Pasal 489-569.

Berikut merupakan rincian dari KUHP pada tabel 2.1, 2.2, dan 2.3 dibawah.

Buku Kesatu: Aturan Umum

Tabel 2.1 Tabel KUPH Buku Pertama

BAB	Prihal	Pasal	Jmlh
I	Batas-batas berlakunya Aturan Pidana dalam Perundang-undangan	1-9	9
II	Pidana	10-43	34
III	Hal-hal yang Menghapuskan, Mengurangi atau Memberatkan Pidana	44-52	9
IV	Percobaan	53-54	2
V	Penyertaan Dalam Tindak Pidana	55-62	8
VI	Perbarengan Tindak Pidana	63-71	9
VII	Mengajukan dan Menarik Kembali Pengaduan dalam Hal Kejahatan yang Hanya Dituntut atas Pengaduan	72-75	4
VIII	Hapusnya Kewenangan Menuntut Pidana dan Menjalankan Pidana	76-85	10
IX	Arti Beberapa Istilah yang Dipakai dalam Kitab Undang-undang	86-102	17
	Aturan Penutup	103	1

Buku Kedua : Kejahatan

Tabel 2.2 Tabel KUPH Buku Kedua

Bab	Prihal	Pasal	Jmlh
I	Kejahatan Terhadap Keamanan Negara	104-129	26
II	Kejahatan terhadap Martabat Presiden dan Wakil Presiden	130-139	10
III	Kejahatan terhadap Negara Sahabat dan Terhadap Kepala Negara Sahabat Serta Wakilnya	139-145	7
IV	Kejahatan Terhadap Melakukan Kewajiban dan Hak Kenegaraan	146-153	8
V	Kejahatan Terhadap Ketertiban Umum	154-181	28

Tabel 2.2 Tabel KUPH Buku Kedua

Bab	Prihal	Pasal	Jmlh
VI	Perkelahian Tanding	182-186	5
VII	Kejahatan yang Membahayakan Keamanan Umum bagi Orang atau Barang	187-206	20
VIII	Kejahatan Terhadap Penguasa Umum	207-241	35
IX	Sumpah Palsu dan Keterangan Palsu	242-243	2
X	Pemalsuan Mata Uang dan Uang Kertas	244-252	9
XI	Pemalsuan Meterai dan Merek	253-262	10
XII	Pemalsuan Surat	263-276	14
XIII	Kejahatan Terhadap Asal-Usul dan Perkawinan	277-280	4
XIV	Kejahatan Terhadap Kesusilaan	281-303	23
XV	Meninggalkan Orang yang Perlu Ditolong	304-309	6
XVI	Penghinaan	310-321	12
XVII	Membuka Rahasia	322-323	2
XVIII	Kejahatan Terhadap Kemerdekaan Orang	324-337	14
XIX	Kejahatan Terhadap Nyawa	338-350	13
XX	Penganiayaan	351-358	8
XXI	Menyebabkan Mati atau Luka-luka Karena Kealpaan	359-361	3
XXII	Pencurian	362-367	6
XXIII	Pemerasan dan Pengancaman	368-371	4
XXIV	Penggelapan	372-377	6
XXV	Perbuatan Curang	378-395	18
XXVI	Perbuatan Merugikan Pemiutang atau Orang yang Mempunyai Hak	396-405	10
XXVII	Menghancurkan atau Merusakkan Barang	406-412	7
XXVIII	Kejahatan Jabatan	413-437	25
XXIX	Kejahatan Pelayaran	438-479	42
XXIX A	Kejahatan Penerbangan dan Kejahatan Terhadap Sarana / Prasarana Penerbangan	479a- 479r	1
XXX	Pemudahan (Penadahan, Pencetak dan Penerbit)	480-485	6
XXXI	Aturan Tentang Pengulangan Kejahatan yang Bersangkutan dengan Berbagai Bab	486-488	3

Buku Ketiga: Pelanggaran

Tabel 2.3 Tabel KUPH Buku Ketiga

Bab	Prihal	Pasal	Jmlh
I	Pelanggaran Keamanan Umum bagi Orang atau Barang dan Kesehatan	489-502	14
II	Pelanggaran Ketertiban Umum	503-520	18
III	Pelanggaran Terhadap Penguasa Umum	521-528	9
IV	Pelanggaran Mengenai Asal-Usul dan Perkawinan	529-530	2
V	Pelanggaran Terhadap Orang yang Memerlukan Pertolongan	531	1
VI	Pelanggaran Kesusilaan	532-547	16
VII	Pelanggaran Mengenai Tanah, Tanaman dan Pekarangan	548-551	4
VIII	Pelanggaran Jabatan	552-559	8
IX	Pelanggaran Pelayaran	560-569	10

2.2.3 Fungsi Hukum Pidana

Hukum pidana ditujukan untuk mengatur kepentingan-kepentingan umum, karena sifatnya yang ditujukan untuk kepentingan umum tersebut, maka fungsi hukum pidana adalah (Prasetyo, 2014):

1. Mengatur hidup kemasyarakatan.
2. Menyelenggarakan tata dalam masyarakat.

Disamping mengatur hidup kemasyarakatan, hukum pidana dapat digunakan sebagai sarana untuk menuju ke *policy* dalam bidang ekonomi, sosial, dan kebudayaan. Dan hukum harus menciptakan suasana masyarakat yang berlandaskan keadilan (Prasetyo, 2014).

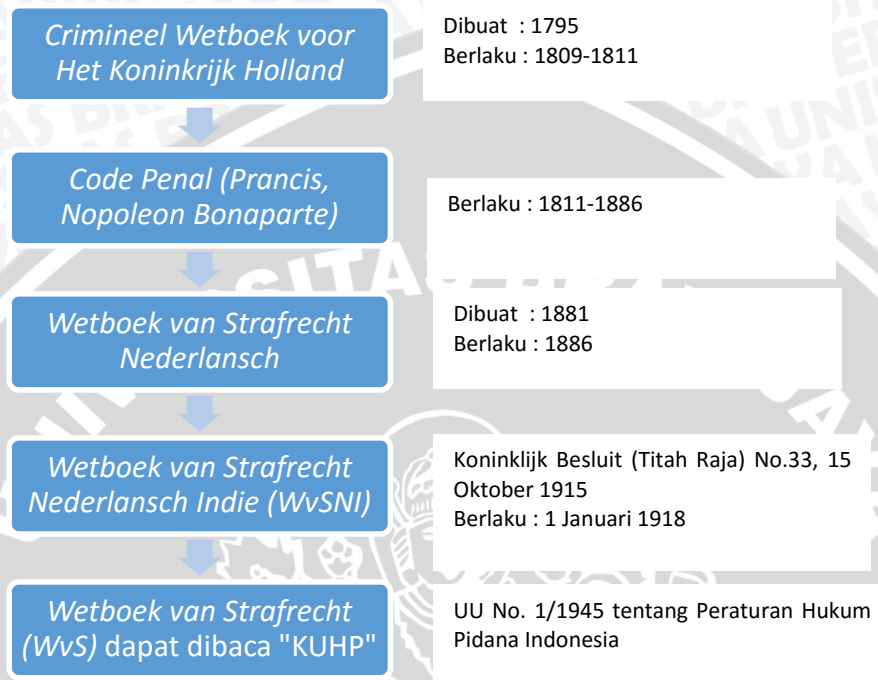
2.2.4 Sistem Hukuman

Sistem hukuman yang dicantumkan dalam Pasal 10 menyatakan bahwa hukuman yang dapat dikenakan kepada seseorang pelaku tindak pidana terdiri dari (Prasetyo, 2014):

1. Hukuman Pokok
 - a. Hukuman mati
 - b. Hukuman penjara
 - c. Hukuman kurungan
 - d. Hukuman denda

2. Hukuman Tambahan
 - a. Pencabutan beberapa hak tertentu
 - b. Perampasan barang-barang tertentu

2.2.5 Sejarah Pembentukan KUHP



Gambar 2.1 Sejarah KUHP

Sumber: Prasetyo (2014)

Sebagaimana diketahui bahwa kodifikasi di Indonesia banyak dipengaruhi oleh kodifikasi yang ada di Belanda. Di Belanda kodifikasi yang pertama terdapat pada tahun 1809 yang disebut dengan *Het Crimineel wet boek voor het koninkrijk Holand*. Kodifikasi pada tahun tersebut berlangsung lama oleh karena pada tahun 1811 sampai dengan tahun 1813 Belanda diduduki oleh Prancis sehingga yang diberlakukan *Code Penal* sampai pada tahun 1866. Sebenarnya sejak kodifikasi yang pertama selama 73 tahun Belanda sudah mempersiapkan rancangan peraturan hukum pidana yang selesai pada tahun 1881, dan diundangkan baru tanggal 1 September 1886. Dan sering disebut *Nederland Wet boek van Strafrecht*. Lalu dianut di Indonesia dan dinamakan KUHP pada tahun 1945 (Prasetyo, 2014).

2.3 Text Mining

2.3.1 Pengertian Text Mining

Text mining memiliki definisi menambang data yang berupa teks dimana sumber data biasanya didapatkan dari dokumen, dan tujuannya adalah mencari kata-kata yang dapat mewakili isi dari dokumen sehingga dapat dilakukan analisa keterhubungan antara dokumen.

Text mining secara luas didefinisikan sebagai proses mencari tahu secara intensif dimana pengguna berinteraksi dengan kumpulan dokumen sepanjang waktu dengan menggunakan serangkaian alat analisis.

Tujuan utama *text mining* adalah mendukung proses *knowledge discovery* pada koleksi dokumen yang besar. Pada prinsipnya, *text mining* adalah bidang ilmu multidisipliner, melibatkan *information retrieval (IR)*, *text analysis*, *information extraction (IE)*, *clustering*, *categorization*, *visualization*, *database technology*, *natural language processing (NLP)*, *machine learning* dan *data mining*. Dapat pula dikatakan bahwa teks mining merupakan salah satu bentuk aplikasi kecerdasan buatan *artificial intelligence (AI)*.

Text mining adalah sebuah ilmu yang bertujuan memberikan solusi dari permasalahan seperti pemrosesan, pengorganisasian atau pengelompokan dan menganalisa *unstructured text* dalam jumlah yang besar. Dalam memberikan solusi, *text mining* mengadopsi dan mengembangkan banyak teknik dari bidang lain, seperti *Data Mining*, *Information Retrieval*, statistik & matematika, *Machine Learning*, *Linguistic*, *Natural Language Processing*, dan *Visualization*. Kegiatan riset untuk *text mining* antara lain ekstraksi dan penyimpanan teks, *preprocessing* akan konten teks, pengumpulan data statistik dan *indexing* dan analisa konten.

Perbedaan mendasar antara teks *mining* dan *data mining* terletak pada sumber data yang digunakan. Pada *data mining*, pola-pola diekstrak dari *database* yang terstruktur, sedangkan di teks *mining*, pola-pola diekstrak dari data tekstual (*natural language*). Secara umum, *database* didesain untuk program dengan tujuan melakukan pemrosesan secara otomatis, sedangkan teks ditulis untuk dibaca langsung oleh manusia (Nafik, 2014).

2.3.2 Information Retrieval

Information Retrieval merupakan sebuah teknik pencarian dengan menggunakan algoritma tertentu untuk mendapatkan hasil pencarian yang relevan berdasarkan kumpulan informasi yang besar. Sebagian besar penggunaan sistem temu kembali adalah pada teks. Pengguna memasukkan kata kunci berupa teks, dan kemudian sistem mengolahnya hingga mendapatkan informasi semantik yang diinginkan oleh pengguna.

Temu Kembali Informasi (*information retrieval*) adalah ilmu pencarian informasi pada dokumen, pencarian untuk metadata yang menjelaskan dokumen, atau mencari dalam *database*, baik relasi *database stand-alone* atau *hypertext database* yang terdapat pada *network* seperti *internet* atau *World Wide Web* berupa data *text*, suara, atau gambar. *Information Retrieval (IR)* adalah ilmu yang lahir dari berbagai disiplin ilmu, baik ilmu komputer, matematika, ilmu kepastakaan, ilmu informasi, psikologi kognitif, linguistik maupun statistik.

Secara prinsip, penyimpanan informasi dan penemuan kembali informasi adalah hal yang sederhana. Misalkan terdapat tempat penyimpanan dokumen-dokumen dan seorang *user* mencari suatu informasi, dimana informasi yang dicari tersebut terdapat pada himpunan dokumen yang menyandung informasi yang

diperlukan oleh *user*. *User* bisa saja memperoleh informasi yang diperlukan dengan membaca semua dokumen dalam tempat penyimpanan, menyimpan informasi-informasi yang relevan dan membuang yang tidak relevan. Hal ini merupakan *perfect retrieval*, tetapi solusi ini tidak praktis. Karena *user* tidak memiliki waktu atau tidak ingin menghabiskan waktunya hanya untuk membaca seluruh himpunan dokumen.

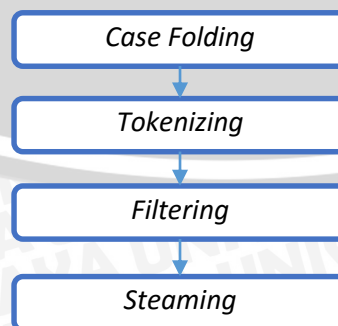
Information retrieval berbeda dengan *database retrieval*. Sistem *database retrieval* umumnya memberikan tepat semua data atau objek yang mempengaruhi kriteria tertentu. Sedangkan *information retrieval* merupakan suatu pendugaan atau dokumen-dokumen atau informasi yang diinginkan pengguna, dengan melihat tingkat kemiripannya. Sistem *database retrieval* dapat menerima *query* yang kompleks dan memberikan semua jawaban sesuai dengan kondisi logis dari *query* bersangkutan. Sedangkan sistem *IR* biasanya memberikan beberapa dokumen atau informasi yang telah diurutkan berdasarkan tingkat kemiripannya dengan *query* yang dimasukkan.

Information Retrieval System atau Sistem Temu Kembali Informasi akan memberikan informasi keberadaan dan keterangan informasi atau dokumen yang berhubungan dengan permintaan pengguna, dimana permintaan pengguna tersebut memakai bahasa natural sebagai bahasa *query*. *Information Retrieval System* memberikan kemudahan kepada pengguna dalam mempresentasikan atau menyampaikan kebutuhannya ke dalam sistem dalam bentuk *query* (Sutisna, 2010).

2.3.3 Text Processing

Suatu data dokumen teks, mempunyai struktur kata yang tidak teratur, maka dalam melakukan proses *text mining* diperlukan beberapa proses tambahan yang bertujuan untuk menyiapkan dan mengubah data teks mentah menjadi lebih terstruktur. Salah satu implementasi dari *text mining* adalah tahap *preprocessing text*. Pada tahap *preprocessing text* ini dilakukan proses seleksi data berupa text pada setiap dokumen, yang kemudian akan dilakukan proses selanjutnya.

Pada tahap *preprocessing* ini melalui beberapa tahapan yaitu *case folding*, *tokenizing*, *filtering*, dan *stemming*. Berikut merupakan tahap dari *preprocessing* yang dapat dilihat pada Gambar 2.2.



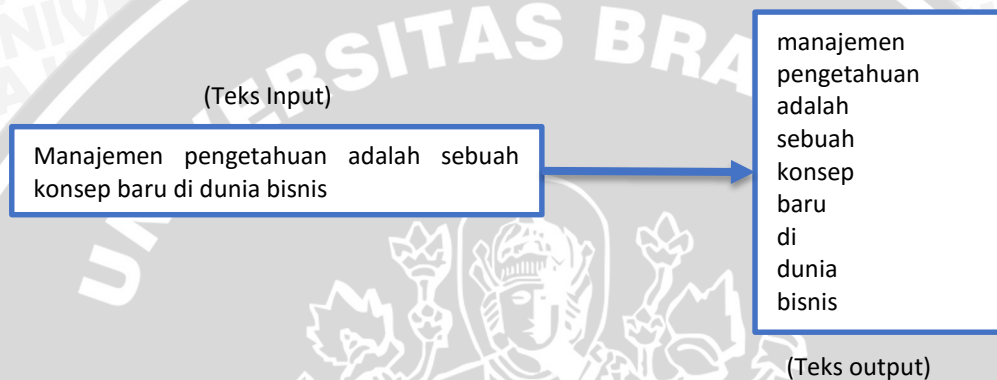
Gambar 2.2 Tahapan *preprocessing*

Sumber: Nugroho (2011)

2.3.3.1 Case Folding dan Tokenizing

Suatu dokumen teks mengandung beragam huruf dan angka. Dalam penulisan teks suatu kalimat dapat terkandung huruf kapital dan huruf kecil yang dapat dipadukan dengan suatu angka maupun delemeter. Oleh karena itu, peran *case folding* dibutuhkan untuk melakukan konversi seluruh teks dalam dokumen menjadi huruf kecil (*lowercase*). Dalam proses *case folding* hanya huruf 'a' sampai dengan 'z' yang diterima, kemudian huruf kapital akan dikonversi menjadi huruf kecil. Karakter selain huruf dihilangkan dan dianggap delimitter.

Tahap *tokenizing / parsing* adalah tahapan dimana dilakukan pemotongan *string input* berdasarkan tiap kata penyusunnya. Contoh dari tahap *case folding* dan *tokenizing* dapat dilihat pada gambar 2.3.



Gambar 2.3 Tahap case folding dan tokenizing

Sumber: Nugroho (2011)

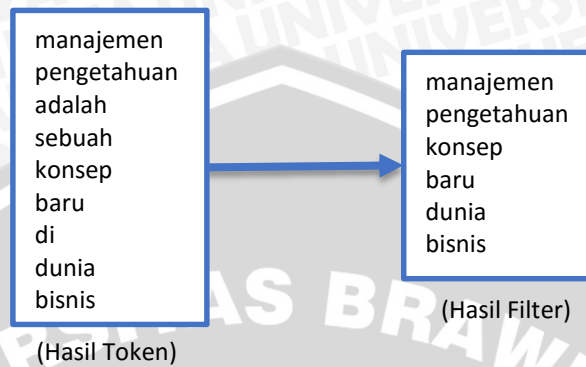
Secara garis besar tokenisasi merupakan proses pemecahan sekumpulan karakter atau kata dalam suatu teks kedalam satuan kata. Terdapat beberapa karakter-karakter tertentu yang dapat diperlakukan atau digunakan sebagai pemisah kata, yaitu karakter *white space*, seperti *enter*, *tabulasi*, *spasi*. Namun untuk karakter petik tunggal ('), titik (.), semicolon (;), titik dua (:) atau lainnya, dapat memiliki peran yang cukup banyak sebagai pemisah kata. Dalam memperlakukan karakter-karakter dalam teks sangat tergantung sekali pada kontek aplikasi yang dikembangkan. Pekerjaan tokenisasi ini akan semakin sulit jika harus memperhatikan struktur bahasa (gramatikal) (Nafik, 2014).

2.3.3.2 Filtering

Filtering merupakan tahap dalam *preprocessing* dimana pada tahap ini melakukan pengambilan kata-kata penting dari hasil tokenisasi. Proses *filtrng* dapat menggunakan *stoptlist*, yaitu membuang kata yang kurang penting. Atau dapat menggunakan *wordlist*, yaitu menyimpan kata-kata penting yang terdapat pada suatu teks dokumen. *Stoptlist / stopword* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan *bag-of-words*.

Contoh kata-kata yang termasuk dalam *stopword* diantaranya yaitu kata "yang", "dan", "di", "ke", "dari" dan lainnya. Kata-kata yang terdapat dalam *stoptlist* merupakan kata-kata yang mempunyai frekuensi tinggi kemunculannya

dalam suatu dokumen teks dan dapat ditemukan hampir dalam setiap dokumen. Penghilangan kata menggunakan *stopword* ini dapat mengurangi ukuran *index* dan waktu dalam melakukan pemrosesan maupun perhitungannya. Selain itu, juga dapat mengurangi level *noise*. Berikut merupakan gambaran dari proses *filtering* yang dapat dilihat pada Gambar 2.4.



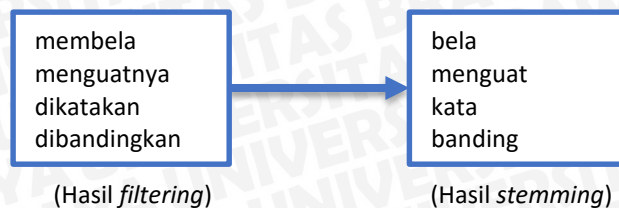
Gambar 2.4 Tahap *filtering*

Sumber: Nugroho (2011)

Namun menggunakan *stoplist* tidak selalu meningkatkan nilai *retrieval*. Penyusunan daftar kata-kata dalam *stoplist* yang tidak sesuai dengan sistem yang dibangun dapat memperburuk kinerja dari sistem *information retrieval*. Belum ada suatu kesimpulan pasti bahwa penggunaan *stoplist* akan selalu meningkatkan nilai *retrieval*, karena dari beberapa penelitian yang telah dilakukan sebelumnya, hasil yang didapatkan cenderung bervariasi (Mahendra, 2008).

2.3.3.3 *Stemming*

Pembuatan indeks dilakukan karena suatu dokumen tidak dapat dikenali langsung oleh suatu sistem temu kembali informasi atau *information retrieval (IR) system*. Oleh karena itu, dokumen tersebut terlebih dahulu perlu dipetakan ke dalam suatu representasi dengan menggunakan teks yang berada di dalam dokumen itu sendiri. Teknik *stemming* diperlukan selain untuk memperkecil jumlah index yang berbeda dari suatu dokumen, juga untuk melakukan pengelompokan kata-kata lain yang memiliki kata dasar dan arti yang serupa, namun memiliki bentuk atau *form* yang berbeda karena mendapatkan imbuhan yang berbeda. Sebagai contoh kata “berjalan”, “menjalankan”, “perjalanan”, akan dilakukan *stemming* ke *root word-nya* yaitu menjadi kata “jalan”. Contoh tahapan *stemming* ini adalah seperti pada gambar 2.5.



Gambar 2.5 Proses stemming

Sumber: Nugroho (2011)

Namun seperti halnya *stoplist*, kinerja *stemming* juga bervariasi dan sering tergantung pada domain bahasa yang digunakan. Proses *stemming* pada teks berbahasa Indonesia berbeda dengan *stemming* pada teks berbahasa Inggris. Pada teks berbahasa Inggris, proses yang diperlukan hanya proses menghilangkan *surfiks* (akhiran). Sedangkan pada teks berbahasa Indonesia, diperlukan bukan hanya proses menghilangkan *surfiks*, tetapi juga membutuhkan proses menghilangkan *prefiks* (awalan) dan *konfiks* (perpaduan awalan dengan akhiran yang membentuk satu kesatuan) (Agusta, 2009).

2.3.4 Apache Lucene

Apache Lucene mempunyai kinerja yang tinggi, *library* mesin pencari teks yang mempunyai fitur lengkap dan seluruhnya ditulis di pemrograman *Java*, salah satunya adalah melakukan *stemming* terhadap suatu kata maupun kalimat dengan baik. Ini adalah teknologi yang cocok untuk hampir semua aplikasi yang memerlukan pencarian teks lengkap, terutama lintas-*platform* (Apache, 2015).

Lucene adalah *library* untuk *Information Retrieval (IR)* yang berkemampuan tinggi dalam pengindeksan dan pencarian. Lucene bukan aplikasi siap pakai (*ready-to-use application*) seperti program *file-search*, *web crawler*, atau *search engine website*. Tetapi merupakan *Application Programming Interface (API)* sederhana yang menangani pengindeksan dan pencarian berbasis *full-text*. Untuk mengintegrasikan *Lucene* dengan aplikasi yang akan dibangun hanya perlu memahami perintah dan aturan *class-class* yang tersedia pada *library Lucene* yang dibangun dalam bahasa pemrograman *Java*. Dalam *information retrieval (IR)*, Lucene dapat berperan dalam *filtering* dan *stemming*. Kelebihan Lucene adalah faktor kemudahan yang ditawarkan karena dibangun dalam bahasa *Java* sehingga integrasi dengan aplikasi yang akan dibangun lebih mudah diatur dan diawasi (Sofyan, 2009).

2.4 TF-IDF

Metode *TF-IDF* merupakan metode pembobotan *term* yang banyak digunakan sebagai metode pembandingan terhadap metode pembobotan baru. Pada metode ini, perhitungan bobot *term* dalam sebuah dokumen dilakukan dengan mengalikan nilai *Term Frequency* dengan *Inverse Document Frequency*.

Term Frequency (TF) adalah faktor yang menentukan bobot *term* pada suatu dokumen berdasarkan jumlah kemunculannya dalam dokumen tersebut. Nilai

jumlah kemunculan suatu kata (*term frequency*) diperhitungkan dalam pemberian bobot terhadap suatu kata. Semakin besar jumlah kemunculan suatu *term* dalam dokumen, semakin besar pula bobotnya dalam dokumen atau akan memberikan nilai kesesuaian yang semakin besar. Persamaan 2.1 digunakan untuk menghitung bobot *term frequency* (*tf weight*).

$$w_{tf_{t,d}} = \begin{cases} 1 + \log_{10} tf_{t,d}, & \text{if } tf_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases} \quad (2.1)$$

$w_{tf_{t,d}}$ = bobot *term frequency* (*tf weight*)

$tf_{t,d}$ = banyaknya kemunculan *term*/kata *t* dalam dokumen *d*

Inverse Document Frequency (*IDF*) adalah pengurangan dominansi *term* yang sering muncul di berbagai dokumen. Hal ini diperlukan karena *term* yang banyak muncul di berbagai dokumen, dapat dianggap sebagai *term* umum (*common term*) sehingga tidak penting nilainya. Sebaliknya faktor kejarangmunculan kata (*term scarcity*) dalam koleksi dokumen harus diperhatikan dalam pemberian bobot. Kata yang muncul pada sedikit dokumen harus dipandang sebagai kata yang lebih penting (*uncommon term*) daripada kata yang muncul pada banyak dokumen. Pembobotan akan memperhitungkan faktor kebalikan frekuensi dokumen yang mengandung suatu kata (*inverse document frequency*), dihitung menggunakan persamaan 2.2 (Zafikri, 2008).

$$idf_t = \log_{10} N/df_t \quad (2.2)$$

idf_t = bobot *inverse document frequency*

N = banyaknya dokumen yang ada

df_t = banyaknya dokumen yang mengandung *term*/ kata *t*

Kemudian rumus untuk menghitung bobot *tf-idf* untuk setiap kata *t* dalam dokumen *d*, merupakan hasil perkalian antara *tf weight* dengan *idf* dihitung menggunakan persamaan 2.3 (Schutze, 2011):

$$w_{t,d} = w_{tf_{t,d}} \times idf_t \quad (2.3)$$

$w_{t,d}$ = bobot *tf-idf* kata *t* dalam dokumen *d*

$w_{tf_{t,d}}$ = bobot *term frequency* (*tf weight*)

idf_t = bobot *inverse document frequency*

2.5 Vector Space Model

Vector Space Model (*VSM*) adalah cara konvensional yang biasa digunakan dalam proses temu kembali informasi. Prosesnya dengan menghitung kemiripan dua buah vektor, yaitu antara vektor dari *corpus* dan vektor dari *query* (Kontostathis, 2007). Untuk melakukan perhitungan terhadap kemiripan antar vektor digunakan rumus *Cosine Similarity* (Parsons, 2009).

2.5.1 Cosine Similarity

Metode *cosine similarity* adalah metode yang digunakan untuk menghitung kesamaan atau kedekatan antar dua dokumen. *Cosine similarity* dapat menyamakan frekuensi kata pada kalimat menggunakan persamaan *Term Frequency (TF)*. *Term frequency* mengekstrak dokumen menjadi proses yang terdiri dari kumpulan kata perkalimat. Tujuannya adalah menyamakan kedua kalimat pada suatu dokumen yang nantinya akan dibandingkan, sehingga dapat melangkah ke tahap selanjutnya yaitu tahap *similarity* (Wati, 2012).

Metode *cosine similarity* digunakan untuk menemukan kesamaan atau kedekatan antar dua dokumen. Semakin besar nilai kesamaan *vector query* dengan *vector* dokumen maka *query* tersebut dipandang semakin relevan dengan dokumen. *Cosine similarity* merupakan dasar perhitungan untuk mendapatkan nilai relevansi antara *query* dengan dokumen dan relevansi antara dokumen (Yulita, 2015). Persamaan 2.4 digunakan untuk menghitung *cosine similarity*.

$$\text{CosSim}(d_i, q) = \frac{d_i \cdot q}{|d_i| |q|} \quad (2.4)$$

Keterangan :

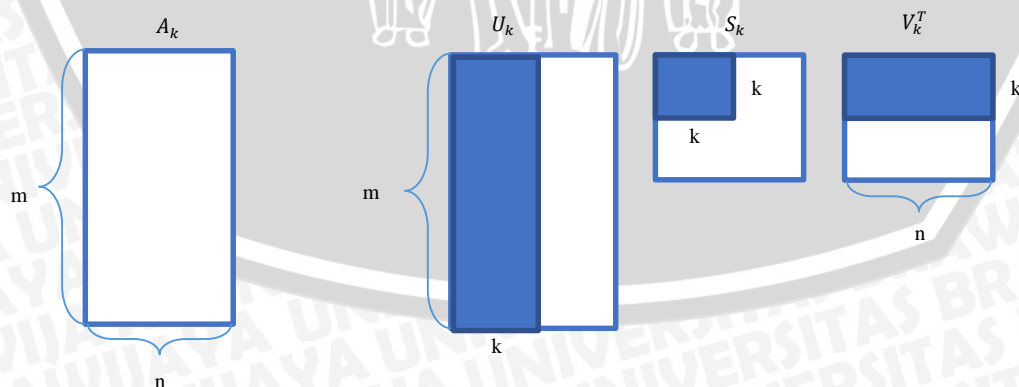
d_i = dokumen *vector* ke i

q = kata kunci/*query vector*

Kedekatan *query* dengan dokumen diindikasikan dengan sudut yang dibentuk. Nilai *cosinus* yang cenderung besar mengindikasikan bahwa dokumen cenderung sesuai *query* (Bunyamin, 2005).

2.6 Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) merupakan model matematis yang digunakan untuk reduksi dimensi data. Proses *SVD* dilakukan dengan mendekomposisi matriks menjadi tiga bagian (Peter, 2009), seperti pada Gambar 2.6



Gambar 2.6 Ilustrasi dekomposisi matriks SVD

Sumber: Peter (2009)

Matriks U dan V adalah matriks *orthonormal*, dimana baris pada matriks U menggambarkan banyaknya baris pada matriks A , sementara kolom pada matriks V menggambarkan banyaknya kolom pada matriks A . K -rank digunakan untuk mereduksi dimensi dari matriks A . Matriks S merupakan matriks simetris yang berisi nilai positif di sepanjang diagonal, daerah selain diagonal berisi 0 (Sari, 2012). Representasi dari SVD dapat dilihat pada persamaan 2.5.

$$A = U S V^T \tag{2.5}$$

Keterangan :

U = matriks *orthonormal eigenvectors* dari AA^T

V = matriks *orthonormal eigenvectors* dari $A^T A$

S = matriks diagonal yang berisi akar kuadrat dari *eigenvalue* dari U atau V yang diurutkan secara *descending* urutan (*singular value*)

Dimana $U^T U = I$, $V^T V = I$, kolom dari U adalah *orthonormal eigenvectors* dari AA^T , kolom V adalah *orthonormal eigenvector* dari $A^T A$, dan S adalah matriks diagonal yang berisi akar kuadrat dari *eigenvalue* dari U atau V yang diurutkan secara *descending* urutan (Baker, 2013).

2.7 Eigenvector dan Eigenvalue

Eigenvectors adalah vektor tak nol yang memenuhi persamaan (Anton, 2010):

$$Ax = \lambda x \tag{2.6}$$

Dapat ditulis dengan :

$$(A - \lambda I)x = 0 \tag{2.7}$$

Keterangan:

A = matriks $n \times n$

x = *eigenvectors* (vektor kolom $n \times 1$)

λ = *eigenvalue* dari matriks A

I = matriks identitas

Menemukan *eigenvalues* dan *eigenvectors* dapat dilakukan dengan memperlakukan matriks sebagai sistem persamaan linear dan memecahkan untuk nilai-nilai variabel yang membentuk komponen *eigenvector* tersebut. Untuk mencari *eigenvalue* digunakan polinomial karakteristik dan persamaan karakteristik dari matriks A . Pertama-tama akan dihitung polinomial karakteristik dari matriks A , menggunakan persamaan 2.8 (Anton, 2010).

$$\det(A - \lambda I) = 0 \tag{2.8}$$

Eigenvalue (λ) adalah nilai karakteristik dari suatu matriks berukuran $n \times n$, sementara *eigenvector* (x) adalah vektor kolom bukan nol yang bila dikalikan dengan suatu matriks berukuran $n \times n$ akan menghasilkan vektor lain yang memiliki nilai kelipatan dari *eigenvector* itu sendiri (Anton, 2010).

Persamaan karakteristik dari matriks A adalah persamaan dengan variabel λ yang digunakan untuk perhitungan nilai dan *eigenvector*. Perhitungan dimulai dengan mencari *eigenvalue*, kemudian dengan *eigenvalue* yang telah diperoleh, akan dihitung *eigenvector* untuk masing – masing nilai yang memenuhi persamaan. *Eigenvector* untuk masing - masing *eigenvalue* kemudian dapat ditentukan dengan melakukan operasi baris elementer atau teknik eliminasi sistem persamaan linear lainnya (Anton, 2010).

2.8 Latent Semantic Indexing (LSI)

Reduksi dari SVD digunakan dalam LSI. LSI merupakan salah satu bentuk teknik proses temu kembali dengan menggunakan *Vector Space Model (VSM)*, untuk menemukan informasi yang relevan. Keterkaitan makna dalam LSI sifatnya tersembunyi. Fungsi matematis di dalam LSI mampu menemukan hubungan semantik antar kata (Sari, 2012). Juga mendapatkan suatu pemodelan yang efektif untuk merepresentasikan hubungan antara kata kunci dan dokumen yang dicari (Ferdian, 2013). Representasi dari LSI dapat dilihat pada persamaan 2.9.

$$q' = q^T \cdot U_k \cdot S_k^{-1} \quad (2.9)$$

Keterangan :

q' = *query vector* representasi dari LSI

q^T = *transpose* dari pembobotan ternormalisasi TF-IDF *query*

U_k = reduksi dimensi k dari matriks U

S_k^{-1} = inverse dari reduksi dimensi k matriks S

Dimana q' adalah *query vector* representasi dari LSI, q^T adalah *transpose* dari pembobotan TF-IDF *query*, U_k adalah reduksi dimensi k dari matriks U, dan S_k^{-1} adalah inverse dari reduksi dimensi k matriks S (Sari, 2012).

2.9 Precision dan Recall

Precision dan *recall* merupakan kumpulan evaluasi untuk mengetahui keakuratan sistem temu kembali secara unranked retrieval, atau pengembalian dokumen tanpa perankingan. *Precision* dan *recall* dihitung hanya dengan melihat keseluruhan rekomendasi untuk masing-masing dokumen tanpa memperhatikan urutan dokumen rekomendasi yang diberikan oleh sistem (Sari, 2012).

Precision merupakan perbandingan jumlah informasi relevan yang didapatkan sistem dengan jumlah seluruh informasi yang terambil oleh sistem baik yang relevan maupun tidak (Nedunchelian, 2011). Persamaan *precision* ditunjukkan pada persamaan 2.10 :

$$Precision = \frac{true\ positive}{true\ positive + false\ positive} \quad (2.10)$$

Recall merupakan perbandingan jumlah informasi relevan yang didapatkan sistem dengan jumlah seluruh informasi relevan yang ada dalam koleksi informasi (baik yang terambil atau tidak terambil oleh sistem) (Nedunchelian, 2011). Persamaan precision ditunjukkan pada persamaan 2.11:

$$Recall = \frac{true\ positive}{true\ positive + false\ negative} \quad (2.11)$$

Keterangan:

True positive : jumlah data yang diekstrak oleh sistem dan manusia.

False positive : jumlah data yang diekstrak oleh sistem tetapi tidak diekstrak oleh manusia.

False negative : jumlah kalimat yang diekstrak oleh manusia tetapi tidak diekstrak oleh sistem.

2.10 Mean Average Precision (MAP)

Tipe evaluasi yang digunakan untuk mengevaluasi sistem temu kembali dengan *ranked retrieval* yaitu menggunakan *Mean Average Precision (MAP)*. Mean Average Precision akan menghitung nilai *precision* masing-masing dokumen yang direkomendasikan oleh sistem dengan memperhatikan urutan dokumen rekomendasi yang diberikan oleh sistem. Dalam konteks sistem temu kembali, dokumen yang dikembalikan dengan memasukkan *top-k* dokumen yang *retrieved*. *Average Precision (AP)* hanya mengambil nilai presisi dari dokumen-dokumen yang relevan dan kemudian hasilnya dibagi dengan jumlah dokumen yang dilibatkan. Pengukuran dari MAP merupakan hasil perhitungan rata-rata dokumen relevan yang *retrieved* dari setiap *query* yang terlibat di dalam sistem, sedangkan dokumen yang tidak relevan nilainya adalah 0 (Sari, 2012). Nilai MAP mempunyai rentang nilai 0 sampai 1, dan dalam sebuah sistem dikatakan baik jika nilai MAP mendekati 1 (Manning, 2009).

Rumus *Mean Average Precision (MAP)* dapat dilihat pada persamaan 2.12 (Manning, 2009):

$$MAP(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m} \sum_{k=1}^{m_j} Precision(R_{jk}) \quad (2.12)$$

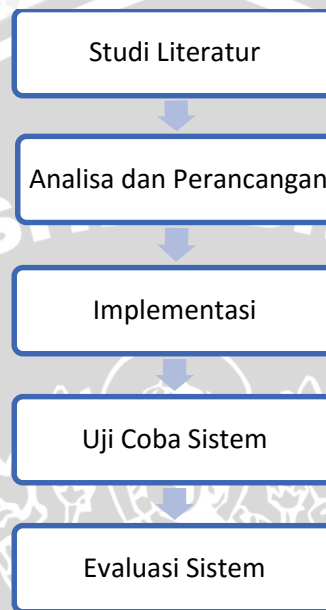
Keterangan:

Q = banyak *query* yang diinputkan

R_{jk} = nilai *precision* dari kumpulan dokumen relevan yang telah di-*ranking*

BAB 3 METODOLOGI DAN PERANCANGAN

Pada bab ini menjelaskan tentang langkah-langkah yang akan dilakukan dalam pengerjaan tugas akhir, yaitu studi literatur, penyusunan dasar teori, analisa dan perancangan, uji coba sistem, dan yang terakhir adalah evaluasi sistem. Berikut ini merupakan gambaran dari tahapan penelitian secara umum yang dapat dilihat pada gambar 3.1.



Gambar 3.1 Gambaran Umum Tahap Penelitian

Penjelasan mengenai diagram alir pada gambar 3.1. diatas adalah:

1. Mempelajari konsep.
2. Melakukan analisa dan merancang sistem untuk melakukan pemilihan terhadap pasal-pasal KUHP secara otomatis berdasarkan inputan berupa uraian kasus kejahatan yang dimasukkan oleh pengguna.
3. Melakukan implementasi sistem berdasarkan analisa perancangan yang sudah dilakukan sebelumnya.
4. Melakukan uji coba terhadap sistem yang dibuat dengan menganalisa hasil dari sistem. Hasil yang dikeluarkan oleh sistem berupa pasal-pasal KUHP yang berkaitan atau yang dikenakan pada suatu uraian kasus yang dimasukkan oleh pengguna.
5. Melakukan evaluasi terhadap hasil uji coba yang telah dilakukan sebelumnya.

3.1 Studi Literatur

Studi literatur mempelajari dasar teori dan tahapan-tahapan yang harus dilakukan dan digunakan untuk menunjang penulisan skripsi. Studi literatur menjadi referensi teori dan acuan pemecahan masalah yang relevan dengan kasus yang diangkat atau permasalahan yang ditemukan. Teori-teori dan referensi

pendukung tersebut dapat diperoleh dari buku, jurnal, artikel, *e-book*, situs-situs di internet, dan penelitian sebelumnya yang berkaitan tentang topik skripsi ini. Referensi utama yang diperlukan untuk menunjang penulisan skripsi ini adalah *Preprocessing (case folding, tokenizing, filtering, dan stemming)*, *Cosine Similarity*, *Singular Value Decomposition (SVD)*, dan *Latent Semantic Indexing (LSI)*.

3.2 Analisa Kebutuhan

Analisa kebutuhan merupakan tahapan dimana dilakukan analisa terhadap spesifikasi yang dibutuhkan dan diinginkan oleh pengguna mengenai suatu sistem yang akan dibangun. Hasil dari analisa kebutuhan pengguna ini digunakan sebagai kerangka informasi untuk mengembangkan sebuah sistem yang sesuai dengan harapan pengguna.

3.2.1 Deskripsi Umum Sistem

Sistem ini akan melakukan pencarian terhadap pasal-pasal yang terdapat pada Kitab Undang-Undang Hukum Pidana (KUHP). Pencarian dilakukan berdasarkan uraian kasus kejahatan yang dimasukkan oleh pengguna kedalam sistem. Sistem akan melakukan *preprocessing* terhadap semua data pasal KUHP yang terdapat dalam *database* beserta *query* atau uraian kasus dari pengguna. Kemudian sistem akan menghitung pembobotan *TF-IDF*, yang dilanjutkan dengan reduksi data menggunakan *Singular Value Decomposition (SVD)*. Lalu dilakukan perhitungan *Latent Semantic Indexing (LSI)* untuk menemukan informasi yang relevan dan untuk menemukan hubungan semantik antar kata. Dan mencari kemiripan antar *corpus* dan *query* dengan *cosine similarity*. Setelah dilakukan beberapa proses perhitungan maka akan didapatkan pasal-pasal yang berkaitan dengan uraian kasus kejahatan yang dimasukkan oleh pengguna.

3.2.2 Kebutuhan Data

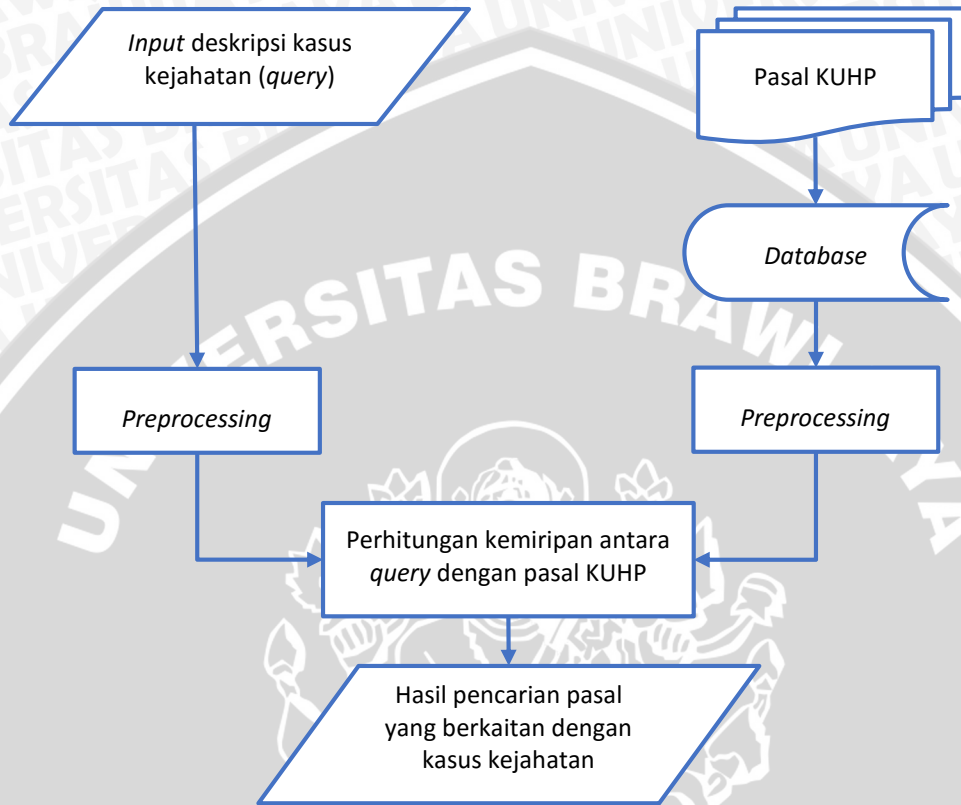
Data yang digunakan dan diolah oleh sistem ini adalah:

1. Data training berupa pasal-pasal dari Kitab Undang-Undang Hukum Pidana khususnya pada buku kedua tentang kejahatan.
2. Data uraian kasus kejahatan yang dimasukkan oleh pengguna ke dalam sistem.

3.3 Perancangan Sistem

Pada perancangan sistem ini akan dijelaskan mengenai rancangan proses yang akan dilakukan oleh sistem agar sesuai dengan harapan pengguna. Pada sistem ini telah diberikan data pada *database* berupa kumpulan pasal KUHP. Pengguna akan memasukkan suatu *query* berupa uraian mengenai kasus kejahatan ke dalam sistem. Kemudian sistem akan melakukan *preprocessing* terhadap semua data pasal KUHP yang terdapat dalam *database* beserta *query* atau uraian kasus dari pengguna. Sistem akan melakukan penghitungan bobot *TF-IDF*, yang dilanjutkan dengan reduksi data menggunakan *Singular Value Decomposition (SVD)*, lalu dilakukan perhitungan *Latent Semantic Indexing (LSI)* untuk menemukan informasi yang relevan dan untuk menemukan hubungan semantik antar kata. Dan mencari

kemiripan antar *corpus* dan *query* dengan *cosine similarity*. Setelah dilakukan beberapa proses perhitungan maka akan didapatkan pasal-pasal yang berkaitan dengan uraian kasus kejahatan yang dimasukkan oleh pengguna. Berikut merupakan gambaran alur kerja sistem secara umum digambarkan pada Gambar 3.2.



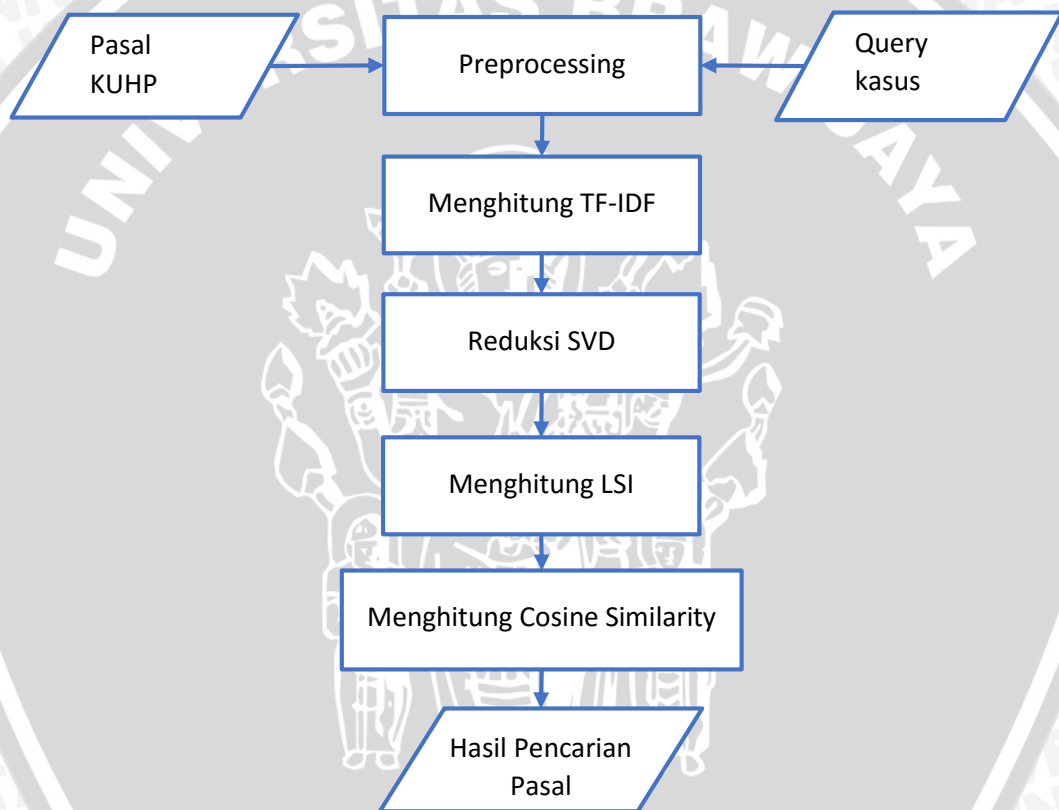
Gambar 3.2 Alur kerja sistem secara umum

Preprocessing merupakan tahapan dimana aplikasi melakukan seleksi data berupa kata yang berfungsi untuk mengurangi *noise* sehingga nantinya akan menghasilkan kumpulan kata-kata penting dalam bentuk kata dasar untuk mempermudah proses perhitungan selanjutnya. Tahap *preprocessing* ini dilakukan pada setiap *query* atau *input* deskripsi kasus kejahatan yang dimasukkan oleh *user*, juga pada setiap pasal KUHP yang terdapat dalam *database*. Tahap *preprocessing* ini meliputi *case folding*, *tokenizing*, *filtering*, dan *stemming* yang telah dijelaskan sebelumnya pada bab 2. Detail dari proses *preprocessing* akan digambarkan pada gambar 3.3.



Gambar 3.3 Skema Aliran Data Preprocessing

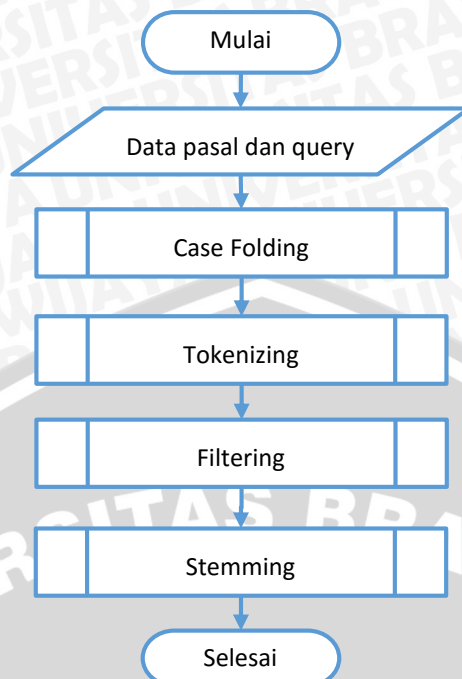
Kemudian kata-kata hasil *preprocessing* akan dilakukan perhitungan untuk menentukan pasal mana saja yang berkaitan dengan *query* atau *input* deskripsi kasus kejahatan yang dimasukkan oleh *user*. Pada proses perhitungan, pertamanya menghitung bobot *TF-IDF* pada *input query* kasus dan dokumen pasal. Kemudian menggunakan *singular value decomposition* (SVD) untuk mereduksi dimensi dari hasil *preprocessing*. Kemudian dilanjutkan dengan menghitung *query vector* dengan algoritma *latent semantic indexing* (LSI). Dilanjutkan dengan menghitung kemiripan antar dokumen pasal dengan *query* dengan *cosine similarity* yang kemudian hasil perhitungannya diurutkan secara *descending* dan juga mengambil *top-n* teratas nilai hasil *cosine similarity* dari hasil pengurutan. Pada gambar 3.4 merupakan gambaran dari proses perhitungan pencarian pasal KUHP yang dilakukan oleh sistem.



Gambar 3.4 Skema aliran data perhitungan

3.3.1 Preprocessing

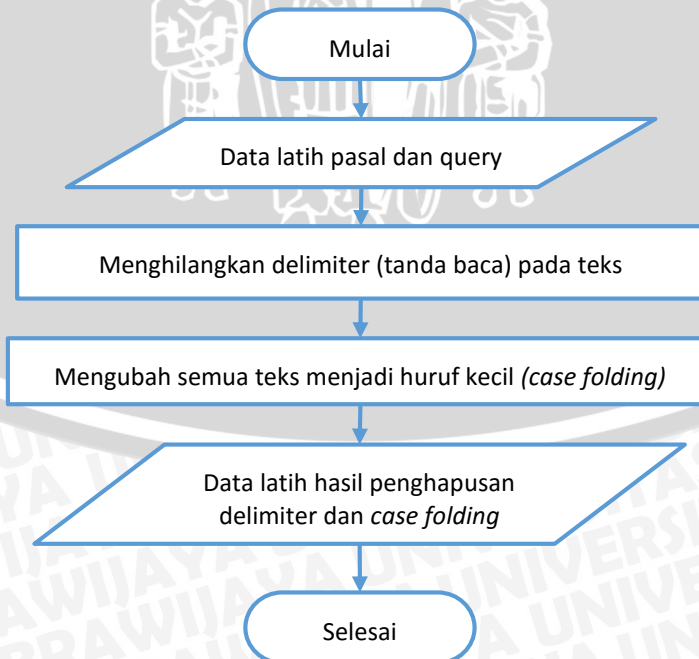
Preprocessing merupakan tahapan dimana suatu sistem melakukan seleksi terhadap data berupa kata yang akan diproses pada setiap dokumen. Proses ini bertujuan untuk mengurangi *noise* atau kata yang tidak penting hingga mendapatkan kata dasar dari suatu kata pada masing-masing dokumen. Sehingga proses penghitungan di tahap selanjutnya menjadi lebih valid. Tahapan yang dilakukan pada *preprocessing* ini yaitu *case folding*, *tokenizing*, *filtering*, dan *stemming*. Pada penelitian ini, proses *filtering* dan *stemming* menggunakan *library Apache Lucene*. Berikut merupakan tahapan *preprocessing* pada gambar 3.5.



Gambar 3.5 Flowchart preprocessing

3.3.1.1 Case Folding

Tahap pertama dalam *preprocessing* yaitu *case folding*. Proses ini digunakan untuk melakukan konversi atau mengubah seluruh huruf pada teks dokumen pasal maupun *query* menjadi huruf kecil (*lower case*), hanya huruf ‘a’ sampai dengan ‘z’ saja yang diterima. Karakter selain huruf yaitu tanda baca atau delimiter akan dihilangkan pada proses *case folding* ini. Berikut merupakan tahapan *case folding* pada gambar 3.6.

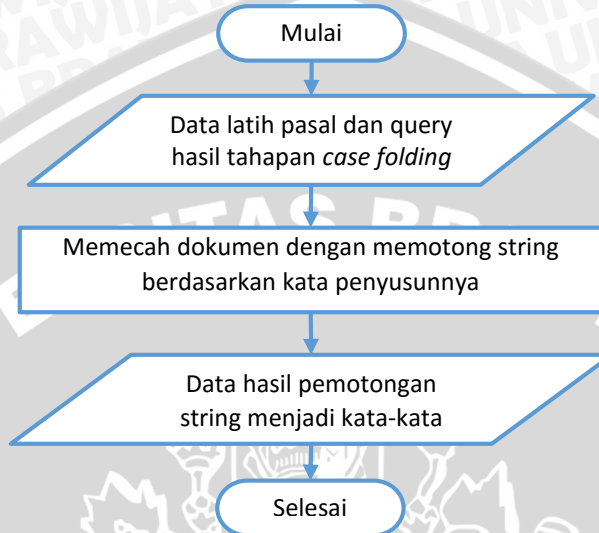


Gambar 3.6 Flowchart case folding



3.3.1.2 Tokenizing

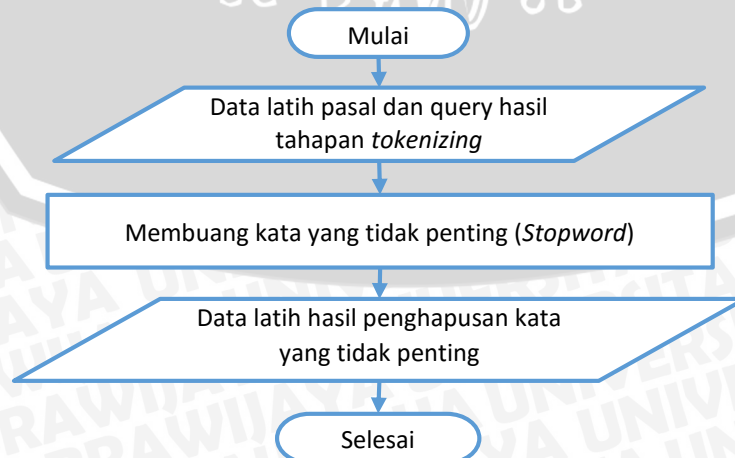
Tokenizing merupakan proses pemotongan string pada suatu dokumen berdasarkan tiap kata yang menyusunnya serta membedakan karakter-karakter tertentu yang dapat diperlakukan sebagai pemisah kata atau bukan. Sehingga proses ini dilakukan pemecahan data latih berupa kalimat menjadi kata-kata. Tahapan ini dilakukan setelah data latih pasal melewati tahap *Case Folding*. Berikut merupakan tahapan *tokenizing* pada gambar 3.7.



Gambar 3.7 Flowchart tokenizing

3.3.1.3 Filtering

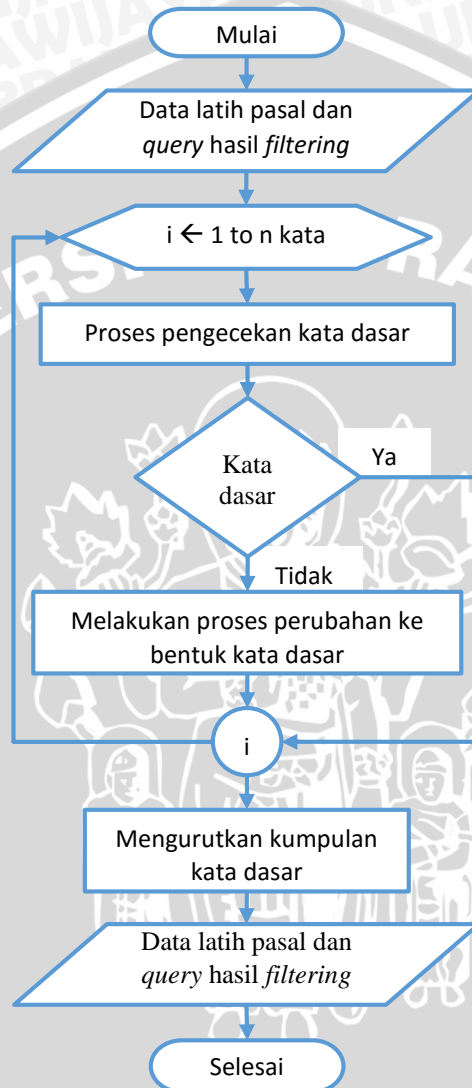
Filtering merupakan proses penghapusan kata-kata yang tidak penting pada data latih. Algoritma yang digunakan pada proses *filtering* ini adalah algoritma *stopword*. *Stopword* merupakan suatu array yang berisi data berupa kata-kata yang tidak memiliki arti. Kata-kata hasil pemrosesan sebelumnya yaitu *tokenizing*, akan dicocokkan dengan tabel pada *stopword*. Jika kata yang diperiksa sama dengan kata yang ada dalam daftar *stopword*, maka kata tersebut akan dihapus. Sebaliknya jika kata yang diperiksa tidak ada dalam *stopword*, maka kata tersebut dijadikan kata penting (*keyword*) dan akan dilakukan proses *stemming* untuk mendapatkan kata dasar. Berikut merupakan tahapan *filtering* pada gambar 3.8.



Gambar 3.8 Flowchart filtering

3.3.1.4 Stemming

Stemming merupakan tahapan dimana dilakukan pencarian terhadap kata dasar dari tiap kata hasil *filtering*. *Stemming* mengubah berbagai bentuk kata ke dalam suatu bentuk kata dasar yang sama. Tahap ini dilakukan pemotongan atau penghilangan imbuhan seperti *prefix* (awalan) dan *suffix* (akhiran) serta *confix* (awalan dan akhiran). Berikut merupakan tahapan *stemming* pada gambar 3.9.



Gambar 3.9 Flowchart stemming

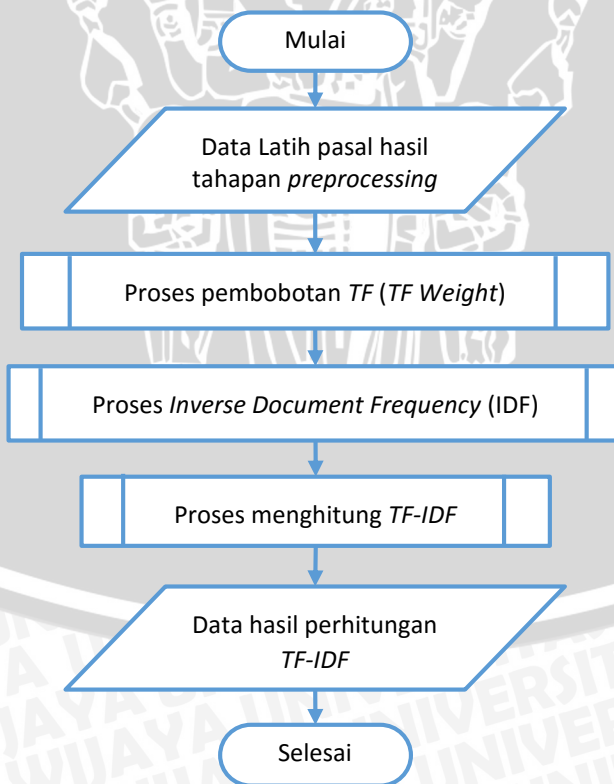
3.3.2 TF-IDF

Langkah pertama dalam menghitung bobot dari *tf-idf*, yaitu dengan menghitung banyaknya kemunculan suatu kata/*term/token* dalam suatu dokumen pasal tertentu ($tf_{t,d}$). Kemudian membuat *inverted index* dari $tf_{t,d}$ yang merupakan suatu struktur data berbentuk matriks, yang digunakan untuk mempermudah dalam merepresentasikan banyaknya kata yang muncul dalam dokumen atau pasal tertentu ($tf_{t,d}$). Contoh penerapan *inverted index* yaitu pada tabel 3.1.

Tabel 3.1 Tabel inverted index

Kata	TF(i,j)		
	Pasal 340	Pasal 346	Pasal 362
ambil	0	1	0
barang	0	2	0
bunuh	2	0	0
gugur	0	0	1
kandung	0	0	1
milik	0	2	0
rencana	2	0	0
wanita	0	0	1
sengaja	1	0	0

Langkah selanjutnya adalah menghitung bobot dari *tf* (*tf wight*) kemudian diikuti dengan menghitung nilai dari *Inverse Document Frequency* (*idf*). Setelah nilai dari *tf wight* dan *idf* didapat, maka langkah selanjutnya yaitu menghitung *tf-idf weighting*. Bobot *tf-idf* dari suatu term/ token/ kata merupakan hasil perkalian antara *tf wight* dengan *idf*. Berikut merupakan tahapan *tf-idf* pada gambar 3.10.

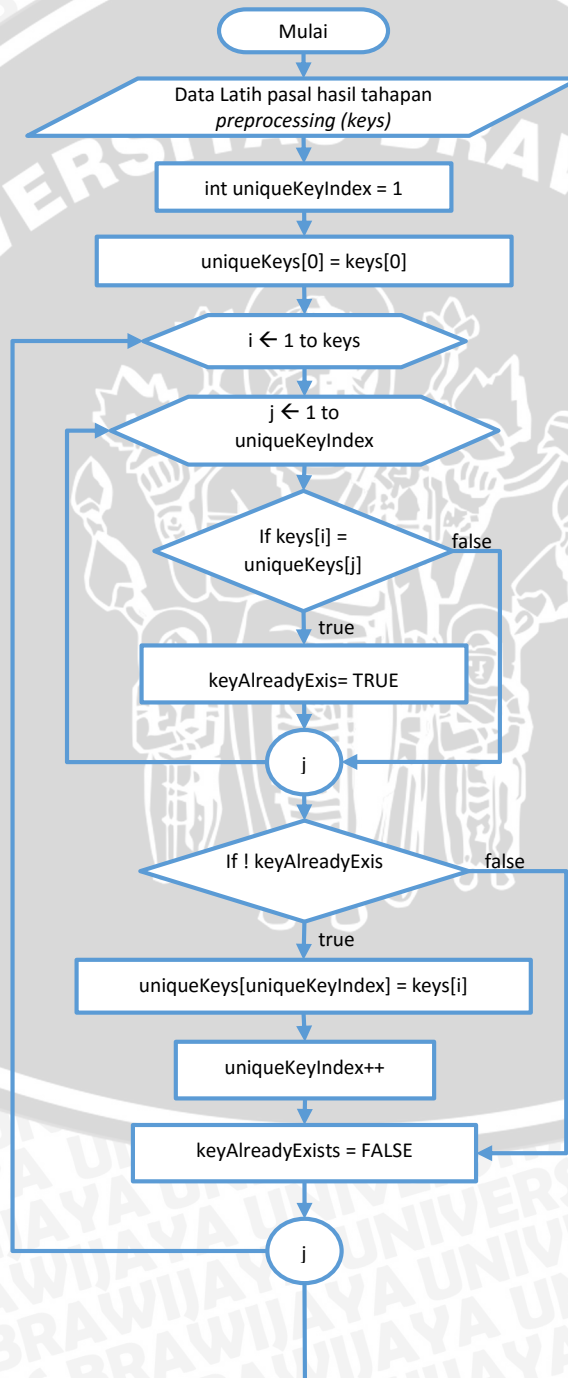


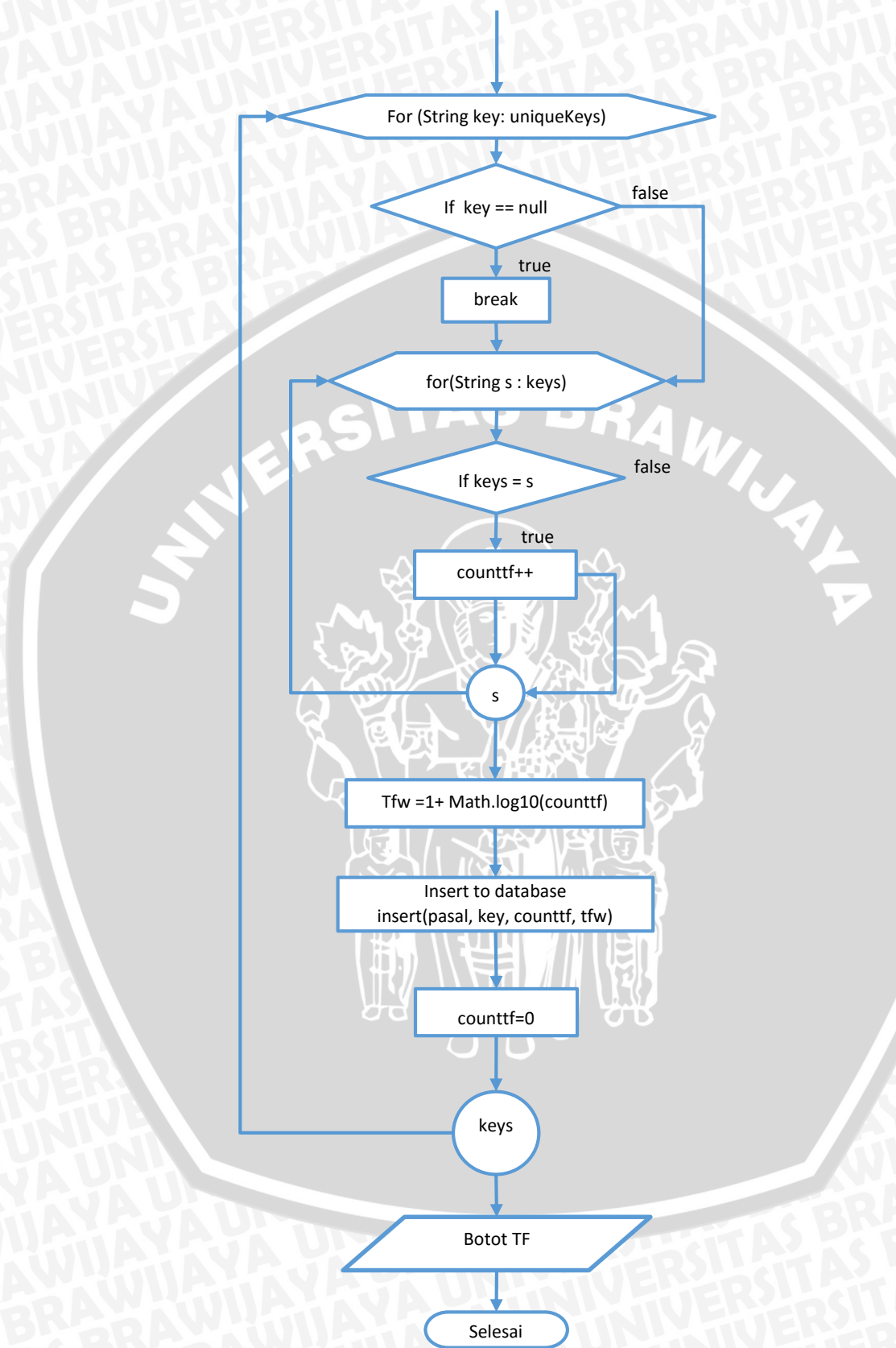
Gambar 3.10 Flowchart TF-IDF



3.3.2.1 Proses Term Frequency

Pada proses pembobotan *tf* ini, langkah awal yang dilakukan yaitu mengambil kata unik yang terdapat pada hasil *preprocessing* seluruh dokumen pasal yang ada. Setelah itu kata unik ini digunakan untuk melakukan perbandingan dengan kata hasil *preprocessing*, untuk mendapatkan nilai *term frequency (tf)* dari setiap kata pada masing-masing dokumen pasal. Setelah mendapat nilai *tf*, dilakukan perhitungan *term frequency weight (tfw)* untuk setiap kata pada masing-masing dokumen pasal. Berikut merupakan *flowchart* tahapan proses *tf* pada gambar 3.11.



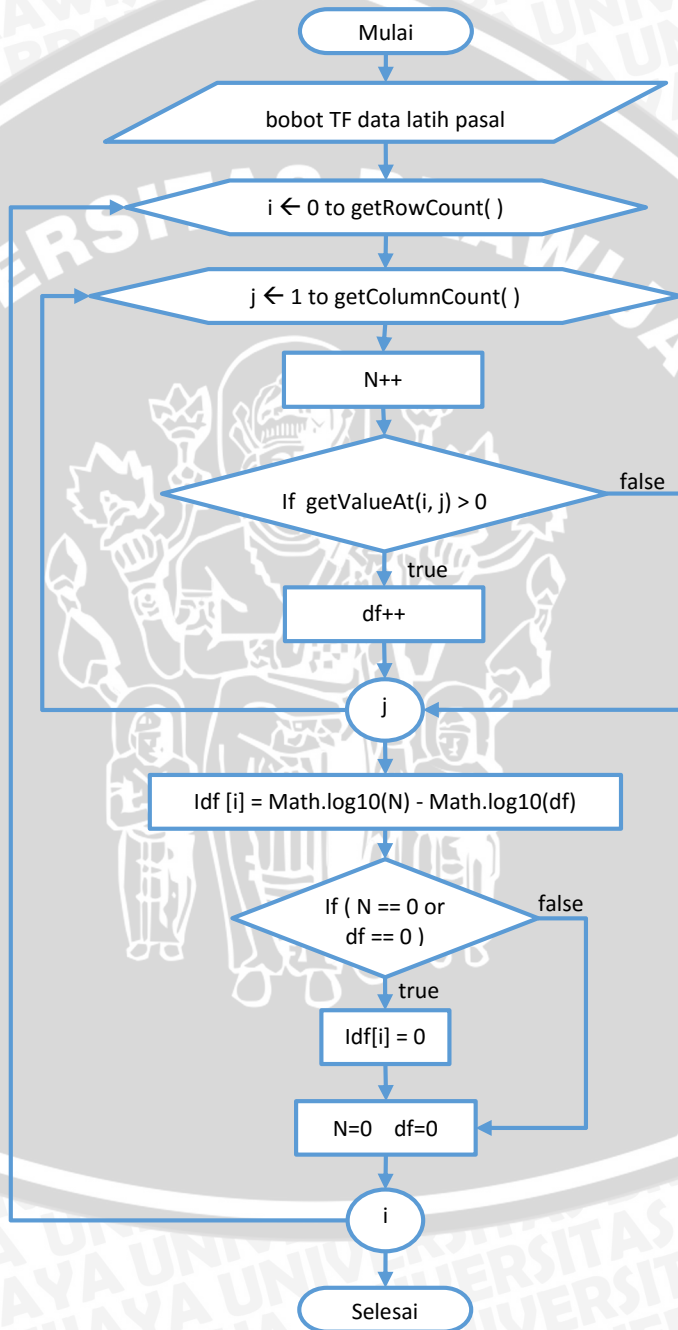


Gambar 3.11 Flowchart proses hitung TF



3.3.2.2 Proses Inverse Document Frequency

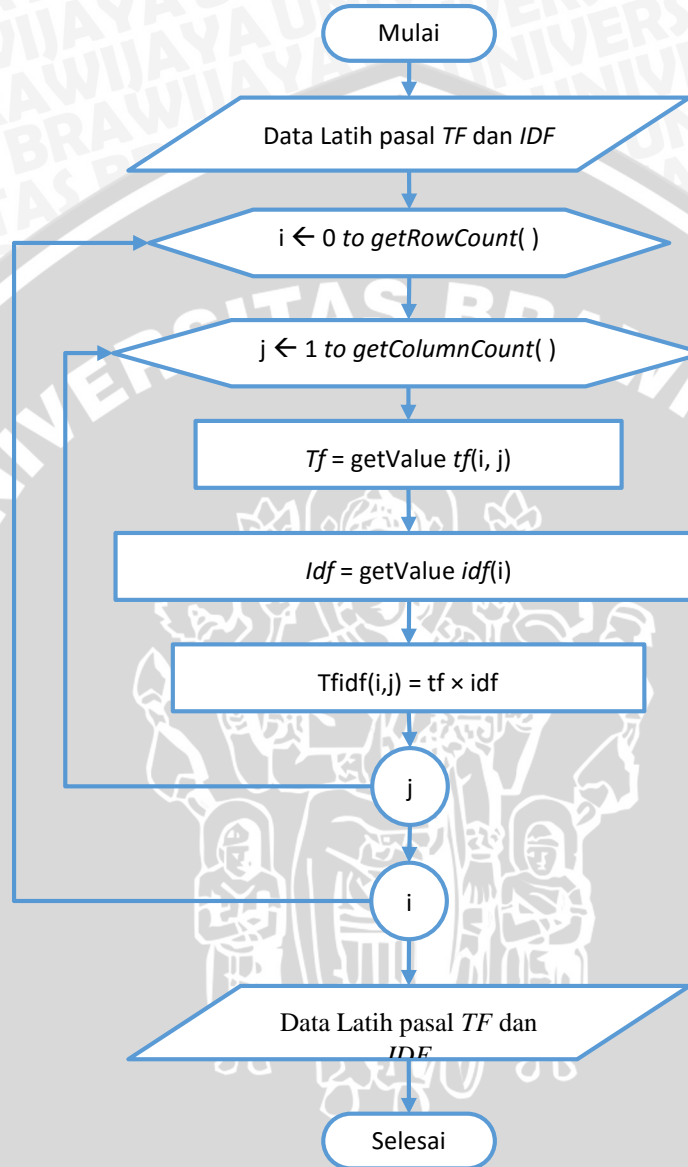
Setelah proses menghitung bobot dari tf dilakukan, maka langkah selanjutnya adalah proses idf . Pada proses ini membutuhkan dua parameter agar idf dapat dihitung yaitu N (banyaknya dokumen yang ada) dan juga df_t (banyaknya dokumen yang mengandung suatu kata tertentu). Kemudian idf didapat dengan membagi N dengan df_t kemudian dikalikan dengan \log_{10} . Berikut merupakan *flowchart* tahapan proses idf pada gambar 3.12.



Gambar 3.12 Flowchart proses hitung IDF

3.3.2.3 Proses TF-IDF

Setelah proses *tf* dan *idf* selesai, maka langkah selanjutnya yaitu proses pencari nilai bobot dari *tf-idf*, dengan cara mengalikan antara *tf* dengan *idf*. Berikut merupakan *flowchart* tahapan proses *tf-idf* pada gambar 3.13.

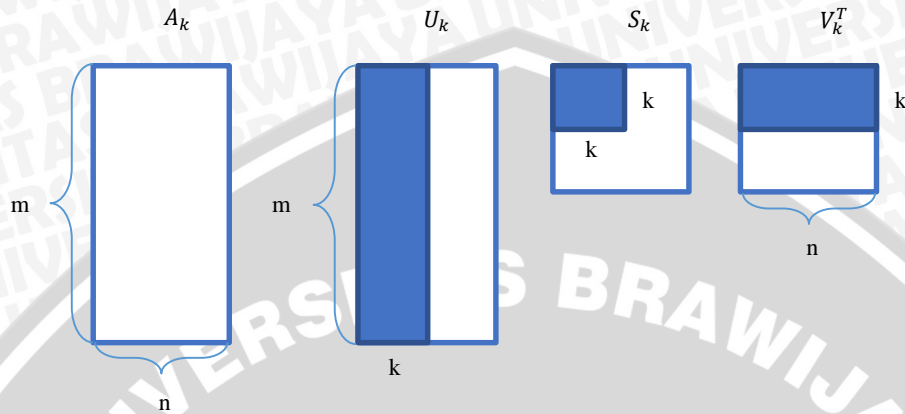


Gambar 3.13 Flowchart proses hitung TF-IDF

3.3.3 Singular Value Decomposition

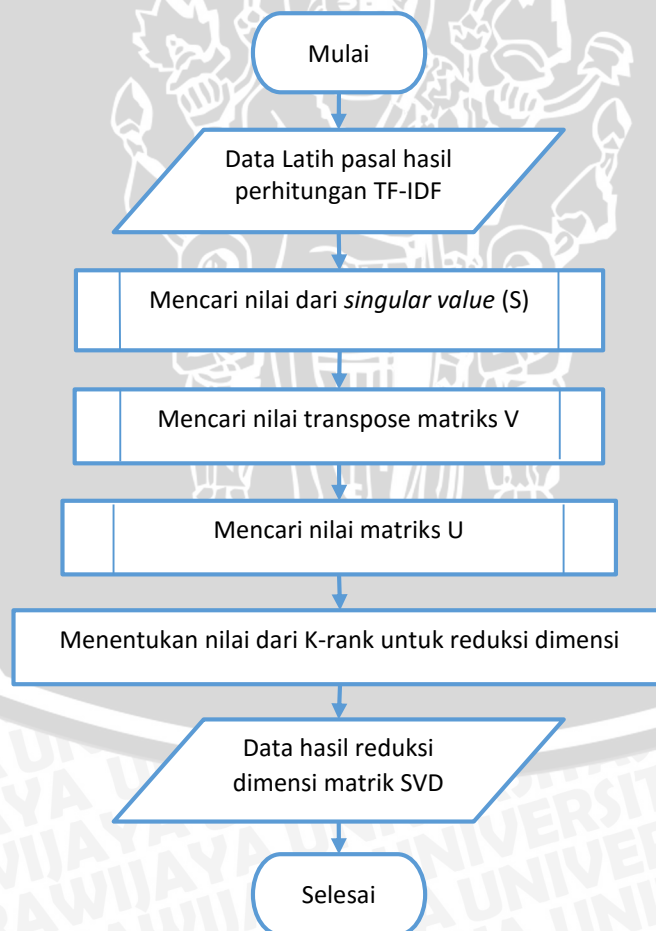
Setelah mendapat nilai bobot dari *tf-idf*, maka langkah selanjutnya yaitu melakukan reduksi dimensi data, sehingga proses perhitungan selanjutnya menjadi lebih efisien dan cepat. *Singular Value Decomposition (SVD)* merupakan model matematis yang digunakan untuk reduksi dimensi data. Proses SVD dilakukan dengan menghitung dekomposisi matriks dari *tf-idf* menjadi tiga bagian yaitu *U*, *S*, dan *V^T*. Pada penelitian ini, proses perhitungan SVD untuk mendapatkan nilai dari *U*, *S*, dan *V^T* menggunakan *library Apache Commons Mathematics*.

Setelah ketiga bagian itu ditemukan nilai matriksnya, maka langkah selanjutnya adalah menentukan nilai dari k -rank yang digunakan untuk mereduksi dimensi dari U , S , dan V^T . Matriks dari U , S , dan V^T yang telah direduksi dimensinya menggunakan k -rank, menjadi U_k , S_k , dan V_k^T . Berikut merupakan ilustrasi reduksi SVD dengan k -rank pada gambar 3.14.



Gambar 3.14 Reduksi SVD dengan k -rank

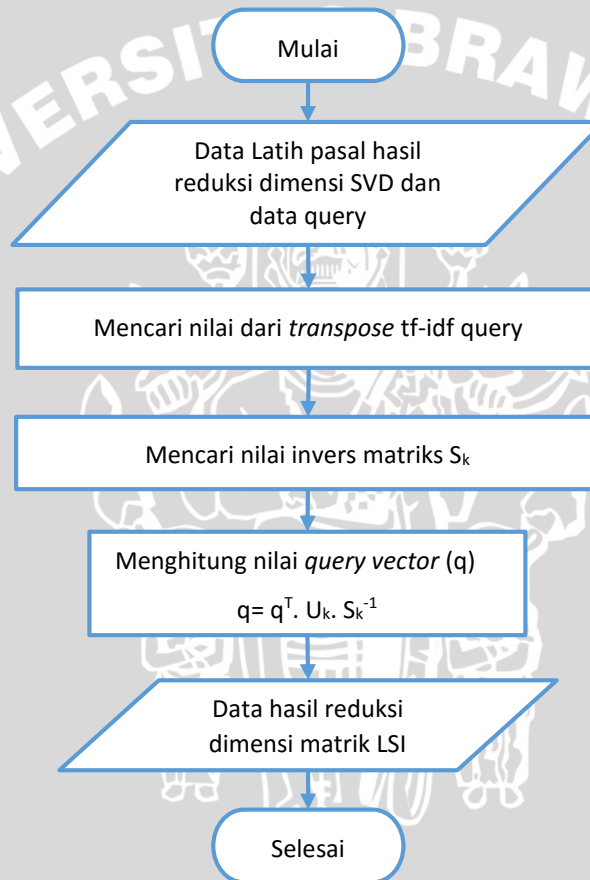
Berikut ini merupakan tahapan SVD yang dapat dilihat pada gambar 3.15.



Gambar 3.15 Flowchart SVD

3.3.4 Latern Semantic Indexing (LSI)

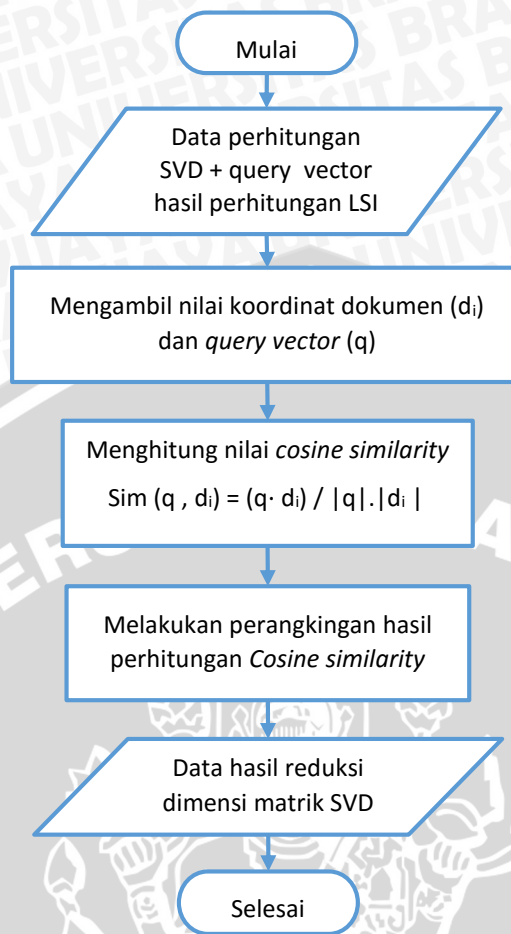
Hasil reduksi dari perhitungan SVD akan digunakan dalam perhitungan LSI, yang mampu menemukan hubungan semantik antar kata, juga mendapatkan suatu pemodelan yang efektif untuk merepresentasikan hubungan antara kata kunci dan dokumen yang dicari. Untuk menghitung LSI, pertama mengambil nilai *tf-idf* dari *query* pada bagian perhitungan *tf-idf* sebelumnya. Kemudian lakukan transpose terhadap nilai *tf-idf query* tersebut (q^T). Selanjutnya menghitung nilai invers dari matriks S_k (S_k^{-1}) pada perhitungan SVD sebelumnya. Langkah terakhir adalah menghitung nilai dari *query vector* (q) dengan cara mengalikan q^T dengan S_k^{-1} dan juga U_k . Maka didapatkan nilai dari perhitungan LSI. Berikut merupakan tahapan LSI pada gambar 3.16.



Gambar 3.16 Flowchart LSI

3.3.5 Cosine Similarity

Perhitungan *cosine similarity* dilakukan setelah mendapatkan nilai dari data hasil proses perhitungan LSI. *Cosine similarity* digunakan untuk menghitung kemiripan antara dua buah vektor, yaitu vektor dokumen pasal dan vektor *query*. Dimana nilai *query vector* (q) diambil dari hasil perhitungan LSI, sedangkan nilai koordinat vektor masing-masing dokumen pasal (d_i) diambil dari matriks V_k (tahap perhitungan SVD) yang mengandung nilai *eigenvector* dari dokumen pasal. Berikut merupakan tahapan perhitungan *cosine similarity* pada gambar 3.17.



Gambar 3.17 Flowchart cosine similarity

3.4 Perancangan User Interface

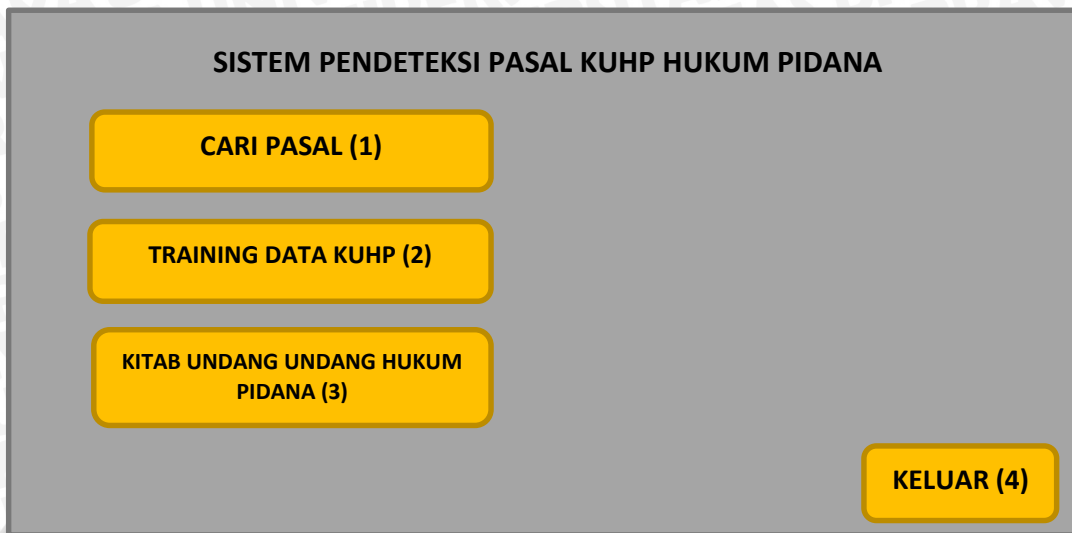
User Interface merupakan mekanisme komunikasi antara pengguna (*user*) berinteraksi dengan sistem. *Interface* dapat menerima informasi dari pengguna (*user*) dan memberikan informasi kepada pengguna (*user*). *Interface* menjembatani antara bahasa manusia dengan bahasa komputer, juga mengkomunikasikan fitur-fitur sistem yang tersedia agar *user* mengerti dan dapat menggunakan sistem tersebut.

3.4.1 Perancangan Tampilan Sistem

Sistem pencarian pasal KUHP berdasarkan kasus ini adalah program yang dibuat dengan menggunakan bahasa pemrograman *Java* dan didukung dengan *database SQL*. Sistem ini akan memberikan *output* berupa pasal yang bersangkutan dengan kasus kejahatan berdasarkan *input* yang diberikan. Sistem ini akan membaca *input* yang diberikan user berupa penjelasan suatu kasus kejahatan. Setelah itu data *input* tersebut akan diproses dan dihitung oleh sistem dan kemudian sistem akan menampilkan hasil *output* sesuai data yang dimasukkan, berupa pasal KUHP yang bersangkutan dengan rincian kasus kejahatan dari *user*. Berikut merupakan gambaran rancangan interface sistem

repository.ub.ac.id

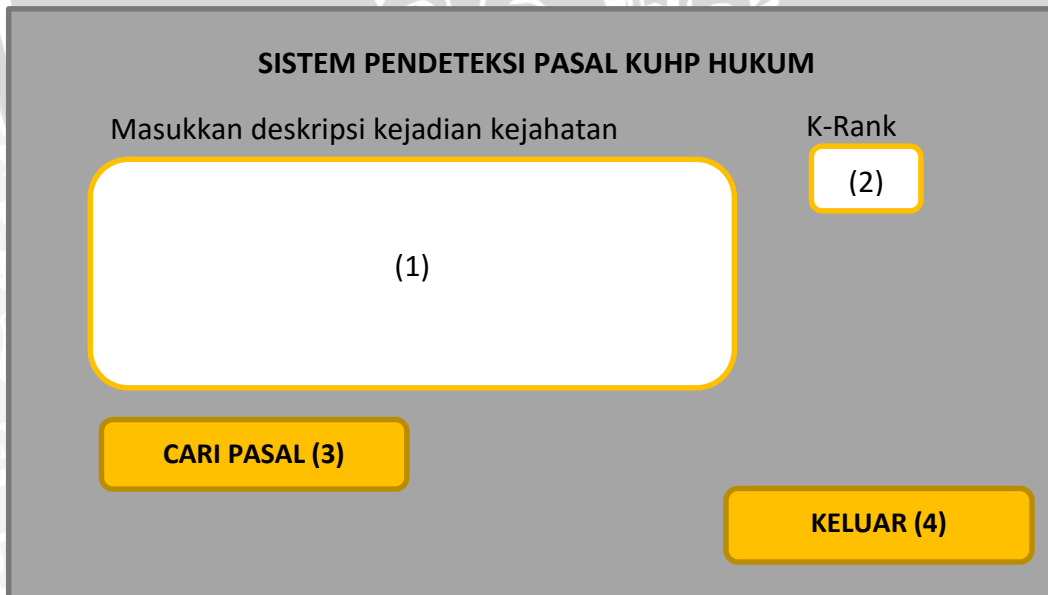
pencaian pasal KUHP serta fitur-fitur yang terdapat didalamnya, yaitu pada gambar 3.18 sampai dengan gambar 3.21.



Gambar 3.18 Rancangan *user interface* menu utama

Berdasarkan Gambar 3.18 dapat dijelaskan sebagai berikut:

1. *Button* Cari Pasal, untuk melakukan proses pencarian pasal KUHP.
2. *Button* Training Data KUHP, untuk melakukan proses training data pasal.
3. *Button* KUHP, untuk melihat data pasal KUHP yang terdapat dalam sistem.
4. *Button* Keluar, untuk keluar dari aplikasi.



Gambar 3.19 Rancangan *user interface* untuk input deskripsi kejadian kejahatan

Berdasarkan Gambar 3.19 dapat dijelaskan sebagai berikut:

1. *Textarea*, tempat *user* untuk memasukkan deskripsi kejadian kejahatan.

2. *Textfield*, tempat *user* untuk memasukkan nilai k-rank.
3. *Button* Cari Pasal, tombol untuk memulai proses pencarian pasal berdasarkan deskripsi yang telah dimasukkan.
4. *Button* Keluar, untuk kembali ke menu utama.

**SISTEM PENDETEKSI PASAL KUHP HUKUM
PIDANA**

Deskripsi kejadian kejahatan (user):

(1)

Pasal yang terkait:

NO	PASAL	ISI
		(2)

Keterangan isi pasal:

(3)

KELUAR (4)

Gambar 3.20 Rancangan *user interface* untuk hasil pencarian pasal

Berdasarkan Gambar 3.20 dapat dijelaskan sebagai berikut:

1. *Textarea*, deskripsi kejadian kejahatan sebelumnya yang dimasukkan oleh *user*.
2. *Table* hasil pencarian pasal, untuk menampilkan hasil pencarian pasal yang berkaitan dengan deskripsi *input user*.
3. *Textarea* keterangan isi pasal, untuk menampilkan isi pasal-pasal secara lengkap pada tabel hasil pencarian diatas, setelah melakukan *click* pada salah satu baris pada tabel hasil pencarian pasal.
4. *Button*, untuk kembali ke menu utama.

**SISTEM PENDETEKSI PASAL KUHP HUKUM
PIDANA**

KUHP:

BAB	PASAL	ISI
		(1)

Keterangan isi pasal:

(2)

KELUAR (3)

Gambar 3.21 Rancangan *user interface* untuk daftar pasal KUHP

Berdasarkan Gambar 3.21 dapat dijelaskan sebagai berikut:

1. *Table*, untuk menampilkan daftar tabel pasal KUHP yang pada *database*.
2. *Textarea*, untuk menampilkan isi secara lengkap pasal pada daftar tabel pasal KUHP, dengan melakukan *click* pada salah satu data pasal pada tabel KUHP.
3. *Button*, untuk kembali ke menu utama.

3.5 Perancangan *Database*

Perancangan *database* pada sistem pencarian pasal KUHP ini digunakan untuk melakukan penyimpanan beberapa parameter, yaitu untuk menyimpan pasal-pasal yang terdapat pada KUHP dan menyimpan beberapa parameter hasil perhitungan dari sistem ini. *Database* pada sistem ini juga dibutuhkan untuk menyimpan data saat melakukan pemrosesan teks pada KUHP dan juga pada deskripsi kejadian kejahatan yang dimasukkan oleh *user*.

3.5.1 Tabel KUHP

Tabel KUHP digunakan untuk menyimpan pasal-pasal yang terdapat pada kitab undang-undang hukum pidana. Data ini kemudian dilakukan *preprocessing* yang akan digunakan dalam perhitungan dengan *query* yang dimasukkan oleh *user*. Terdapat 4 parameter pada tabel KUHP yaitu id, bab, pasal, dan isi. Struktur table KUHP dapat dilihat pada tabel 3.2.

Tabel 3.2 Struktur tabel KUHP

No	Field Name	Type	Size
1	<u>id</u>	Int	4
2	bab	Varchar	6
3	pasal	Varchar	6
4	isi	Text	~

3.5.2 Tabel Training

Tabel *Training* digunakan untuk menyimpan data dari hasil *preprocessing* yang dilakukan terhadap semua pasal KUHP yang terdapat pada *database* sistem pencarian pasal ini. Data hasil proses *preprocessing* yang disimpan pada tabel *training* ini adalah berupa kata dasar hasil *preprocessing* beserta jumlah masing-masing kata dari masing-masing pasal pada KUHP. Pada tabel ini terdapat beberapa *field* yaitu pasal, kata, *tf*, *tfw*, *idf*, dan *tfidf*. Fungsi *field* kata yaitu untuk menyimpan kata dasar hasil *preprocessing*. *Field* pasal untuk menyimpan nomor pasal yang berkaitan dengan masing-masing kata dasar pada *field* kata. *Field* *tf* berfungsi untuk menyimpan jumlah dari kata dasar yang muncul pada masing-masing pasal. *Field* *tfw* (*term frequency weight*) untuk menyimpan hasil perhitungan bobot dari *tf* pada masing-masing kata dasar pada pasal. *Field* *idf* dan *tfidf* digunakan untuk menyimpan hasil perhitungan *idf* dan *tfidf*. Struktur tabel *training* dapat dilihat pada tabel 3.3.

Tabel 3.3 Struktur tabel training

No	Field Name	Type	Size
1	pasal	Varchar	6
2	kata	Varchar	20
3	tf	Int	5
4	tfw	Float	~
5	idf	Float	~
6	tfidf	Float	~

3.5.3 Tabel Testing

Terdapat 3 parameter pada tabel *testing* yaitu kata, *tf*, dan *tfw*. Tabel *testing* berisikan data hasil proses *preprocessing* dari query yang dimasukkan oleh user. Tabel ini berisi kata hasil *preprocessing* pada query, jumlah kata (*tf*), dan juga hasil perhitungan bobot *tf* dari masing-masing kata. Struktur tabel *testing* dapat dilihat pada tabel 3.4.

Tabel 3.4 Struktur tabel *testing*

No	Field Name	Type	Size
1	kata	Varchar	6
2	tf	Int	20
3	tfw	Float	~

3.6 Perhitungan Manual

Berikut merupakan contoh hitungan manual dari sistem ini. Diberikan data training dan *query* sebagai berikut.

Data Training (Database Pasal)

1. Pasal 340
Barang siapa dengan sengaja melakukan pembunuhan dengan rencana.
2. Pasal 362
Barang siapa mengambil barang milik orang lain.
3. Pasal 346
Seseorang wanita yang menggugurkan kandungan.

Query (penjelasan kasus kejahatan yang di masukkan user) :

Pembunuhan dengan rencana

3.6.1 Langkah 1 : Preprocessing

Langkah pertama yang harus dilakukan adalah melakukan tahap *preprocessing* (*case folding*, *tokenizing*, *filtering*, dan *stemming*) terhadap data training atau data pasal pada *database*, juga pada *query* yang dimasukkan oleh *user*. Hasil dari *preprocessing* adalah sebagai berikut:

Data Training Pasal KUHP

- **Case Folding**
 1. Pasal 340
barang siapa dengan sengaja melakukan pembunuhan dengan rencana
 2. Pasal 362
barang siapa mengambil barang milik orang lain
 3. Pasal 346
beseorang wanita yang menggugurkan kandungan

- **Tokenizing**

1. Pasal 340

|barang|siapa|dengan|sengaja|melakukan|pembunuhan|dengan|rencana|

2. Pasal 362

|barang|siapa|mengambil|barang|milik|orang|lain|

3. Pasal 346

|seseorang|wanita|yang|menggugurkan|kandungan|

- **Filtering**

1. Pasal 340

|sengaja|pembunuhan|rencana|

2. Pasal 362

|mengambil|barang|milik|

3. Pasal 346

|wanita|menggugurkan|kandungan|

- **Stemming**

1. Pasal 340

|sengaja|bunuh|rencana|

2. Pasal 362

|ambil|barang|milik|

3. Pasal 346

|wanita|gugur|kandung|

Query (penjelasan kasus kejahatan yang di masukkan user) :

- **Case Folding**

pembunuhan dengan rencana

- **Tokenizing**

|pembunuhan|dengan|rencana|

- **Filtering**

|pembunuhan|rencana|

- **Stemming**

|bunuh|rencana|

Berikut ini pada tabel 3.5 merupakan hasil preprocessing dalam bentuk *inverted index*.

Tabel 3.5 Tabel tf

Tf	Pasal			Query
	340	346	362	
ambil	0	1	0	0
barang	0	1	0	0
bunuh	1	0	0	1
gugur	0	0	1	0
kandung	0	0	1	0
milik	0	1	0	0
rencana	1	0	0	1
wanita	0	0	1	0
sengaja	1	0	0	0

3.6.2 Langkah 2 : Perhitungan *Term Frequency Weight (TF)*

Kemudian dilakukan proses perhitungan bobot dari *tf* (*tf weight*) terhadap masing masing kata. Berikut merupakan contoh perhitungan untuk mencari nilai bobot *tf* menggunakan kata “ambil” pada dokumen pasal “346” menggunakan persamaan 2.1:

$$w_{tf_{t,d}} = \begin{cases} 1 + \log_{10} tf_{t,d}, & \text{if } tf_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases}$$

$$= 1 + \log_{10} 1$$

$$= 1 + 0 = 1$$

Berikut ini pada tabel 3.6 merupakan hasil seluruh perhitungan bobot dari *tf* (*tf weight*):

Tabel 3.6 Tabel TF Weight

Tf weight	Pasal			Query
	340	346	362	
ambil	0	1	0	0
barang	0	1	0	0
bunuh	1	0	0	1
gugur	0	0	1	0
kandung	0	0	1	0
milik	0	1	0	0

rencana	1	0	0	1
wanita	0	0	1	0
sengaja	1	0	0	0

3.6.3 Langkah 3 : Perhitungan *Inverse Document Frequency (IDF)*

Setelah menghitung bobot dari tf , maka langkah selanjutnya adalah menghitung idf dengan membagi N (banyaknya dokumen yang ada) dengan df_t (banyaknya dokumen yang mengandung suatu kata tertentu) kemudian dikalikan dengan \log_{10} . Berikut merupakan contoh perhitungan idf pada kata “ambil” dengan banyak dokumen (N) adalah 3, menggunakan persamaan 2.2.

$$\begin{aligned}
 idf_t &= \log_{10} N/df_t \\
 &= \log_{10} 3/1 \\
 &= 0.477121
 \end{aligned}$$

Berikut ini pada tabel 3.7 merupakan hasil perhitungan dari idf :

Tabel 3.7 Tabel DF dan IDF

DF		IDF	
ambil	1	ambil	0.477121
barang	1	barang	0.477121
bunuh	1	bunuh	0.477121
gugur	1	gugur	0.477121
kandung	1	kandung	0.477121
milik	1	milik	0.477121
rencana	1	rencana	0.477121
wanita	1	wanita	0.477121
sengaja	1	sengaja	0.477121

3.6.4 Langkah 4 : Perhitungan *TF-IDF*

Setelah mendapatkan nilai dari tf dan idf , maka langkah selanjutnya yaitu mencari nilai bobot dari $tf-idf$, dengan cara mengalikan antara tf dengan idf . Berikut merupakan contoh perhitungan pada kata “ambil” dokumen pasal “346” menggunakan persamaan 2.3.

$$\begin{aligned}
 W_{t,d} &= W_{tf,t,d} \times idf_t \\
 &= 1 \times 0.477121 \\
 &= 0.477121
 \end{aligned}$$



Sehingga didapat perhitungan nilai bobot dari *tf-idf* yaitu yang dapat dilihat pada tabel 3.8.

Tabel 3.8 Tabel TF-IDF

TF-IDF	Pasal			Query
	340	346	362	
ambil	0	0.477121	0	0
barang	0	0.477121	0	0
bunuh	0.477121	0	0	0.477121
gugur	0	0	0.477121	0
kandung	0	0	0.477121	0
milik	0	0.477121	0	0
rencana	0.477121	0	0	0.477121
wanita	0	0	0.477121	0
sengaja	0.477121	0	0	0

3.6.5 Langkah 5 : Perhitungan *Singular Value Decomposition (SVD)*

Untuk meringkas proses perhitungan manual SVD, maka digunakan data sederhana sebagai contoh perhitungan manual SVD, yaitu dengan menggunakan contoh data matriks A sebagai berikut:

$$A = \begin{bmatrix} 5 & 5 \\ -1 & 7 \end{bmatrix}$$

Berikut ini merupakan contoh sederhana dari perhitungan manual SVD, untuk menemukan dekomposisi dari matriks A, yaitu matriks U, S, dan V^T .

Pertama-tama melakukan perkalian $A^T A$.

$$A^T = \begin{bmatrix} 5 & -1 \\ 5 & 7 \end{bmatrix}$$

$$A^T A = \begin{bmatrix} 5 & -1 \\ 5 & 7 \end{bmatrix} \begin{bmatrix} 5 & 5 \\ -1 & 7 \end{bmatrix} = \begin{bmatrix} 26 & 18 \\ 18 & 74 \end{bmatrix}$$

Kemudian menghitung *eigenvalues* dari $A^T A$ dan lakukan pengurutan secara *descending*. Akar dari *eigenvalue* merupakan nilai *singular value* dari matriks A.

$$A^T A - C = \begin{bmatrix} 26 - c & 18 \\ 18 & 74 - c \end{bmatrix}$$

$$\begin{aligned} |A^T A - C| &= (26 - c)(74 - c) - (18)(18) \\ &= c^2 - 100c + 1600 \end{aligned}$$

Sehingga

$$c_1 = 80 \qquad c_2 = 20 \qquad \rightarrow \text{eigenvalues}$$

$$s_1 = \sqrt{80} = 8.944 \qquad s_2 = \sqrt{20} = 4.472 \qquad \rightarrow \text{singular values}$$

Jadi didapatkan matrik S dengan meletakkan nilai *singular values* secara *descending* sepanjang diagonal, sebagai berikut:

$$S = \begin{bmatrix} 8.944 & 0 \\ 0 & 4.472 \end{bmatrix}$$

$$S^{-1} = \begin{bmatrix} 0.1118 & 0 \\ 0 & 0.2236 \end{bmatrix}$$

Gunakan *eigenvalues* $c_1 = 80$ untuk menghitung *eigenvectors* dari $A^T A$.

Perhitungan untuk $c_1 = 80$

$$A^T A - C = \begin{bmatrix} 26 - 80 & 18 \\ 18 & 74 - 80 \end{bmatrix} = \begin{bmatrix} -54 & 18 \\ 18 & -6 \end{bmatrix}$$

$$(A^T A - C) X_1 = 0$$

$$\begin{bmatrix} -54 & 18 \\ 18 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$-54x_1 + 18x_2 = 0$$

$$18x_1 - 6x_2 = 0$$

Jadi nilai dari x_2 untuk kedua persamaan diatas adalah $x_2 = 3x_1$

$$X_1 = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ 3x_1 \end{bmatrix}$$

Hitung panjang dokumen

$$L = \sqrt{x_1^2 + x_2^2} = \sqrt{x_1^2 + 3x_1^2} = \sqrt{4x_1^2} = 2x_1$$

$$X_1 = \begin{bmatrix} x_1/L \\ x_2/L \end{bmatrix} = \begin{bmatrix} x_1/2x_1 \\ 3x_1/2x_1 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 1.5 \end{bmatrix}$$

Gunakan *eigenvalues* $c_1 = 20$ untuk menghitung *eigenvectors* dari $A^T A$.

$c_1 = 20$

$$A^T A - C = \begin{bmatrix} 26 - 20 & 18 \\ 18 & 74 - 20 \end{bmatrix} = \begin{bmatrix} -6 & 18 \\ 18 & 54 \end{bmatrix}$$

$$(A^T A - C) X_1 = 0$$

$$\begin{bmatrix} -6 & 18 \\ 18 & 54 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$-6x_1 + 18x_2 = 0$$

$$18x_1 + 54x_2 = 0$$

Jadi nilai dari x_2 untuk kedua persamaan diatas adalah $x_2 = -3x_1$

$$X_2 = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -3x_2 \\ x_2 \end{bmatrix}$$

Hitung panjang dokumen

$$L = \sqrt{x_1^2 + x_2^2} = \sqrt{-3x_2 + x_2} = \sqrt{4x_2^2} = 2x_2$$

$$X_2 = \begin{bmatrix} x_1/L \\ x_2/L \end{bmatrix} = \begin{bmatrix} -3x_2/L \\ x_2/L \end{bmatrix} = \begin{bmatrix} -3x_2/2x_2 \\ x_2/2x_2 \end{bmatrix} = \begin{bmatrix} -1.5 \\ 0.5 \end{bmatrix}$$

Setelah mendapatkan nilai X_1 dan X_2 , maka didapatkan nilai dari matrik V^T , yaitu:

$$V = [X_1 \ X_2] = \begin{bmatrix} 0.5 & -1.5 \\ 1.5 & 0.5 \end{bmatrix}$$

$$V^T = \begin{bmatrix} 0.5 & 1.5 \\ -1.5 & 0.5 \end{bmatrix}$$

Kemudian menghitung matriks U dengan menggunakan persamaan $U=AVS^{-1}$.

$$U = AVS^{-1} = \begin{bmatrix} 5 & 5 \\ -1 & 7 \end{bmatrix} \begin{bmatrix} 0.5 & -1.5 \\ 1.5 & 0.5 \end{bmatrix} \begin{bmatrix} 0.1118 & 0 \\ 0 & 0.2236 \end{bmatrix}$$

$$U = AVS^{-1} = \begin{bmatrix} 1.118 & -1.118 \\ 1.118 & 1.118 \end{bmatrix}$$

Untuk pembuktian, hitung SVD dengan $A=USV^T$

$$A = USV^T = \begin{bmatrix} 1.118 & -1.118 \\ 1.118 & 1.118 \end{bmatrix} \begin{bmatrix} 8.944 & 0 \\ 0 & 4.472 \end{bmatrix} \begin{bmatrix} 0.5 & 1.5 \\ -1.5 & 0.5 \end{bmatrix}$$

$$A = USV^T = \begin{bmatrix} 12.5 & 12.5 \\ -2.5 & 17.5 \end{bmatrix} \approx \begin{bmatrix} 5 & 5 \\ -1 & 7 \end{bmatrix}$$

Maka, dari perhitungan diatas didapatkan matriks U, S, V^T sebagai berikut:

$$U = \begin{bmatrix} 1.118 & -1.118 \\ 1.118 & 1.118 \end{bmatrix} \quad S = \begin{bmatrix} 8.944 & 0 \\ 0 & 4.472 \end{bmatrix} \quad V^T = \begin{bmatrix} 0.5 & 1.5 \\ -1.5 & 0.5 \end{bmatrix}$$

Berdasarkan contoh sederhana cara perhitungan manual SVD diatas, maka setelah mendapatkan nilai *tf-idf* pada ,kemudian melakukan perhitungan *svd* terhadap *tf-idf* dari data training, dimana nilai *tf-idf* merupakan matriks A . Pada penelitian ini, proses perhitungan *SVD* untuk mendapatkan nilai dari U, S , dan V^T menggunakan *library Apache Commons Mathematics*. Secara garis besar untuk menemukan matriks U, S dan V , menggunakan persamaan 2.5.

$$A = USV^T$$

Untuk mencari nilai dari *singular value* (S) maka harus menentukan nilai *eigenvalue* dari matriks $A^T A$ dan urutkan nilai tersebut secara *descending*. Kemudian akarkan nilai *eigenvalue* tersebut untuk mendapatkan nilai *singular value* (S). Sehingga didapat nilai S hasil perhitungan *SVD* menggunakan *library Apache Commons Mathematics* pada tabel 3.9 sebagai berikut.

Tabel 3.9 Hasil perhitungan *singular values* (s)

Singular Values (S)		
0.826	0	0
0	0.826	0
0	0	0.826

Gunakan nilai dari *eigenvalues* yang telah diperoleh pada perhitungan sebelumnya untuk menghitung nilai *eigenvectors* dari $A^T A$. Kemudian letakkan

eigenvectors pada matriks V dan hitung *transpose* dari matriks V , sehingga didapat nilai V^T hasil perhitungan *SVD* menggunakan *library Apache Commons Mathematics* pada tabel 3.10.

Tabel 3.10 Hasil perhitungan V^T

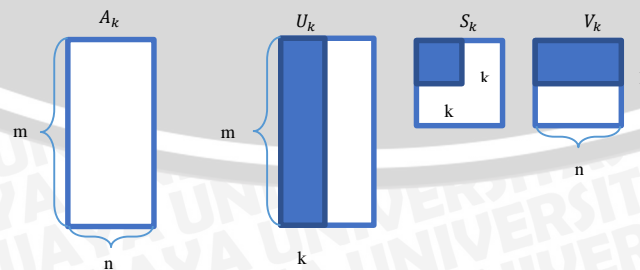
Matrix V^T		
0	0	1
1	0	0
0	1	0

Untuk menghitung U , gunakan $U = AVS^{-1}$. Sehingga didapatkan nilai matriks U hasil perhitungan *SVD* menggunakan *library Apache Commons Mathematics* pada tabel 3.11.

Tabel 3.11 Hasil perhitungan U

Matrix U		
0	0	0.577
0	0	0.577
0	0.577	0
0.577	0	0
0.577	0	0
0	0	0.577
0	0.577	0
0.577	0	0
0	0.577	0

Setelah mendapatkan nilai dari U , S dan V , maka selanjutnya menentukan nilai dari k -rank yang akan digunakan untuk mereduksi dimensi dari matriks A yang dapat di lihat pada gambar 3.22, dimana matriks U menggambarkan banyaknya baris pada matriks A . Sementara matriks V menggambarkan banyaknya kolom pada matriks A . Serta matriks S merupakan matriks simetris yang berisi nilai positif di sepanjang diagonal.



Gambar 3.22 Reduksi SVD

3.6.6 Langkah 6 : Menentukan *K-Rank*

Dengan menggunakan contoh perhitungan SVD sebelumnya, maka didapatkan nilai matriks S , U , dan V^T dari matriks A sebagai berikut.

$$A = \begin{bmatrix} 0 & 0.477 & 0 \\ 0 & 0.477 & 0 \\ 0.477 & 0 & 0 \\ 0 & 0 & 0.477 \\ 0 & 0.477 & 0 \\ 0.477 & 0 & 0 \\ 0 & 0 & 0.477 \\ 0.477 & 0 & 0 \end{bmatrix}$$

$$S = \begin{bmatrix} 0.826 & 0 & 0 \\ 0 & 0.826 & 0 \\ 0 & 0 & 0.826 \end{bmatrix}$$

$$V^T = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$U = \begin{bmatrix} 0 & 0 & 0.577 \\ 0 & 0 & 0.577 \\ 0 & 0.577 & 0 \\ 0.577 & 0 & 0 \\ 0.577 & 0 & 0 \\ 0 & 0 & 0.577 \\ 0 & 0.577 & 0 \\ 0.577 & 0 & 0 \\ 0 & 0.577 & 0 \end{bmatrix}$$

Untuk nilai dari *K-rank* pada kasus ini diberi nilai 2. Sehingga nilai matriks dari U , S dan V menjadi sebagai berikut.

$$K\text{-rank} = 2$$

$$S_k = \begin{bmatrix} 0.826 & 0 \\ 0 & 0.826 \end{bmatrix}$$

$$V_k^T = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

$$U_k = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0.577 \\ 0.577 & 0 \\ 0.577 & 0 \\ 0 & 0 \\ 0 & 0.577 \\ 0.577 & 0 \\ 0 & 0.577 \end{bmatrix}$$

3.6.7 Langkah 7 : Menghitung *Latent Semantic Indexing (LSI)*

Untuk menemukan *query vector* (q) dari *query* sebelumnya, kalikan nilai *transpose* dari *tf-idf query* dengan nilai matriks U_k kemudian dikalikan lagi dengan matriks S_k^{-1} sehingga didapatkan koordinat dari *query vector* (q), dengan menggunakan persamaan 2.9.

$$q' = q^T \cdot U_k \cdot S_k^{-1}$$

$$q^T = [0 \ 0 \ 0.477 \ 0 \ 0 \ 0 \ 0.477 \ 0 \ 0]$$

$$S_k^{-1} = \begin{bmatrix} \frac{1}{0.826} & 0 \\ 0 & \frac{1}{0.826} \end{bmatrix}$$

$$U_k = \begin{bmatrix} 0.000 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.577 \\ 0.577 & 0.000 \\ 0.577 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.577 \\ 0.577 & 0.000 \\ 0.000 & 0.577 \end{bmatrix}$$

Sehingga nilai q yang didapat adalah

$$q = [0 \ 0 \ 0.477 \ 0 \ 0 \ 0 \ 0.477 \ 0 \ 0] \begin{bmatrix} 0.000 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.577 \\ 0.577 & 0.000 \\ 0.577 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.577 \\ 0.577 & 0.000 \\ 0.000 & 0.577 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0.826 & 0 \\ 0 & 1 \\ 0 & 0.826 \end{bmatrix}$$

$$q = [0 \ 0.667]$$

Ambil nilai matriks V_k pada perhitungan langkah 6 sebelumnya. Nilai matriks V_k mengandung koordinat vektor dari masing-masing dokumen pasal, sebagai berikut.

	v_k^T	\rightarrow	v_k	
0	0	1	0	1
1	0	0	0	0
			1	0

Sehingga didapat koordinat vektor dari masing-masing dokumen pasal yang digunakan dalam perhitungan *cosine similarity*, yaitu:

- $d_1(0, 1)$
- $d_2(0, 0)$
- $d_3(1, 0)$

3.6.8 Langkah 8 : Menghitung *Cosine Similarity*

Setelah mendapatkan *query vector* (q) dan koordinat masing-masing dokumen (d_i) maka langkah selanjutnya adalah menghitung *cosine similarity* untuk menghitung kedekatan antara dokumen dengan query, menggunakan persamaan 2.4.

$$sim(q, d_i) = \frac{q \cdot d_i}{|q||d_i|}$$



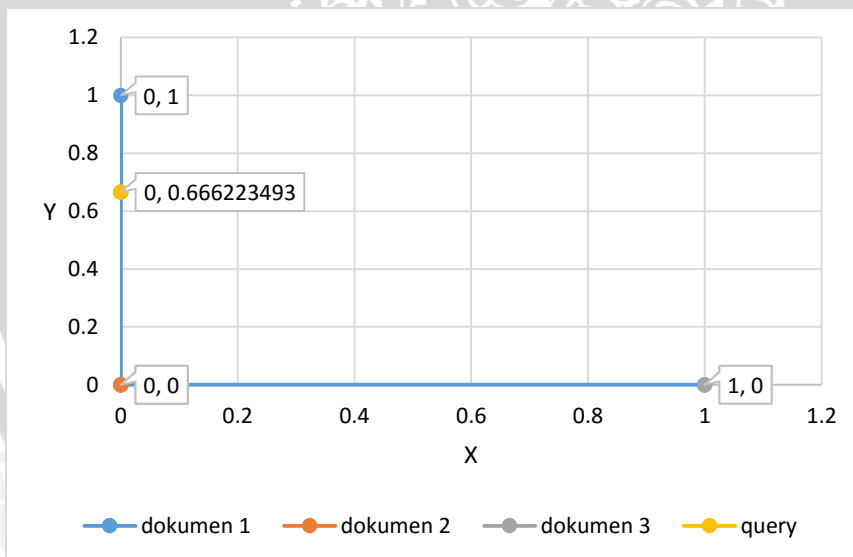
$$\begin{aligned} \text{sim}(q, d_1) &= \frac{(0)(0) + (0.667)(1)}{\sqrt{(0)^2 + (0.667)^2} \sqrt{(0)^2 + (1)^2}} = \frac{0.667}{0.67} = 1.00 \\ \text{sim}(q, d_2) &= \frac{(0)(0) + (0.667)(0)}{\sqrt{(0)^2 + (0.667)^2} \sqrt{(0)^2 + (0)^2}} = \frac{0}{0} = \infty \\ \text{sim}(q, d_3) &= \frac{(0)(1) + (0.667)(0)}{\sqrt{(0)^2 + (0.667)^2} \sqrt{(1)^2 + (0)^2}} = \frac{0}{0.445} = 0 \end{aligned}$$

Setelah melakukan perhitungan *cosine similarity*, dilakukan perangkingan nilai *cosine similarity* dari nilai terbesar hingga terkecil.

$$d_1 > d_2 > d_3$$

Dari hasil perhitungan di atas maka hasil nilai *cosine similarity* terbesar yaitu pada dokumen 1 (pasal 340) dengan nilai 1.00, sehingga dokumen yang paling relevan dengan *query* yang dimasukkan adalah dokumen 1 (pasal 340). Pasal 340 berbunyi, “barang siapa dengan sengaja melakukan pembunuhan dengan rencana”. Dimana pasal 340 tersebut sesuai dengan *query* deskripsi kejahatan yang dimasukkan, yaitu “pembunuhan dengan rencana”.

Berikut merupakan diagram vektor hasil perhitungan *cosine similarity* pada gambar 3.23.



Gambar 3.23 Diagram vektor hasil perhitungan *cosine similarity*

BAB 4 IMPLEMENTASI

Bab ini menjelaskan tentang implementasi dari proses perancangan yang telah dijelaskan pada bab sebelumnya. Implementasi ini dilakukan untuk mengetahui akurasi dari sistem yang dirancang.

4.1 Spesifikasi Sistem

Spesifikasi yang sesuai sangat dibutuhkan oleh sistem agar dapat berjalan dengan optimal. Spesifikasi sistem disini terbagi menjadi 2 yaitu spesifikasi perangkat keras (*hardware*) dan spesifikasi perangkat lunak (*software*).

4.1.1 Spesifikasi Perangkat Keras (*Hardware*)

Perangkat keras (*hardware*) yang digunakan dalam pembuatan dan pengembangan sistem pencarian pasal KUHP ini menggunakan komputer dengan spesifikasi sebagai berikut :

1. Toshiba Qosmio F750
2. Processor Intel® Core™ i7-2630QM CPU @ 2.00GHz
3. NVIDIA GEFORCE GT 540M 2 GB
4. Memory DDR3 8 GB RAM
5. Harddisk 1 TB
6. Monitor 15,5"
7. Keyboard

4.1.2 Spesifikasi Perangkat Lunak (*Software*)

Perangkat lunak yang diinstall pada komputer untuk pembuatan dan pengembangan sistem pencarian pasal KUHP:

1. Sistem Operasi Windows 10 Home (64-bit)
2. Netbean 8.0.2
3. XAMPP 3.2.1
4. Apache 2.4.10
5. MySQL 4.2.7.1
6. Adobe Photoshop CS6 (64-bit)

4.2 Implementasi Antarmuka Sistem

Implementasi antarmuka merupakan implementasi untuk mengetahui cara kerja aplikasi menggunakan *interface* atau antarmuka dari sistem yang sudah dirancang dan dijelaskan pada bab 3. Pada implementasi ini terdapat antarmuka menu utama, melihat pasal pada KUHP, preprocessing data KUHP, dan pencarian pasal.

4.2.1 Antarmuka Menu Sistem

Pada menu utama sistem ini terdapat beberapa tombol untuk menuju ke berbagai *submenu*. Beberapa *submenu* yang terdapat pada menu utama ini yaitu tombol *submenu* pencarian pasal, *training* data KUHP, dan tombol kitab undang undang hukum pidana. Menu utama dari sistem ini dapat dilihat pada gambar 4.1 berikut.



Gambar 4.1 Halaman menu utama

4.2.2 Antarmuka Kitab Undang Undang Hukum Pidana

Antarmuka pada *submenu* KUHP ini berguna untuk melihat semua data KUHP yang terdapat pada *database*, berupa nomer bab, nomer pasal, dan isi pasal dari suatu pasal terkait. Pada *submenu* ini terdapat tombol untuk melakukan penambahan data pasal KUHP kedalam *database*. Antarmuka KUHP ini dapat dilihat pada gambar 4.2.

KITAB UNDANG UNDANG HUKUM PIDANA

Tambah Pasal

Pasal KUHP

NO	ID	BAB	PASAL	ISI
1	1	XIV	340	Barangsiapa dengan sengaja dan dengan rencana terle...
2	2	XIV	346	Seorang wanita yang sengaja menggugurkan atau mem...
3	3	XIV	362	Barangsiapa mengambil barang sesuatu, yang seluruh...
4	4	XIV	351	Penganiayaan diancam dengan pidana penjara paling l...

Isi Pasal

Keluar

Gambar 4.2 Halaman KUHP

4.2.3 Antarmuka Training KUHP

Pada submenu *Training* KUHP ini dilakukan proses *preprocessing* terhadap seluruh pasal, kemudian dilanjutkan dengan perhitungan TF-IDF. Antarmuka ini menampilkan data pasal KUHP yang disertai dengan tabel *inverted index* hasil proses *preprocessing* kata pada seluruh pasal, menampilkan perhitungan *Term Frequency* (TF) dan *Inverse Document Frequency* (IDF) pada gambar 4.4, serta menampilkan tabel hasil perhitungan TF-IDF pada gambar 4.5. Pada antarmuka ini terdapat tombol *Singular Value Decomposition* (SVD), yang berguna untuk melanjutkan proses perhitungan *Singular Value Decomposition* (SVD), dari proses perhitungan TF-IDF. Antarmuka dari preprocessing KUHP terdapat pada gambar 4.3.

TRAINING KUHP

NO	ID	BAB	PASAL	ISI
1	1	XIV	340	Barangsiapa dengan sengaja dan dengan renc...
2	2	XIV	346	Seorang wanita yang sengaja mengguurkan at...
3	3	XIV	362	Barangsiapa mengambil barang sesuatu, yang ...
4	4	XIV	351	Penganiayaan diancam dengan pidana penjara ...

Isi Pasal
Barangsiapa dengan sengaja dan dengan rencana terlebih dahulu merampas nya wa orang lain, diancam karena pembunuhan dengan rencana, dengan pidana mat i atau pidana penjara seumur hidup atau selama waktu tertentu, paling lama dua p uluh tahun.

Term Frequency

kata	340	346	362	351	IDF
ambil	0.0	0.0	1.0	0.0	0.602059991...
ancam	1.0	1.0	1.0	1.0	0.0
aniaya	0.0	0.0	0.0	1.0	0.602059991...
ati	0.0	1.0	0.0	0.0	0.602059991...
barang	0.0	0.0	1.0	0.0	0.602059991...
barangsiapa	1.0	0.0	1.0	0.0	0.301029995...
bulan	0.0	0.0	0.0	1.0	0.602059991...
bunuh	1.0	0.0	0.0	0.0	0.602059991...
curi	0.0	0.0	1.0	0.0	0.602059991...
delap	0.0	0.0	0.0	1.0	0.602059991...
denda	0.0	0.0	1.0	1.0	0.301029995...

TF-IDF

Kata	340	346	362	351
ambil	0.0	0.0	0.602059991327...	0.0
ancam	0.0	0.0	0.0	0.0
aniaya	0.0	0.0	0.0	0.602059991327...
ati	0.0	0.602059991327...	0.0	0.0
barang	0.0	0.0	0.602059991327...	0.0
barangsiapa	0.301029995663...	0.0	0.301029995663...	0.0
bulan	0.0	0.0	0.0	0.602059991327...
bunuh	0.602059991327...	0.0	0.0	0.0
curi	0.0	0.0	0.602059991327...	0.0
delap	0.0	0.0	0.0	0.602059991327...
denda	0.0	0.0	0.301029995663...	0.301029995663...

SINGULAR VALUE DECOMPOSITION (SVD)

Gambar 4.3 Halaman preprocessing KUHP

Term Frequency

kata	340	346	362	351	IDF
ambil	0.0	0.0	1.0	0.0	0.602059991...
ancam	1.0	1.0	1.0	1.0	0.0
aniaya	0.0	0.0	0.0	1.0	0.602059991...
ati	0.0	1.0	0.0	0.0	0.602059991...
barang	0.0	0.0	1.0	0.0	0.602059991...
barangsiapa	1.0	0.0	1.0	0.0	0.301029995...
bulan	0.0	0.0	0.0	1.0	0.602059991...
bunuh	1.0	0.0	0.0	0.0	0.602059991...
curi	0.0	0.0	1.0	0.0	0.602059991...
delap	0.0	0.0	0.0	1.0	0.602059991...
denda	0.0	0.0	1.0	1.0	0.301029995...

Gambar 4.4 Tabel Inverted Index TF dan IDF

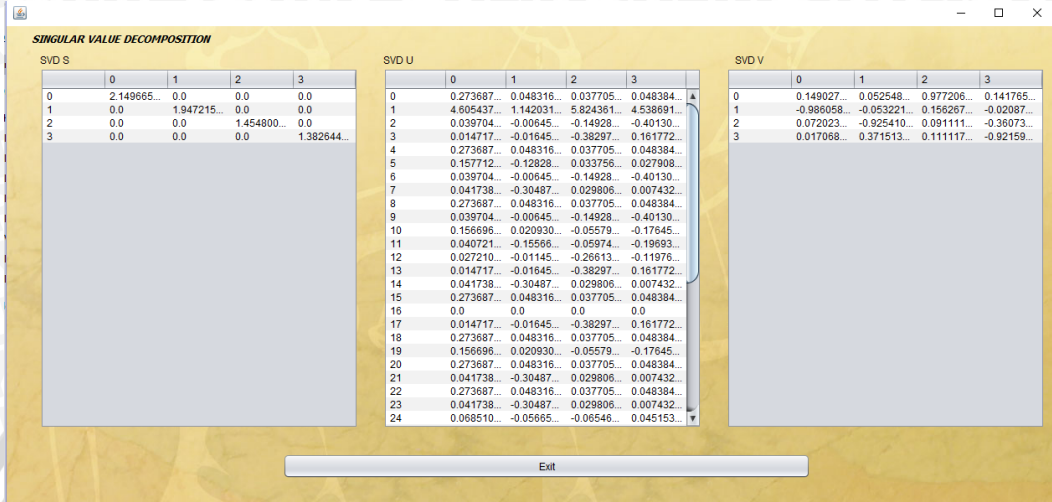
TF-IDF

Kata	340	346	362	351
ambil	0.0	0.0	0.602059991327...	0.0
ancam	0.0	0.0	0.0	0.0
aniaya	0.0	0.0	0.0	0.602059991327...
ati	0.0	0.602059991327...	0.0	0.0
barang	0.0	0.0	0.602059991327...	0.0
barangsiapa	0.301029995663...	0.0	0.301029995663...	0.0
bulan	0.0	0.0	0.0	0.602059991327...
bunuh	0.602059991327...	0.0	0.0	0.0
curi	0.0	0.0	0.602059991327...	0.0
delap	0.0	0.0	0.0	0.602059991327...
denda	0.0	0.0	0.301029995663...	0.301029995663...

Gambar 4.5 Tabel Inverted Index TF- IDF

4.2.4 Antarmuka Singular Value Decomposition (SVD) KUHP

Antarmuka SVD berguna untuk menampilkan hasil perhitungan dari proses SVD berupa matriks S, matriks U, dan matriks V. Antarmuka SVD dapat dilihat pada gambar 4.6 berikut.



The screenshot displays the 'SINGULAR VALUE DECOMPOSITION' interface with three matrix results:

	0	1	2	3
0	2.149665...	0.0	0.0	0.0
1	0.0	1.947215...	0.0	0.0
2	0.0	0.0	1.454800...	0.0
3	0.0	0.0	0.0	1.382644...

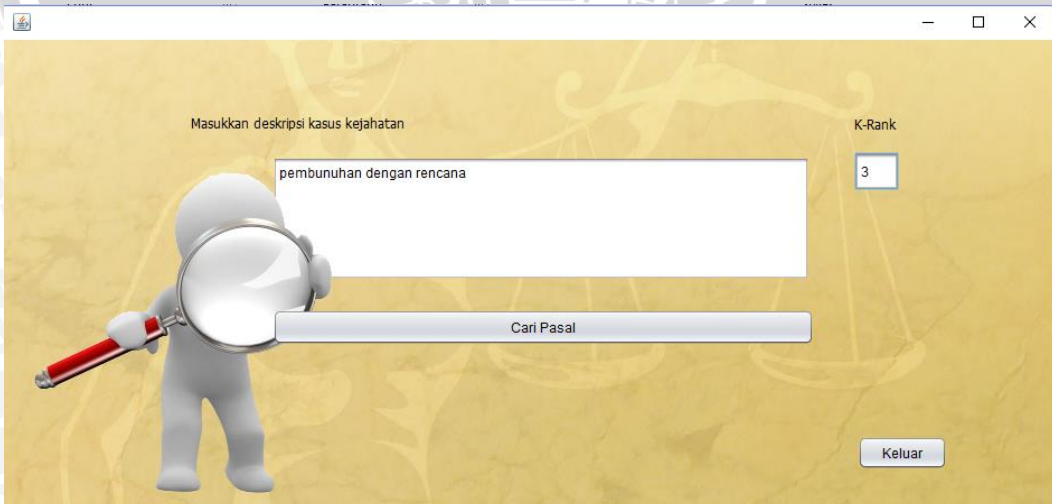
	0	1	2	3
0	0.273687...	0.048316...	0.037705...	0.048384...
1	4.605437...	1.142031...	5.824361...	4.538691...
2	0.039704...	-0.00645...	-0.14928...	-0.40130...
3	0.014717...	-0.01645...	-0.38297...	0.161772...
4	0.273687...	0.048316...	0.037705...	0.048384...
5	0.157712...	-0.12828...	0.033756...	0.027908...
6	0.039704...	-0.00645...	-0.14928...	-0.40130...
7	0.041738...	-0.30487...	0.029806...	0.007432...
8	0.273687...	0.048316...	0.037705...	0.048384...
9	0.039704...	-0.00645...	-0.14928...	-0.40130...
10	0.156696...	0.020930...	-0.05579...	-0.17645...
11	0.040721...	-0.15566...	-0.05974...	-0.19693...
12	0.027210...	-0.01145...	-0.25613...	-0.11976...
13	0.014717...	-0.01645...	-0.38297...	0.161772...
14	0.041738...	-0.30487...	0.029806...	0.007432...
15	0.273687...	0.048316...	0.037705...	0.048384...
16	0.0	0.0	0.0	0.0
17	0.014717...	-0.01645...	-0.38297...	0.161772...
18	0.273687...	0.048316...	0.037705...	0.048384...
19	0.156696...	0.020930...	-0.05579...	-0.17645...
20	0.273687...	0.048316...	0.037705...	0.048384...
21	0.041738...	-0.30487...	0.029806...	0.007432...
22	0.273687...	0.048316...	0.037705...	0.048384...
23	0.041738...	-0.30487...	0.029806...	0.007432...
24	0.068510...	-0.05665...	-0.06546...	0.045153...

	0	1	2	3
0	0.149027...	0.052548...	0.977206...	0.141765...
1	-0.986058...	-0.053221...	0.156267...	-0.02087...
2	0.072023...	-0.925410...	0.091111...	-0.36073...
3	0.017068...	0.371513...	0.111117...	-0.92159...

Gambar 4.6 Halaman hitung SVD

4.2.5 Antarmuka Pencarian Pasal KUHP

Pada antarmuka pencarian pasal KUHP ini berfungsi untuk melakukan *input query* berupa deskripsi suatu kasus kejahatan yang ingin dicari keterkaitannya terhadap pasal yang pada KUHP. Kemudian menginputkan nilai K-Rank yang diinginkan untuk melakukan mereduksi dimensi matriks pada perhitungan SVD selanjutnya. Untuk melakukan proses pencarian pasal tekan tombol Cari Pasal. Antarmuka pencarian pasal terdapat pada gambar 4.7 berikut.



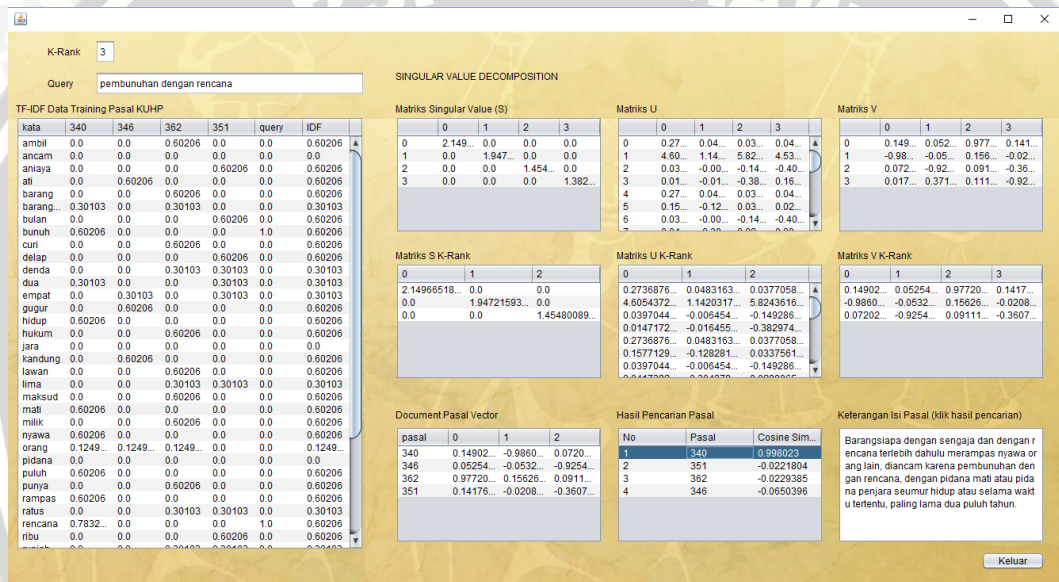
The screenshot shows the search interface with the following elements:

- Label: "Masukkan deskripsi kasus kejahatan"
- Input field: "pembunuhan dengan rencana"
- Label: "K-Rank"
- Input field: "3"
- Button: "Cari Pasal"
- Button: "Keluar"

Gambar 4.7 Halaman pencarian pasal KUHP

4.2.6 Antarmuka Hasil Pencarian Pasal KUHP

Antarmuka hasil pencarian pasal KUHP menampilkan hasil dari perhitungan *query* atau deskripsi kasus yang telah dimasukkan pada gambar 4.7, dengan data pasal KUHP yang telah di training sebelumnya pada gambar 4.3 dan 4.6. Antarmuka ini melakukan proses reduksi matriks SVD berdasarkan K-Rank yang dimasukkan. Kemudian melakukan perhitungan *Latent Semantic Indexing (LSI)*, dilanjutkan dengan menghitung *Cosine Similarity* antara *vector query* dengan *vector* pasal. Hasil *Cosine Similarity* kemudian di urutkan berdasarkan nilai yang terbesar. Pada *interface* ini ditampilkan tabel *inverted index* TF-IDF *query* dan dokumen pasal pada gambar 4.9, tabel hasil perhitungan SVD yang telah direduksi dengan K-Rank pada gambar 4.10, dan juga tabel hasil *Cosine Similarity* yang diurutkan secara *descending* pada gambar 4.11. Berikut merupakan antarmuka dari hasil pencarian pasal KUHP pada gambar 4.8.



Gambar 4.8 Halaman hasil pencarian pasal KUHP

BAB 5 HASIL ANALISA DAN PEMBAHASAN

Pada bab ini menjelaskan mengenai hasil dan pembahasan skenario pengujian. Bab 5 berisi skenario pengujian, hasil pengujian dan analisa hasil. Pengujian akurasi dilakukan untuk mengetahui tingkat akurasi dari sistem pencarian pasal pada KUHP menggunakan metode *Cosine Similarity* dan *Latent Semantic Indexing* (LSI).

5.1 Skenario Pengujian

Skenario pengujian yang digunakan untuk mengevaluasi sistem temu kembali informasi pada penelitian ini yaitu menggunakan evaluasi *ranked retrieval* dengan *Mean Average Precision* (MAP). Data yang digunakan adalah pasal KUHP dengan jumlah pasal pada data training yaitu sebanyak 60 pasal, sedangkan pada skenario pengujian ini menggunakan deskripsi kasus kejahatan sebanyak 6 kasus, yang terdiri dari berbagai macam kasus kejahatan yang terkait dengan pasal KUHP pada Buku Kedua tentang kejahatan, dengan nilai *k-rank* untuk masing-masing kasus yaitu 5, 10, 20, 30, 40, 50 dan 59.

Deskripsi kasus kejahatan dimasukkan kedalam sistem temu kembali informasi beserta nilai *k-rank* yang kemudian menghasilkan pasal-pasal yang terkait dengan deskripsi kasus yang dimasukkan. Hasil *output* pasal-pasal yang direkomendasikan oleh sistem ini kemudian dilakukan perhitungan dengan *Mean Average Precision* (MAP) yang akan dihitung nilai *precision* masing-masing dokumen pasal yang direkomendasikan dengan memperhatikan urutan pasal yang diberikan oleh sistem, yaitu sebanyak 5 dokumen pasal teratas berdasarkan besar nilai bobot *cosine similarity*.

5.2 Hasil Pengujian

Pada pengujian ini, jumlah deskripsi kasus kejahatan (*query*) yang diujikan sebanyak 6 deskripsi kasus kejahatan yang diambil dari pakar hukum. Dimana 6 kasus tersebut berkaitan dengan kasus pada KUHP Buku Kedua, dengan nilai *k-rank* untuk masing-masing kasus yaitu 5, 10, 20, 30, 40,50 dan 59. Data uji atau deskripsi kasus kejahatan yang diujikan dapat dilihat pada Tabel 5.1.

Tabel 5.1 Data Uji Deskripsi Kasus (*Query*)

No	Deskripsi	Pasal Relevan	Bab
1	Budi dengan sengaja dan dengan rencana terlebih dahulu merampas nyawa Ali. Dalam pembunuhan berencana ini, Budi melakukan penganiayaan berat terhadap Ali hingga mengakibatkan kematian.	340, 338 355	XIX XX
2	Seorang wanita yang sengaja menggugurkan atau mematikan kandungannya. Wanita ini menggugurkan atau mematikan kandungannya dengan persetujuan orang tuanya. Wanita ini dibantu oleh seorang dokter, bidan atau juru obat dalam melakukan pengguguran kandungan. Kemudian perbuatan menggugurkan kandungan itu mengakibatkan matinya wanita tersebut.	346, 348, 349	XIX

Tabel 5.1 Data Uji Deskripsi Kasus (*Query*) (lanjutan)

No	Deskripsi	Pasal Relevan	Bab
3	Dua orang pencuri melakukan pencurian pada waktu malam dalam sebuah rumah yang tertutup, pelaku pencurian masuk ke dalam rumah dengan memanjat tembok kemudian merusak pintu. Untuk mempermudah pencurian, pencuri menusuk pemilik rumah hingga luka-luka berat. Pencuri kemudian melarikan diri bersama barang curian.	365	XXII
4	Angga melakukan fitnah kepada Bepriandi dengan melakukan tuduhan kepada Bepriandi tanpa bukti yang jelas, dan tuduhan yang dilakukan Angga tersebut bertentangan dengan apa yang dilakukan oleh Bepriandi, sehingga karena fitnah tersebut nama baik Bepriandi menjadi tercemar.	311	XVI
5	Pendemo dengan sengaja menghancurkan merusakkan atau membikin tak dapat dipakai bangunan, telepon, listrik, atau saluran yang digunakan untuk keperluan.	408	XXVII
6	Seseorang melakukan penggelapan barang	372, 373, 374, 375	XXIV

Query 1

Budi dengan sengaja dan dengan rencana terlebih dahulu merampas nyawa Ali. Dalam pembunuhan berencana ini, Budi melakukan penganiayaan berat terhadap Ali hingga mengakibatkan kematian.

- Pasal relevan berdasarkan pakar = 340, 338, dan 355.

$$Precision P = \frac{true\ positive}{true\ positive + false\ positive}$$

$$P\ k\text{-rank}\ 5\ \text{pasal}\ 355 = \frac{1}{1+0} = 1$$

$$P\ k\text{-rank}\ 5\ \text{pasal}\ 354 = \frac{1}{1+1} = 0.5$$

$$P\ k\text{-rank}\ 5\ \text{pasal}\ 338 = \frac{2}{2+1} = 0.6667$$

$$Recall R = \frac{true\ positive}{true\ positive + false\ negative}$$

$$R\ k\text{-rank}\ 5\ \text{pasal}\ 355 = \frac{1}{3} = 0.3333$$

$$R\ k\text{-rank}\ 5\ \text{pasal}\ 354 = \frac{1}{3} = 0.3333$$

$$R\ k\text{-rank}\ 5\ \text{pasal}\ 338 = \frac{2}{3} = 0.6667$$

Berikut merupakan hasil perhitungan *precision* dan *recall* serta perhitungan *Average Precision* dari 5 pasal teratas hasil perhitungan *cosine similarity* yang di



rekomendasikan oleh sistem temu kembali informasi pasal KUHP berdasarkan query 1 atau data uji 1, yaitu pada Tabel 5.2 sampai Tabel 5.8.

K-rank 5

Tabel 5.2 Average Precision query 1 k-rank 5

pasal	355	354	338	358	347
precision	1	0.5	0.6667	0.5	0.4
recall	0.3333	0.3333	0.6667	0.6667	0.6667

Average Precision query 1, k-rank 5 = $(1 + 0.6667)/3 = 0.5556$

K-rank 10

Tabel 5.3 Average Precision query 1 k-rank 10

pasal	355	351	354	353	359
precision	1	0.5	0.3333	0.25	0.2
recall	0.3333	0.3333	0.3333	0.3333	0.3333

Average Precision query 1, k-rank 10 = $(1)/3 = 0.3333$

K-rank 20

Tabel 5.4 Average Precision query 1 k-rank 20

pasal	343	355	353	338	341
precision	0	0.5	0.3333	0.5	0.4
recall	0	0.3333	0.3333	0.6667	0.6667

Average Precision query 1, k-rank 20 = $(0.5 + 0.5)/3 = 0.3333$

K-rank 30

Tabel 5.5 Average Precision query 1 k-rank 30

pasal	355	354	340	353	338
precision	1	0.5	0.6667	0.5	0.6
recall	0.3333	0.3333	0.6667	0.6667	1

Average Precision query 1, k-rank 30 = $(1 + 0.6667 + 1)/3 = 0.7556$

K-rank 40

Tabel 5.6 Average Precision query 1 k-rank 40

pasal	355	338	340	343	354
precision	1	1	1	0.75	0.6
recall	0.3333	0.6667	1	1	1

Average Precision query 1, k-rank 40 = $(1 + 1 + 1)/3 = 1$



K-rank 50

Tabel 5.7 Average Precision query 1 k-rank 50

pasal	340	338	354	355	350
precision	1	1	0.6667	0.75	0.6
recall	0.3333	0.6667	0.6667	1	1

Average Precision query 1, k-rank 50 = $(1 + 1 + 0.75)/3 = 0.9167$

K-rank 59

Tabel 5.8 Average Precision query 1 k-rank 59

pasal	338	350	348	354	355
precision	1	0.5	0.3333	0.25	0.4
recall	0.3333	0.3333	0.3333	0.3333	0.6667

Average Precision query 1, k-rank 59 = $(1 + 0.4)/3 = 0.4667$

Keterangan :

= pasal rekomendasi sistem yang relevan dengan pakar

Query 2

Seorang wanita yang sengaja menggugurkan atau mematikan kandungannya. Wanita ini menggugurkan atau mematikan kandungannya dengan persetujuan orang tuanya. Wanita ini dibantu oleh seorang dokter, bidan atau juru obat dalam melakukan pengguguran kandungan. Kemudian perbuatan menggugurkan kandungan itu mengakibatkan matinya wanita tersebut.

- Pasal relevan berdasarkan pakar = 346, 348, dan 349.

Berikut merupakan hasil perhitungan *precision* dan *recall* serta perhitungan *Average Precision* dari 5 pasal teratas hasil perhitungan *cosine similarity* yang di rekomendasikan oleh sistem temu kembali informasi pasal KUHP berdasarkan query 2 atau data uji 2, yaitu pada Tabel 5.9 sampai Tabel 5.15.

K-rank 5

Tabel 5.9 Average Precision query 2 k-rank 5

Pasal	346	347	376	370	348
Precision	1	0.5	0.3333	0.25	0.4
Recall	0.3333	0.3333	0.3333	0.3333	0.6667

Average Precision query 2, k-rank 5 = $(1 + 0.4)/3 = 0.4667$

K-rank 10

Tabel 5.10 Average Precision query 2 k-rank 10

pasal	348	354	355	347	353
precision	1	0.5	0.3333	0.25	0.2
recall	0.3333	0.3333	0.3333	0.3333	0.3333

Average Precision query 2, k-rank 10 = $(1)/3 = 0.3333$

K-rank 20

Tabel 5.11 Average Precision query 2 k-rank 20

pasal	346	347	348	355	338
precision	1	0.5	0.6667	0.5	0.4
recall	0.3333	0.3333	0.6667	0.6667	0.6667

Average Precision query 2, k-rank 20 = $(1 + 0.6667)/3 = 0.5556$

K-rank 30

Tabel 5.12 Average Precision query 2 k-rank 30

pasal	346	348	347	349	359
precision	1	1	0.6667	0.75	0.6
recall	0.3333	0.6667	0.6667	1	1

Average Precision query 2, k-rank 30 = $(1 + 1 + 0.75)/3 = 0.9167$

K-rank 40

Tabel 5.13 Average Precision query 2 k-rank 40

pasal	348	346	347	349	341
precision	1	1	0.6667	0.75	0.6
recall	0.3333	0.6667	0.6667	1	1

Average Precision query 2, k-rank 40 = $(1 + 1 + 0.75)/3 = 0.9167$

K-rank 50

Tabel 5.14 Average Precision query 2 k-rank 50

pasal	347	346	348	349	354
precision	0	0.5	0.6667	0.75	0.6
recall	0	0.3333	0.6667	1	1

Average Precision query 2, k-rank 50 = $(0.5 + 0.6667 + 0.75)/3 = 0.6389$



K-rank 59

Tabel 5.15 Average Precision query 2 k-rank 59

pasal	347	346	349	348	340
precision	0	0.5	0.6667	0.75	0.6
recall	0	0.3333	0.6667	1	1

$Average\ Precision\ query\ 2,\ k-rank\ 59 = (0.5 + 0.6667 + 0.75)/3 = 0.6389$

Keterangan :

= pasal rekomendasi sistem yang relevan dengan pakar

Query 3

Dua orang pencuri melakukan pencurian pada waktu malam dalam sebuah rumah yang tertutup, pelaku pencurian masuk ke dalam rumah dengan memanjat tembok kemudian merusak pintu. Untuk mempermudah pencurian, pencuri menusuk pemilik rumah hingga luka-luka berat. Pencuri kemudian melarikan diri bersama barang curian.

- Pasal relevan berdasarkan pakar = 365

Berikut merupakan hasil perhitungan *precision* dan *recall* serta perhitungan *Average Precision* dari 5 pasal teratas hasil perhitungan *cosine similarity* yang di rekomendasikan oleh sistem temu kembali informasi pasal KUHP berdasarkan *query 3* atau data uji 3, yaitu pada Tabel 5.16 sampai Tabel 5.22.

K-rank 5

Tabel 5.16 Average Precision query 3 k-rank 5

Pasal	365	364	341	342	408
precision	1	0.5	0.3333	0.25	0.2
Recall	1	1	1	1	1

$Average\ Precision\ query\ 3,\ k-rank\ 5 = 1/1 = 1$

K-rank 10

Tabel 5.17 Average Precision query 3 k-rank 10

Pasal	365	364	363	355	347
precision	1	0.5	0.3333	0.25	0.2
Recall	1	1	1	1	1

$Average\ Precision\ query\ 3,\ k-rank\ 10 = 1/1 = 1$

K-rank 20

Tabel 5.18 Average Precision query 3 k-rank 20

Pasal	365	364	363	373	407
precision	1	0.5	0.3333	0.25	0.2
Recall	1	1	1	1	1

Average Precision query 3, k-rank 20 = 1/1 = 1

K-rank 30

Tabel 5.19 Average Precision query 3 k-rank 30

pasal	364	365	373	362	372
precision	0	0.5	0.3333	0.25	0.2
recall	0	0	0	0	1

Average Precision query 3, k-rank 30 = 0.5/1 = 0.5

K-rank 40

Tabel 5.20 Average Precision query 3 k-rank 40

pasal	364	365	362	373	313
precision	0	0.5	0.3333	0.25	0.2
recall	0	0	0	1	1

Average Precision query 3, k-rank 40 = 0.5/1 = 0.5

K-rank 50

Tabel 5.21 Average Precision query 3 k-rank 50

pasal	364	362	365	411	372
precision	0	0	0.3333	0.25	0.2
recall	0	0	0	1	1

Average Precision query 3, k-rank 50 = 0.3333/1 = 0.3333

K-rank 59

Tabel 5.22 Average Precision query 3 k-rank 59

pasal	364	362	365	350	370
precision	0	0	0.3333	0.25	0.2
recall	0	0	1	1	1

Average Precision query 3, k-rank 59 = (0.3333)/1 = 0.3333



Keterangan :

 = pasal rekomendasi sistem yang relevan dengan pakar

Query 4

Angga melakukan fitnah kepada Bepriandi dengan melakukan tuduhan kepada Bepriandi tanpa bukti yang jelas, dan tuduhan yang dilakukan Angga tersebut bertentangan dengan apa yang dilakukan oleh Bepriandi, sehingga karena fitnah tersebut nama baik Bepriandi menjadi tercemar.

- Pasal relevan berdasarkan pakar = 311

Berikut merupakan hasil perhitungan *precision* dan *recall* serta perhitungan *Average Precision* dari 5 pasal teratas hasil perhitungan *cosine similarity* yang di rekomendasikan oleh sistem temu kembali informasi pasal KUHP berdasarkan *query 4* atau data uji 4, yaitu pada Tabel 5.23 sampai Tabel 5.29.

K-rank 5

Tabel 5.23 Average Precision query 4 k-rank 5

pasal	314	311	377	310	361
precision	0	0.5	0.3333	0.25	0.2
recall	0	1	1	1	1

Average Precision query 4, k-rank 5 = $0.5/1 = 0.5$

K-rank 10

Tabel 5.24 Average Precision query 4 k-rank 10

pasal	314	311	313	312	344
precision	0	0.5	0.3333	0.25	0.2
recall	0	1	1	1	1

Average Precision query 4, k-rank 10 = $0.5/1 = 0.5$

K-rank 20

Tabel 5.25 Average Precision query 4 k-rank 20

pasal	311	313	314	317	310
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 4, k-rank 20 = $1/1 = 1$

K-rank 30

Tabel 5.26 Average Precision query 4 k-rank 30

pasal	311	313	317	314	374
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 4, k-rank 30 = 1/1 = 1

K-rank 40

Tabel 5.27 Average Precision query 4 k-rank 40

pasal	311	313	317	314	310
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 4, k-rank 40 = 1/1 = 1

K-rank 50

Tabel 5.28 Average Precision query 4 k-rank 50

pasal	311	314	317	310	344
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 4, k-rank 50 = 1/1 = 1

K-rank 59

Tabel 5.29 Average Precision query 4 k-rank 59

pasal	311	317	314	344	310
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 4, k-rank 59 = 1/1 = 1

Keterangan :

= pasal rekomendasi sistem yang relevan dengan pakar

Query 5

Pendemo dengan sengaja menghancurkan merusakkan atau membikin tak dapat dipakai bangunan, telepon, listrik, atau saluran yang digunakan untuk keperluan.

- Pasal relevan berdasarkan pakar = 408

Berikut merupakan hasil perhitungan *precision* dan *recall* serta perhitungan *Average Precision* dari 5 hasil perhitungan *cosine similarity* pasal



teratas yang di rekomendasikan oleh sistem temu kembali informasi pasal KUHP berdasarkan *query* 5 atau data uji 5, yaitu pada Tabel 5.30 sampai Tabel 5.36.

K-rank 5

Tabel 5.30 Average Precision query 5 k-rank 5

pasal	354	338	408	355	358
precision	0	0	0.3333	0.25	0.2
recall	0	0	1	1	1

Average Precision query 5, k-rank 5 = $0.3333/1 = 0.3333$

K-rank 10

Tabel 5.31 Average Precision query 5 k-rank 10

pasal	408	406	410	409	312
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 5, k-rank 10 = $1/1 = 1$

K-rank 20

Tabel 5.32 Average Precision query 5 k-rank 20

pasal	408	409	410	406	312
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 5, k-rank 20 = $1/1 = 1$

K-rank 30

Tabel 5.33 Average Precision query 5 k-rank 30

pasal	408	409	410	406	312
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 5, k-rank 30 = $1/1 = 1$

K-rank 40

Tabel 5.34 Average Precision query 5 k-rank 40

pasal	408	409	406	312	410
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 5, k-rank 40 = $1/1 = 1$



K-rank 50

Tabel 5.35 Average Precision query 5 k-rank 50

pasal	408	409	406	312	410
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 5, k-rank 50 = 1/1 = 1

K-rank 59

Tabel 5.36 Average Precision query 5 k-rank 59

pasal	408	409	312	406	410
precision	1	0.5	0.3333	0.25	0.2
recall	1	1	1	1	1

Average Precision query 5, k-rank 59 = 1/1 = 1

Keterangan :

= pasal rekomendasi sistem yang relevan dengan pakar

Query 6

Seseorang melakukan penggelapan barang

- Pasal relevan berdasarkan pakar = 372, 373, 374, dan 375

Berikut merupakan hasil perhitungan *precision* dan *recall* serta perhitungan *Average Precision* dari 5 hasil perhitungan *cosine similarity* pasal teratas yang di rekomendasikan oleh sistem temu kembali informasi pasal KUHP berdasarkan *query 6* atau data uji 6, yaitu pada Tabel 5.37 sampai Tabel 5.43.

K-rank 5

Tabel 5.37 Average Precision query 6 k-rank 5

pasal	407	409	359	374	406
precision	0	0	0	0.25	0.2
recall	0	0	0	0.25	0.25

Average Precision query 6, k-rank 5 = 0.25/4 = 0.0625

K-rank 10

Tabel 5.38 Average Precision query 6 k-rank 10

pasal	373	372	345	352	375
precision	1	1	0.667	0.5	0.6
recall	0.25	0.5	0.5	0.5	0.75

Average Precision query 6, k-rank 10 = $1 + 1 + 0.6/4 = 0.65$

K-rank 20

Tabel 5.39 Average Precision query 6 k-rank 20

pasal	375	372	373	374	345
precision	1	1	1	1	0.8
recall	0.25	0.5	0.75	1	1

Average Precision query 6, k-rank 20 = $1 + 1 + 1 + 1/4 = 1$

K-rank 30

Tabel 5.40 Average Precision query 6 k-rank 30

pasal	374	372	375	373	364
precision	1	1	1	1	0.8
recall	0.25	0.5	0.75	1	1

Average Precision query 6, k-rank 30 = $1 + 1 + 1 + 1/4 = 1$

K-rank 40

Tabel 5.41 Average Precision query 6 k-rank 40

pasal	374	372	373	362	375
precision	1	1	1	0.75	0.8
recall	0.25	0.5	0.75	0.75	1

Average Precision query 6, k-rank 40 = $1 + 1 + 1 + 0,8/4 = 0,95$

K-rank 50

Tabel 5.42 Average Precision query 6 k-rank 50

pasal	373	372	374	375	354
precision	1	1	1	1	0.8
recall	0.25	0.5	0.75	1	1

Average Precision query 6, k-rank 50 = $1 + 1 + 1 + 1/4 = 1$

K-rank 59

Tabel 5.43 Average Precision query 6 k-rank 59

pasal	373	372	374	375	354
precision	1	1	1	1	0.8
recall	0.25	0.5	0.75	1	1

Average Precision query 6, k-rank 59 = $1 + 1 + 1 + 1/4 = 1$

Keterangan :



= pasal rekomendasi sistem yang relevan dengan pakar

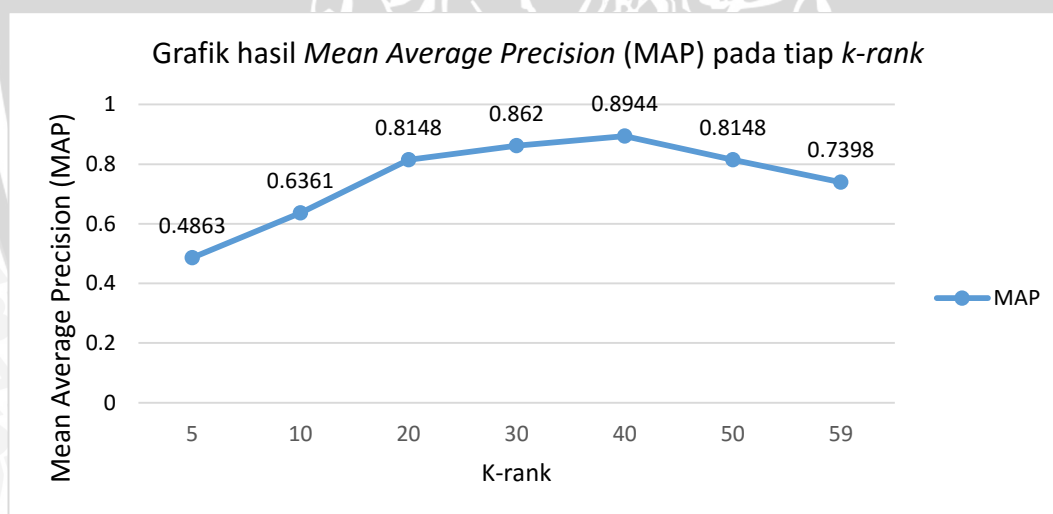
Berikut ini merupakan tabel hasil perhitungan nilai *Average Precision* dari 6 query atau kasus uji yang diujikan terhadap nilai beberapa *k-rank* yaitu 5, 10, 20, 30, 40, 50 dan 59. Kemudian nilai *Average Precision* dari masing-masing *k-rank* dirata-rata sehingga didapatkan nilai *Mean Average Precision* (MAP) dari masing-masing *k-rank*, yang dapat dilihat pada Tabel 5.44.

Tabel 5.44 Mean Average Precision

Query	k-rank						
	5	10	20	30	40	50	59
Q1	0.5556	0.3333	0.3333	0.7556	1	0.9167	0.4667
Q2	0.4667	0.3333	0.5556	0.9167	0.9167	0.6389	0.6389
Q3	1	1	1	0.5	0.5	0.3333	0.3333
Q4	0.5	0.5	1	1	1	1	1
Q5	0.3333	1	1	1	1	1	1
Q6	0.0625	0.65	1	1	0.95	1	1
MAP	0.4863	0.6361	0.8148	0.8620	0.8944	0.8148	0.7398

Keterangan :

= nilai MAP tertinggi



Gambar 5.1 Grafik hasil Mean Average Precision (MAP) pada tiap k-rank

5.3 Hasil Analisa

Analisa hasil dari nilai *Average Precision* (AP) dari masing-masing pengujian, tergantung dari nilai *k-rank* yang digunakan. Pada Gambar 5.1 menunjukkan bahwa akurasi sistem akan mengalami peningkatan jika nilai *k-rank* yang dimasukkan semakin besar, tetapi pada saat nilai *k-rank* melebihi nilai 40, performa sistem mengalami penurunan kembali.

Tabel 5.44 menunjukkan bahwa hasil dari evaluasi *Mean Average Precision* (MAP) pada masing-masing nilai *k-rank* yaitu 5, 10, 20, 30, 40, 50 dan 59. Pada Terlihat bahwa hasil *Mean Average Precision* (MAP) memiliki nilai yang rendah pada *k-rank* = 5 dengan nilai MAP = 0.4863. Sedangkan pada *k-rank* = 10, nilai MAP mengalami sedikit peningkatan, yaitu dengan nilai MAP = 0.6361. Terjadi peningkatan nilai MAP yang cukup signifikan yaitu dari MAP = 0.6361 pada *k-rank* =10, menjadi MAP = 0.8148 pada *k-rank*=20. Nilai MAP semakin membaik dan meningkat saat nilai semakin bertambah yaitu *k-rank* = 30, *k-rank* ini menghasilkan peningkatan nilai MAP yaitu dengan nilai MAP = 0.8620. Dari hasil analisis data diatas, nilai MAP pada *k-rank* 5, 10, 20, dan 30 kurang optimal. *K-rank* mereduksi dimensi fitur, semakin kecil nilai *k-rank* semakin besar data yang direduksi, sehingga pada *k-rank* 5, 10, 20, dan 30, banyak data atau informasi yang hilang akibat reduksi dimensi fitur. Reduksi dimensi fitur dapat menghilangkan *noice* pada data, tetapi juga dapat membuat informasi atau data yang dibutuhkan hilang.

Sistem menunjukkan peningkatan MAP yang terbaik dan tertinggi pada saat nilai *k-rank* bernilai 40 dengan nilai MAP yang hampir mendekati 1 dengan nilai MAP yaitu sebesar 0.8944. Saat nilai *k-rank* ditingkatkan menjadi 50 dan 59, sistem mengalami penurunan nilai MAP yang cukup signifikan yaitu menjadi 0.8148 dan 0.7398. Pada *k-rank* 50 dan 59, penambahan nilai *k-rank* memungkinkan untuk menghasilkan informasi yang lebih baik. Tetapi penambahan *k-rank* dapat mengurangi keakurasian informasi yang diberikan, karena masih terdapat banyak *noise* yang terdapat pada data. Sehingga pada penelitian ini, *k-rank* dengan nilai 40, sistem memiliki nilai MAP paling optimal, sehingga sistem ini dapat mengembalikan kebutuhan informasi yang dibutuhkan dengan baik dan akurat.



BAB 6 PENUTUP

6.1 Kesimpulan

1. Sistem yang dikembangkan dalam pencarian pasal KUHP berdasarkan kasus kejahatan pada penelitian ini menunjukkan hasil yang cukup baik, dimana nilai Mean Average Precision (MAP) yang dihasilkan mendekati nilai 1. Pada penelitian ini digunakan data berupa pasal pada Kitab Undang-Undang Hukum Pidana Buku Kedua tentang Kejahatan. Jika dimensi antar jumlah kata dan banyaknya pasal mempunyai jumlah yang besar maka waktu komputasi yang dihasilkan juga semakin lama. Sehingga setelah proses *preprocessing* pasal, digunakan proses reduksi dimensi *Singular Value Decomposition* (SVD) yang dapat mengurangi jumlah dimensi. Proses *Latent Semantic Indexing* (LSI) yang menggunakan SVD juga digunakan untuk mencari keterkaitan makna antar kata yang tersembunyi. Proses matematis dalam LSI mampu menunjukkan hubungan semantik antar kata. *Cosine similarity* digunakan untuk menemukan kesamaan atau kedekatan antar dua dokumen. Semakin besar nilai kesamaan *vector query* dengan *vector* dokumen maka *query* tersebut dipandang semakin relevan dengan dokumen.
2. Pada penelitian ini sistem menunjukkan nilai MAP yang terbaik dan tertinggi pada saat nilai *k-rank* bernilai 40 dengan nilai MAP yaitu sebesar 0.8944. Sehingga sistem ini dapat mengembalikan kebutuhan informasi yang dibutuhkan dengan baik dan akurat. Tetapi pemilihan *k-rank* yang optimal tidak dapat ditentukan secara pasti karena banyaknya jumlah kata dan dokumen yang berbeda akan memungkinkan untuk menghasilkan *k-rank* optimal yang berbeda pula.

6.2 Saran

Dari hasil penelitian ini, beberapa saran yang perlu ditambahkan dan diperbaiki untuk pengembangan sistem pencarian pasal KUHP lebih lanjut yaitu:

1. Perlu dilakukan perbandingan dalam perhitungan pembobotan kata menggunakan TF-IDF dengan metode pembobotan kata lainnya untuk menentukan pembobotan mana yang lebih cepat dan efisien.
2. Untuk meningkatkan akurasi dalam proses stemming perlu dilakukan perbandingan penggunaan stemming Apache Lucene dengan metode stemming lainnya seperti stemming Arifin dan lain-lain.

DAFTAR PUSTAKA

- Agusta, Ledy. 2009. *Perbandingan Algoritma Stemming Porter dengan Algoritma Nazief & Adriani untuk Stemming Dokumen Teks Bahasa Indonesia*. Fakultas Teknologi Informasi. Universitas Kristen Satya Wacana.
- Anton, H., & Rorres C. 2010 . *Elementary Linear Algebra: 9th edition* . John Wiley and Sons. New York
- The Apache Software Foundation, 2015. *Apache Lucene Core*. [online] Tersedia di: <<http://lucene.apache.org/core>> [Diakses 6 Juli 2015]
- Atiq ,Dian Astitul. 2015. *Indonesia Negara Hukum*. [online] Tersedia di: <http://academia.edu/6234004/Indonesia_Negara_Hukum> [Diakses 25 April 2015]
- Baker, Kirk. 2013. *Singular Value Decomposition Tutorial*. Department of Linguistics. The Ohio State University. Amerika Serikat
- Ferdian, Edward. Hadisaputra, Rian. Madjid, Nurkholis. 2013. *Penerapan Metode Latent Semantic Indexing Pada Search Engine*. Laboratorium Ilmu Dan Rekayasa Komputasi Departemen Teknik Informatika, Institut Teknologi Bandung. Bandung
- Feldman, Ronen., James Sanger. 2007. *The Text Mining Handbook*. Cambridge: Cambridge University Press
- Kontostathis, April. 2007. *Essential Dimensions of Latent Semantic Indexing (LSI)*. Departemen Matematika dan Ilmu Komputer Universitas Ursinus. USA
- Kozareva, Zornitsa. 2013. *Latent Semantic Analysis*. USC/ISI. Marina del Rey. California
- Mahendra, I Putu Adhi Kerta. 2008. *Penggunaan Algoritma Semut dan Conflif Stripping Stemmer untuk Klasifikasi Dokumen Berita Berbahasa Indonesia*. Teknik Informatika, Fakultas Teknologi Informasi, Universitas Institut teknologi Sepuluh Nopember. Surabaya
- Manning, Christoper.D, Raghavan, Prabhakar, dan Schutze, H. 2009. *An Introduction to Information Retrieval*. Cambridge.England.Cambridge University Press
- Nafik, Muhammad Zakiya. 2014. *Sistem Penilaian Otomatis Jawaban Esai Menggunakan Algoritma Levenshtein Distance*. Skripsi Teknik Informatika Universitas Brawijaya. Malang
- Nedunchelian, R., R. Muthucumarasamy, dan E. Saranathan. 2011. Comparison of Multi Document Summarization Techniques. *International Journal of Computer Applications*

- Nugroho, Eko. 2011. *Perancangan Sistem Deteksi Plagiarisme Dokumen Teks Dengan Menggunakan Algoritma Rabin-Karp*. Program studi Ilmu Komputer, Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Brawijaya. Malang.
- Parsons, Kathryn, McCormac, A., Butavicius, M, Dennis*, S, dan Ferguson, L. 2009. *The Use of Context-Based Information Retrieval Technique*. Australia. Defence Science and Technology Organization
- Peter, Rakesh, G, Shivapratap, Dvya G, dan Soman KP. 2009. *Evaluation of SVD and NMF Fungsis for Latent Semantic Analysis*. India. Amrita University
- Prasetyo, Teguh. 2014. *Hukum Pidana*. Jakarta: Rajawali Pers
- Republika. 2015. *Keadilan Nenek Asyani*. [online] Tersedia di: <<http://www.republika.co.id/berita/koran/opini-koran/15/03/18/nle9wq-keadilan-nenek-asyani>> [Diakses 18 Maret 2015]
- Samat, N.Ab, Murad, M.A.A, Abdullah, M.T, dan Atan, R. 2009. *Malay Documents Clustering Algorithm Based on Singular Value Decomposition*. Malaysia. Faculty of Computer Science and Information Technology. University Putra Malaysia.
- Sari, Yuita Arum. Achmad Ridok, Marji. 2012. *Penentuan Lirik Lagu Berdasarkan Emosi Menggunakan Sistem Temu Kembali Informasi Dengan Metode Latent Semantic Indexing (Lsi)*. Teknik Informatika, Institut Teknologi Sepuluh Nopember (ITS) Dan Program Teknik Informatika dan Ilmu Komputer, Surabaya dan Malang
- Schutze, Hinrich. 2011. *Introduction to Information Retrieval*. Institute for Natural Language Processing. University of Stuttgart : Jerman
- Sofyan, Herry. 2009. *Sistem Pencarian Citra Digital Menggunakan Content-Based*. Jurusan Teknik Informatika UPN Veteran. Yogyakarta.
- Sutisna, Utis. 2010. *Koreksi Ejaan Query bahasa Indonesia Menggunakan Algoritma Damerau Levenshtein*. Departemen Ilmu Komputer, IPB. Bogor
- Wati, Arrosyida Rizqa Buana. 2012. *Sistem Penilaian Otomatis Jawaban Essay Menggunakan Deteksi Similarity*. Program Studi Teknik Informatika. Fakultas Teknologi Industri. Universitas Pembangunan Nasional "Veteran". Jawa Timur
- Zafikri, Atika. 2008. *Implementasi Metode Term Frequency Inverse Document Frequency (TF-IDF) Pada Sistem Temu Kembali Informasi*. Tugas Akhir Program Studi Ilmu Komputer Fakultas Matematika Dan Ilmu Pengetahuan Alam Universitas Sumatera Utara : Medan

LAMPIRAN A PASAL KUHP

A.1 Data Training Pasal KUHP Buku Kedua Tentang Kejahatan

No	Bab	Pasal	Isi
1	IX	242	<p>(1) Barang siapa dalam keadaan di mana undang-undang menentukan supaya memberi keterangan di atas sumpah atau mengadakan akibat hukum kepada keterangan yang demikian, dengan sengaja memberi keterangan palsu di atas sumpah, baik dengan lisan atau tulisan, secara pribadi maupun oleh kuasanya yang khusus ditunjuk untuk itu, diancam dengan pidana penjara paling lama tujuh tahun.</p> <p>(2) Jika keterangan palsu di atas sumpah diberikan dalam perkara pidana dan merugikan terdakwa atau tersangka, yang bersalah diancam dengan pidana penjara paling lama sembilan tahun.</p> <p>(3) Disamakan dengan sumpah adalah janji atau penguatan yang diharuskan menurut aturan-aturan umum atau yang menjadi pengganti sumpah.</p> <p>(4) Pidana pencabutan hak berdasarkan pasal 35 No. 1 - 4 dapat dijatuhkan.</p>
2	XIX	338	Barang siapa dengan sengaja merampas nyawa orang lain, diancam karena pembunuhan dengan pidana penjara paling lama lima belas tahun.
3	XIX	339	Pembunuhan yang diikuti, disertai atau didahului oleh suatu perbuatan pidana, yang dilakukan dengan maksud untuk mempersiapkan atau mempermudah pelaksanaannya, atau untuk melepaskan diri sendiri maupun peserta lainnya dari pidana dalam hal tertangkap tangan, ataupun untuk memastikan penguasaan barang yang diperolehnya secara melawan hukum, diancam dengan pidana penjara seumur hidup atau selama waktu tertentu, paling lama dua puluh tahun.
4	XIX	340	Barangsiapa dengan sengaja dan dengan rencana terlebih dahulu merampas nyawa orang lain, diancam karena pembunuhan dengan rencana, dengan pidana mati atau pidana penjara seumur hidup atau selama waktu tertentu, paling lama dua puluh tahun.
5	XIX	341	Seorang ibu yang karena takut akan ketahuan melahirkan anak pada saat anak dilahirkan atau tidak lama kemudian, dengan sengaja merampas nyawa anaknya, diancam karena membunuh anak sendiri, dengan pidana penjara paling lama tujuh tahun.
6	XIX	342	Seorang ibu yang untuk melaksanakan niat yang ditentukan karena takut akan ketahuan bahwa ia akan melahirkan anak pada saat anak dilahirkan atau tidak lama kemudian merampas nyawa anaknya, diancam karena melakukan pembunuhan anak sendiri dengan rencana, dengan pidana penjara paling lama sembilan tahun.
7	XIX	343	Kejahatan yang diterangkan dalam pasal 341 dan 342 dipandang bagi orang lain yang turut serta melakukan, sebagai pembunuhan atau pembunuhan anak dengan rencana.
8	XIX	344	Barang siapa merampas nyawa orang lain atas permintaan orang itu sendiri yang jelas dinyatakan dengan kesungguhan hati, diancam dengan pidana penjara paling lama dua belas tahun.

No	Bab	Pasal	Isi
9	XIX	345	Barang siapa sengaja mendorong orang lain untuk bunuh diri, menolongnya dalam perbuatan itu atau memberi sarana kepadanya untuk itu, diancam dengan pidana penjara paling lama empat tahun kalau orang itu jadi bunuh diri.
10	XIX	346	Seorang wanita yang sengaja menggugurkan atau mematikan kandungannya atau menyuruh orang lain untuk itu, diancam dengan pidana penjara paling lama empat tahun.
11	XIX	347	(1) Barang siapa dengan sengaja menggugurkan atau mematikan kandungan seorang wanita tanpa persetujuannya, diancam dengan pidana penjara paling lama dua belas tahun. (2) Jika perbuatan itu mengakibatkan matinya wanita tersebut diancam dengan pidana penjara paling lama lima belas tahun.
12	XIX	348	(1) Barang siapa dengan sengaja menggugurkan atau mematikan kandungan seorang wanita dengan persetujuannya, diancam dengan pidana penjara paling lama lima tahun enam bulan. (2) Jika perbuatan itu mengakibatkan matinya wanita tersebut, diancam dengan pidana penjara paling lama tujuh tahun.
13	XIX	349	Jika seorang dokter, bidan atau juru obat membantu melakukan kejahatan berdasarkan pasal 346, ataupun melakukan atau membantu melakukan salah satu kejahatan yang diterangkan dalam pasal 347 dan 348, maka pidana yang ditentukan dalam pasal itu dapat ditambah dengan sepertiga dan dapat dicabut hak untuk menjalankan pencarian dalam mana kejahatan dilakukan.
14	XIX	350	Dalam hal pemidanaan karena pembunuhan, karena pembunuhan dengan rencana, atau karena salah satu kejahatan berdasarkan Pasal 344, 347 dan 348, dapat dijatuhkan pencabutan hak berdasarkan pasal 35 No. 1- 5.
15	XX	351	(1) Penganiayaan diancam dengan pidana penjara paling lama dua tahun delapan bulan atau pidana denda paling banyak empat ribu lima ratus rupiah, (2) Jika perbuatan mengakibatkan luka-luka berat, yang bersalah diancam dengan pidana penjara paling lama lima tahun. (3) Jika mengakibatkan mati, diancam dengan pidana penjara paling lama tujuh tahun. (4) Dengan penganiayaan disamakan sengaja merusak kesehatan. (5) Percobaan untuk melakukan kejahatan ini tidak dipidana.
16	XX	352	(1) Kecuali yang tersebut dalam pasal 353 dan 356, maka penganiayaan yang tidak menimbulkan penyakit atau halangan untuk menjalankan pekerjaan jabatan atau pencarian, diancam, sebagai penganiayaan ringan, dengan pidana penjara paling lama tiga bulan atau pidana denda paling banyak empat ribu lima ratus rupiah. Pidana dapat ditambah sepertiga bagi orang yang melakukan kejahatan itu terhadap orang yang bekerja padanya, atau menjadi bawahannya. (2) Percobaan untuk melakukan kejahatan ini tidak dipidana.
17	XX	353	(1) Penganiayaan dengan rencana lebih dahulu, diancam dengan pidana penjara paling lama empat tahun.

No	Bab	Pasal	Isi
			<p>(2) Jika perbuatan itu mengakibatkan luka-luka berat, yang bersalah dikenakan pidana penjara paling lama tujuh tahun.</p> <p>(3) Jika perbuatan itu mengakibatkan kematian yang bersalah diancam dengan pidana penjara paling lama sembilan tahun</p>
18	XX	354	<p>(1) Barang siapa sengaja melukai berat orang lain, diancam karena melakukan penganiayaan berat dengan pidana penjara paling lama delapan tahun.</p> <p>(2) Jika perbuatan itu mengakibatkan kematian, yang bersalah diancam dengan pidana penjara paling lama sepuluh tahun.</p>
19	XX	355	<p>(1) Penganiayaan berat yang dilakukan dengan rencana terlebih dahulu, diancam dengan pidana penjara paling lama dua belas tahun.</p> <p>(2) Jika perbuatan itu mengakibatkan kematian, yang bersalah diancam dengan pidana penjara paling lama lima belas tahun.</p>
20	XX	356	<p>Pidana yang ditentukan dalam pasal 351, 353, 354 dan 355 dapat ditambah dengan seperti</p> <ol style="list-style-type: none"> 1. bagi yang melakukan kejahatan itu terhadap ibunya, ayahnya yang sah, istrinya atau anaknya; 2. jika kejahatan itu dilakukan terhadap seorang pejabat ketika atau karena menjalankan tugasnya yang sah; 3. jika kejahatan itu dilakukan dengan memberikan bahan yang berbahaya bagi nyawa atau kesehatan untuk dimakan atau diminum.
21	XX	357	Dalam hal pemidanaan karena salah satu kejahatan berdasarkan pasal 353 dan 355, dapat dijatuhkan pencabutan hak berdasarkan pasal 30 No. 1 - 4.
22	XX	358	<p>Mereka yang sengaja turut serta dalam penyerangan atau perkelahian di mana terlibat beberapa orang, selain tanggung jawab masing-masing terhadap apa yang khusus dilakukan olehnya, diancam:</p> <ol style="list-style-type: none"> 1. dengan pidana penjara paling lama dua tahun delapan bulan, jika akibat penyerangan atau perkelahian itu ada yang luka-luka berat; 2. dengan pidana penjara paling lama empat tahun, jika akibatnya ada yang mati.
23	XXI	359	Barang siapa karena kesalahannya (kealpaannya) menyebabkan orang lain mati, diancam dengan pidana penjara paling lama lima tahun atau pidana kurungan paling lama satu tahun.
24	XXI	360	<p>(1) Barang siapa karena kesalahannya (kealpaannya) menyebabkan orang lain mendapat luka-luka berat, diancam dengan pidana penjara paling lama lima tahun atau pidana kurungan paling lama satu tahun.</p> <p>(2) Barang siapa karena kesalahannya (kealpaannya) menyebabkan orang lain luka-luka sedemikian rupa sehingga timbul penyakit atau halangan menjalankan pekerjaan jabatan atau pencarian selama waktu tertentu, diancam dengan pidana penjara paling lama sembilan bulan atau pidana kurungan paling lama enam bulan atau pidana denda paling tinggi empat ribu lima ratus rupiah.</p>

No	Bab	Pasal	Isi
25	XXI	361	Jika kejahatan yang diterangkan dalam bab ini dilakukan dalam menjalankan suatu jabatan atau pencarian, maka pidana ditambah dengan sepertiga dan yang bersalah dapat dicabut haknya untuk menjalankan pencarian dalam mana dilakukan kejahatan dan hakim dapat memerintahkan supaya putusannya diumumkan.
26	XXII	362	Barang siapa mengambil barang sesuatu, yang seluruhnya atau sebagian kepunyaan orang lain, dengan maksud untuk dimiliki secara melawan hukum, diancam karena pencurian, dengan pidana penjara paling lama lima tahun atau pidana denda paling banyak sembilan ratus rupiah.
27	XXII	363	(1) Diancam dengan pidana penjara paling lama tujuh tahun: <ol style="list-style-type: none"> 1. pencurian ternak; 2. pencurian pada waktu ada kebakaran, letusan, banjir gempa bumi, atau gempa laut, gunung meletus, kapal karam, kapal terdampar, kecelakaan kereta api, huru-hara, pemberontakan atau bahaya perang; 3. pencurian di waktu malam dalam sebuah rumah atau pekarangan tertutup yang ada rumahnya, yang dilakukan oleh orang yang ada di situ tidak diketahui atau tidak dikehendaki oleh yang berhak; 4. pencurian yang dilakukan oleh dua orang atau lebih; 5. pencurian yang untuk masuk ke tempat melakukan kejahatan, atau untuk sampai pada barang yang diambil, dilakukan dengan merusak, memotong atau memanjat, atau dengan memakai anak kunci palsu, perintah palsu atau pakaian jabatan palsu. <p>(2) Jika pencurian yang diterangkan dalam butir 3 disertai dengan salah satu hal dalam butir 4 dan 5, maka diancam dengan pidana penjara paling lama sembilan tahun.</p>
28	XXII	364	Perbuatan yang diterangkan dalam pasal 362 dan pasal 363 butir 4, begitu pun perbuatan yang diterangkan dalam pasal 363 butir 5, apabila tidak dilakukan dalam sebuah rumah atau pekarangan tertutup yang ada rumahnya, jika harga barang yang dicuri tidak lebih dari dua puluh lima rupiah, diancam karena pencurian ringan dengan pidana penjara paling lama tiga bulan atau pidana denda paling banyak dua ratus lima puluh rupiah.
29	XXII	365	(1) Diancam dengan pidana penjara paling lama sembilan tahun pencurian yang didahului, disertai atau diikuti dengan kekerasan atau ancaman kekerasan, terhadap orang dengan maksud untuk mempersiapkan atau mempermudah pencurian, atau dalam hal tertangkap tangan, untuk memungkinkan melarikan diri sendiri atau peserta lainnya, atau untuk tetap menguasai barang yang dicuri.

No	Bab	Pasal	Isi
			<p>3. jika masuk ke tempat melakukan kejahatan dengan merusak atau memanjat atau dengan memakai anak kunci palsu, perintah palsu atau pakaian jabatan palsu.</p> <p>4. jika perbuatan mengakibatkan luka-luka berat.</p> <p>(3) Jika perbuatan mengakibatkan kematian maka diancam dengan pidana penjara paling lama lima belas tahun.</p> <p>(4) Diancam dengan pidana mati atau pidana penjara seumur hidup atau selama waktu tertentu paling lama dua puluh tahun, jika perbuatan mengakibatkan luka berat atau kematian dan dilakukan oleh dua orang atau lebih dengan bersekutu, disertai pula oleh salah satu hal yang diterangkan dalam no. 1 dan 3.</p>
30	XXII	366	Dalam hal pemidanaan berdasarkan salah satu perbuatan yang dirumuskan dalam pasal 362, 363, dan 365 dapat dijatuhkan pencabutan hak berdasarkan pasal 35 No. 1 - 4.
31	XXII	367	<p>(1) Jika pembuat atau pembantu dari salah satu kejahatan dalam bab ini adalah suami (istri) dari orang yang terkena kejahatan dan tidak terpisah meja dan ranjang atau terpisah harta kekayaan, maka terhadap pembuat atau pembantu itu tidak mungkin diadakan tuntutan pidana.</p> <p>(2) Jika dia adalah suami (istri) yang terpisah meja dan ranjang atau terpisah harta kekayaan, atau jika dia adalah keluarga sedarah atau semenda, baik dalam garis lurus maupun garis menyimpang derajat kedua maka terhadap orang itu hanya mungkin diadakan penuntutan jika ada pengaduan yang terkena kejahatan.</p> <p>(3) Jika menurut lembaga matriarkal kekuasaan bapak dilakukan oleh orang lain daripada bapak kandung (sendiri), maka ketentuan ayat di atas berlaku juga bagi orang itu.</p>
32	XXIII	368	<p>(1) Barang siapa dengan maksud untuk menguntungkan diri sendiri atau orang lain secara melawan hukum, memaksa seorang dengan kekerasan atau ancaman kekerasan untuk memberikan barang sesuatu, yang seluruhnya atau sebagian adalah kepunyaan orang itu atau orang lain, atau supaya membuat hutang maupun menghapuskan piutang, diancam karena pemerasan dengan pidana penjara paling lama sembilan tahun.</p> <p>(2) Ketentuan pasal 365 ayat kedua, ketiga, dan keempat berlaku bagi kejahatan ini.</p>
33	XXIII	369	<p>(1) Barang siapa dengan maksud untuk menguntungkan diri sendiri atau orang lain secara melawan hukum, dengan ancaman pencemaran baik dengan lisan maupun tulisan, atau dengan ancaman akan membuka rahasia, memaksa seorang supaya memberikan barang sesuatu yang seluruhnya atau sebagian kepunyaan orang itu atau orang lain, atau supaya membuat hutang atau menghapuskan piutang, diancam dengan pidana penjara paling lama empat tahun.</p> <p>(2) Kejahatan ini tidak dituntut kecuali atas pengaduan orang yang terkena kejahatan.</p>
34	XXIII	370	Ketentuan pasal 367 berlaku bagi kejahatan-kejahatan yang di rumuskan dalam bab ini.

No	Bab	Pasal	Isi
35	XXIII	371	Dalam hal pemidanaan berdasarkan salah satu kejahatan yang dirumuskan dalam bab ini dapat dijatuhkan pencabutan hak berdasarkan pasal 35 no. 1 - 4.
36	XXIV	372	Barang siapa dengan sengaja dan melawan hukum memiliki barang sesuatu yang seluruhnya atau sebagian adalah kepunyaan orang lain, tetapi yang ada dalam kekuasaannya bukan karena kejahatan diancam karena penggelapan, dengan pidana penjara paling lama empat tahun atau pidana denda paling banyak sembilan ratus rupiah.
37	XXIV	373	Perbuatan yang dirumuskan dalam pasal 372 apabila yang digelapkan bukan ternak dan harganya tidak lebih dari dua puluh lima rupiah, diancam sebagai penggelapan ringan dengan pidana penjara paling lama tiga bulan atau pidana denda paling banyak dua ratus lima puluh rupiah.
38	XXIV	374	Penggelapan yang dilakukan oleh orang yang penguasaannya terhadap barang disebabkan karena ada hubungan kerja atau karena pencarian atau karena mendapat upah untuk itu, diancam dengan pidana penjara paling lama lima tahun.
39	XXIV	375	Penggelapan yang dilakukan oleh orang yang karena terpaksa diberi barang untuk disimpan, atau yang dilakukan oleh wali pengampu, pengurus atau pelaksana surat wasiat, pengurus lembaga sosial atau yayasan, terhadap barang sesuatu yang dikuasanya selaku demikian, diancam dengan pidana penjara paling lama enam tahun.
40	XXIV	376	Ketentuan dalam Pasal 367 berlaku bagi kejahatan-kejahatan yang dirumuskan dalam bab ini.
41	XXIV	377	(1) Dalam hal pemidanaan berdasarkan salah satu kejahatan yang dirumuskan dalam pasal 372, 374, dan 375 hakim dapat memerintahkan supaya putusan diumumkan dan dicabutnya hak-hak berdasarkan pasal 35 No. 1 4. (2) Jika kejahatan dilakukan dalam menjalankan pencarian maka dapat dicabut haknya untuk menjalankan pencarian itu.
42	XXVII	406	(1) Barangsiapa dengan sengaja dan melawan hukum menghancurkan, merusakkan, membikin tak dapat dipakai atau menghilangkan barang sesuatu yang seluruhnya atau sebagian milik orang lain, diancam dengan pidana penjara paling lama dua tahun delapan bulan atau pidana denda paling banyak empat ribu lima ratus rupiah. (2) Dijatuhkan pidana yang sama terhadap orang yang dengan sengaja dan melawan hukum membunuh, merusakkan, membikin tak dapat digunakan atau menghilangkan hewan, yang seluruhnya atau sebagian milik orang lain.
43	XXVII	407	(1) Perbuatan-perbuatan yang dirumuskan dalam pasal 406 jika harga kerugian tidak lebih dari dua puluh lima rupiah diancam dengan pidana penjara paling lama tiga bulan atau pidana denda paling banyak dua ratus lima puluh rupiah. (2) Jika perbuatan yang dirumuskan dalam pasal 406 ayat kedua itu dilakukan dengan memasukkan bahan-bahan yang merusakkan nyawa atau kesehatan, atau jika hewan itu termasuk dalam Pasal 101, maka ketentuan ayat pertama tidak berlaku.

No	Bab	Pasal	Isi
44	XXVII	408	Barang siapa dengan sengaja dan melawan hukum menghancurkan merusakkan atau membikin tak dapat dipakai bangunan-bangunan kereta api trem, telegram telepon atau listrik, atau bangunan saluran gas, air atau saluran yang digunakan untuk keperluan diancam dengan pidana penjara paling lama empat tahun.
45	XXVII	409	Barang siapa yang karena kesalahan (kealpaan) menyebabkan bangunan-bangunan tersebut dalam pasal di atas dihancurkan, dirusakkan atau dibikin tak dapat dipakai, diancam dengan pidana kurungan paling lama satu bulan atau pidana denda paling banyak seribu lima ratus rupiah.
46	XXVII	410	Barang siapa dengan sengaja dan melawan hukum menghancurkan atau membikin tak dapat dipakai suatu gedung atau kapal yang seluruhnya atau sebagian milik orang lain, diancam dengan pidana penjara paling lama lima tahun.
47	XXVII	411	Ketentuan pasal 367 diterapkan bagi kejahatan yang dirumuskan dalam bab ini.
48	XXVII	412	Jika salah satu kejahatan yang dirumuskan dalam bab ini dilakukan oleh dua orang atau lebih dengan bersekutu, maka pidana: ditambah sepertiga kecuali dalam hal yang dirumuskan pasal 407 ayat pertama.
49	XVI	310	(1) Barang siapa sengaja menyerang kehormatan atau nama baik seseorang dengan menuduhkan sesuatu hal, yang maksudnya terang supaya hal itu diketahui umum, diancam karena pencemaran dengan pidana penjara paling lama sembilan bulan atau pidana denda paling banyak empat ribu lima ratus rupiah. (2) Jika hal itu dilakukan dengan tulisan atau gambaran yang disiarkan, dipertunjukkan atau ditempelkan di muka umum, maka diancam karena pencemaran tertulis dengan pidana penjara paling lama satu tahun empat bulan atau pidana denda paling banyak empat ribu lima ratus rupiah. (3) Tidak merupakan pencemaran atau pencemaran tertulis, jika perbuatan jelas dilakukan demi kepentingan umum atau karena terpaksa untuk membela diri.
50	XVI	311	(1) Jika yang melakukan kejahatan pencemaran atau pencemaran tertulis dibolehkan untuk membuktikan apa yang dituduhkan itu benar, tidak membuktikannya, dan tuduhan dilakukan bertentangan dengan apa yang diketahui, maka dia diancam melakukan fitnah dengan pidana penjara paling lama empat tahun. (2) Pencabutan hak-hak berdasarkan pasal 35 No. 1 - 3 dapat dijatuhkan.
51	XVI	312	Pembuktian akan kebenaran tuduhan hanya dibolehkan dalam hal-hal berikut: <ol style="list-style-type: none"> 1. apabila hakim memandang perlu untuk memeriksa kebenaran itu guna menimbang keterangan terdakwa, bahwa perbuatan dilakukan demi kepentingan umum, atau karena terpaksa untuk membela diri; 2. apabila seorang pejabat dituduh sesuatu hal dalam menjalankan tugasnya.

No	Bab	Pasal	Isi
52	XVI	313	Pembuktian yang dimaksud dalam pasal 312 tidak dibolehkan, jika hal yang dituduhkan hanya dapat dituntut atas pengaduan dan pengaduan tidak dimajukan.
53	XVI	314	(1) Jika yang dihina, dengan putusan hakim yang menjadi tetap, dinyatakan bersalah atas hal yang dituduhkan, maka pemidanaan karena fitnah tidak mungkin. (2) Jika dia dengan putusan hakim yang menjadi tetap dibebaskan dari hal yang dituduhkan, maka putusan itu dipandang sebagai bukti sempurna bahwa hal yang dituduhkan tidak benar. (3) Jika terhadap yang dihina telah dimulai penuntutan pidana karena hal yang dituduhkan padanya, maka penuntutan karena fitnah dihentikan sampai mendapat putusan yang menjadi tetap tentang hal yang dituduhkan.
54	XVI	315	Tiap-tiap penghinaan dengan sengaja yang tidak bersifat pencemaran atau pencemaran tertulis yang dilakuknn terhadap seseorang, baik di muka umum dengan lisan atau tulisan, maupun di muka orang itu sendiri dengan lisan atau perbuatan, atau dengan surat yang dikirimkan stau diterimakan kepadanya, diancam karena penghinaan ringan dengan pidana penjara paling lama empat bulan dua minggu atau pidana denda paling banyak empat ribu lima ratus rupiah.
55	XVI	316	Pidana yang ditentukan dalam pasal-pasal sebelumnya dalam bab ini, dapat ditambah dengan sepertiga jika yang dihina adalah seorang pejabat pada waktu atau karena menjalankan tugasnya yang sah.
56	XVI	317	(1) Barang siapa dengan sengaja mengajukan pengaduan atau pemberitahuan palsu kepada penguasa, baik secara tertulis maupun untuk dituliskan, tentang seseorang sehingga kehormatan atau nama baiknya terserang, diancam karena melakukan pengaduan fitnah, dengan pidana penjara paling lama empat tahun, (2) Pencabutan hak-hak berdasarkan pasal 35 No, 1 - 3 dapat dijatuhkan.
57	XVI	318	(1) Barang siapa dengan sesuatu perbuatan sengaja menimbulkan secara palsu persangkaan terhadap seseorang bahwa dia melakukan suatu perbuatan pidana, diancam karena menimbulkan persangkaan palsu, dengan pidana penjara paling lama empat tahun. (2) Pencabutan hak-hak berdasarkan pasal 35 No. 1 - 3 dapat dijatuhkan.
58	XVI	319	Penghinaan yang diancam dengan pidana menurut bab ini, tidak dituntut jika tidak ada pengaduan dari orang yang terkena kejahatan itu, kecuali berdasarkan pasal 316.
59	XVI	320	(1) Barang siapa terhadap seseorang yang sudah mati melakukan perbuatan yang kalau orang itu masih hidup akan merupakan pencemaran atau pencemaran tertulis, diancam dengan pidana penjara paling lama empat bulan dua minggu atau pidana denda paling banyak empat ribu lima ratus rupiah. (2) Kejahatan ini tidak dituntut kalau tidak ada pengaduan dari salah seorang keluarga sedarah maupun semenda dalam garis lurus atau menyimpang sampai derajat kedua dari yang mati itu, atau atas pengaduan suami (istri) nya.

No	Bab	Pasal	Isi
			(3) Jika karena lembaga matriarkal kekuasaan bapak dilakukan oleh orang lain daripada bapak, maka kejahatan juga dapat dituntut atas pengaduan orang itu.
60	XVI	321	<p>(1) Barang siapa menyiarkan, mempertunjukkan atau menempelkan di muka umum tulisan atau gambaran yang isinya menghina atau bagi orang ymg sudah mati mencemarkan namanya, dengan maksud supaya isi surat atau gambar itu ditahui atau lehih diketahui oleh umum, diancam dengan pidana penjara paling lama satu hulan dua minggu atau pidana denda paling banyak empat ribu lima ratus rupiah.</p> <p>(2) Jika Yang bersalah melakukan kejahatan tersebut dalam menjalankan pencariannya, sedangkan ketika itu belum lampau dua tahun sejak adanya pembedanaan yang menjadi tetap karena kejahatan semacam itu juga, maka dapat dicabut haknya untuk menjalankan pencarian tersebut.</p> <p>(3) Kejahatan ini tidak dituntut kalau tidak ada pengaduan dari orang yang ditunjuk dalam pasal 319 dan pasal 320, ayat kedua dan ketiga.</p>

A.2 Data Uji Deskripsi Kasus (*Query*)

No	Deskripsi	Pasal Relevan	Bab
1	Budi dengan sengaja dan dengan rencana terlebih dahulu merampas nyawa Ali. Dalam pembunuhan berencana ini, Budi melakukan penganiayaan berat terhadap Ali hingga mengakibatkan kematian.	340, 338 355	XIX XX
2	Seorang wanita yang sengaja menggugurkan atau mematikan kandungannya. Wanita ini menggugurkan atau mematikan kandungannya dengan persetujuan orang tuanya. Wanita ini dibantu oleh seorang dokter, bidan atau juru obat dalam melakukan pengguguran kandungan. Kemudian perbuatan menggugurkan kandungan itu mengakibatkan matinya wanita tersebut.	346, 348, 349	XIX
3	Dua orang pencuri melakukan pencurian pada waktu malam dalam sebuah rumah yang tertutup, pelaku pencurian masuk ke dalam rumah dengan memanjat tembok kemudian merusak pintu. Untuk mempermudah pencurian, pencuri menusuk pemilik rumah hingga luka-luka berat. Pencuri kemudian melarikan diri bersama barang curian.	365	XXII
4	Angga melakukan fitnah kepada Bepriandi dengan melakukan tuduhan kepada Bepriandi tanpa bukti yang jelas, dan tuduhan yang dilakukan Angga tersebut bertentangan dengan apa yang dilakukan oleh Bepriandi, sehingga karena fitnah tersebut nama baik Bepriandi menjadi tercemar.	311	XVI
5	Pendemo dengan sengaja menghancurkan merusakkan atau membikin tak dapat dipakai bangunan, telepon, listrik, atau saluran yang digunakan untuk keperluan.	408	XXVII
6	Seseorang melakukan penggelapan barang	372, 373, 374, 375	XXIV

LAMPIRAN B SOURCE CODE

B.1 Tokenizing

```
public void tokenizing(String kata) {
    try {
        Pattern p = Pattern.compile("[^a-zA-Z\\s]+");
        Matcher m = p.matcher(kata);
        if (m.find()) {
            kata = kata.replaceAll(m.pattern().toString(), " ");
        }
        kata = kata.replaceAll("\n", " ");
        stemming(kata);
        System.out.println("Done");
    } catch (Exception e) {
        System.out.println("\n error");
    }
}
```

B.2 Stemming

```
public void preproses(String kata) {
    try {
        IndonesianAnalyzer ai = new IndonesianAnalyzer();
        QueryParser qp = new QueryParser("", ai);
        String[] keys = qp.parse(kata).toString().split(" ");
        tf(keys);
    } catch (ParseException ex) {

    }

    Logger.getLogger(CariPasal.class.getName()).log(Level.SEVERE,
    null, ex);
}
}
```

B.3 Term Frequency (TF)

```
public void tf(String[] keys) {
    String[] uniqueKeys;
    int counttf = 0;
    uniqueKeys = getUniqueKeys(keys);
    for (String key : uniqueKeys) {
        if (null == key) {
            break;
        }
        for (String s : keys) {
            if (key.equals(s)) {
                counttf++;
            }
        }
        System.out.println("Jumlah kata [" + key + "] : " +
        counttf);
        insert(key, counttf);
        counttf = 0;
    }
}
```

B.4 getUniqueKeys

```
private static String[] getUniqueKeys(String[] keys) {
    String[] uniqueKeys = new String[keys.length];
    uniqueKeys[0] = keys[0];
    int uniqueKeyIndex = 1;
    boolean keyAlreadyExists = false;
    for (int i = 1; i < keys.length; i++) {
        for (int j = 0; j <= uniqueKeyIndex; j++) {
            if (keys[i].equals(uniqueKeys[j])) {
                keyAlreadyExists = true;
            }
        }
        if (!keyAlreadyExists) {
            uniqueKeys[uniqueKeyIndex] = keys[i];
            uniqueKeyIndex++;
        }
        keyAlreadyExists = false;
    }
    return uniqueKeys;
}
```

B.5 IDF

```
public void setIDF() {
    int N = 0, df = 0;
    double idf;
    String pasal = null, kata = null;
    int col = model2.getColumnCount();
    int row = model2.getRowCount();
    System.out.println(col + " " + row);
    List<Object> idfList = new ArrayList<>();
    for (int i = 0; i < row; i++) {
        for (int j = 1; j < col; j++) {
            N++;
            if (Double.parseDouble(model2.getValueAt(i,
j).toString()) > 0) {
                df++;
            }
        }
        idf = Math.log10(N) - Math.log10(df);
        if (N == 0 || df == 0) {
            idf = 0;
        }
        idfList.add(idf);
        insertidf(model2.getValueAt(i, 0).toString(), idf);
        N = 0;
        df = 0;
    }
    model2.addColumn("IDF", idfList.toArray());
    System.out.println("IDF OK\n");
}
```

B.6 TF-IDF

```

public void setTFIDF() {
    readkatatfidfl();
    double tfidf;
    int col = model2.getColumnCount();
    int row = model2.getRowCount();
    double tf, idf;
    for (int i = 1; i < col - 1; i++) {
        String label = model2.getColumnName(i);
        modeltfidf.addColumn(label);
    }
    for (int i = 0; i < row; i++) {
        for (int j = 1; j < col - 1; j++) {
            tf = Double.parseDouble(model2.getValueAt(i,
j).toString());
            idf = Double.parseDouble(model2.getValueAt(i,
model2.getColumnCount() - 1).toString());
            tfidf = tf * idf;
            modeltfidf.setValueAt(tfidf, i, j);
            //SAVE DATA TFIDF
            inserttfidf(model2.getColumnName(j),
model2.getValueAt(i, 0).toString(), tfidf);
            System.out.println(model2.getColumnName(j));
            System.out.println(model2.getValueAt(i,
0).toString() + "\n");
        }
    }
    System.out.println("TF-IDF OK\n");
}

```

B.7 Singular Value Decomposition (SVD)

```

public void setSVD() {
    int col = modeltfidf.getColumnCount();
    int row = modeltfidf.getRowCount();
    double tfidf;
    double matriks_tfidf[][] = new double[row][col - 1];

    for (int i = 0; i < row; i++) {
        for (int j = 1; j < col; j++) {
            tfidf = Double.parseDouble(modeltfidf.getValueAt(i,
j).toString());
            matriks_tfidf[i][j - 1] = tfidf;
            System.out.println("GET TF IDF= " + i + j + " " +
modeltfidf.getValueAt(i, j).toString());
        }
    }
    System.out.println("SET SVD ON WORKING");

    S(matriks_tfidf);
    VT(matriks_tfidf);
    U(matriks_tfidf);

    System.out.println("SVD SELESAI");
}

```

B.8 Matriks S

```

public void S(double matriks_tfidf[][]) {
    RealMatrix n = new Array2DRowRealMatrix(matriks_tfidf);
    SingularValueDecomposition svss = new
SingularValueDecomposition(n);
    try {
        RealMatrix S = svss.getS();
        //GET S
        double SM[][] = new
double[S.getRowDimension()][S.getColumnDimension()];
        System.out.println("Hasil Perhitungan Singular Value");
        System.out.println("-----");
        //MEMASUKKAN DATA S KE ARRAY
        for (int i = 0; i < S.getRowDimension(); i++) {
            for (int j = 0; j < S.getColumnDimension(); j++) {
                System.out.print(S.getEntry(i, j));
                System.out.print(" ");
                SM[i][j] = S.getEntry(i, j);
            }
            System.out.println();
        }
        System.out.println();
        //SET TABEL KOLOM
        int col = S.getColumnDimension();
        int row = S.getRowDimension();
        model_s.addColumn("");
        for (int j = 0; j < col; j++) {
            model_s.addColumn(j);
        }
        //SET TABEL BARIS
        for (int i = 0; i < row; i++) {
            List<Object> al = new ArrayList<>();
            al.add(i);
            model_s.addRow(al.toArray());
        }
        //SET ISI TABEL
        for (int i = 0; i < row; i++) {
            for (int j = 0; j < col; j++) {
                model_s.setValueAt(SM[i][j], i, j + 1);
            }
        }
    } catch (Exception e) {
        System.out.println("err");
    }
}

```

B.9 Matriks Vt

```

public void VT(double matriks_tfidf[][]) {
    RealMatrix n = new Array2DRowRealMatrix(matriks_tfidf);
    SingularValueDecomposition svss = new
SingularValueDecomposition(n);

    try {
        //GET V
        RealMatrix V = svss.getVT();
    }
}

```

```

        double                VM[][]                =                new
double[V.getRowDimension()][V.getColumnDimension()];
System.out.println("Hasil Perhitungan VT");
System.out.println("-----");
for (int i = 0; i < V.getRowDimension(); i++) {
    for (int j = 0; j < V.getColumnDimension(); j++) {
        System.out.print(V.getEntry(i, j));
        System.out.print(" ");
        VM[i][j] = V.getEntry(i, j);
    }
    System.out.println();
}
System.out.println();
//SET TABEL KOLOM
int col = V.getColumnDimension();
int row = V.getRowDimension();
model_v.addColumn("");
for (int j = 0; j < col; j++) {
    model_v.addColumn(j);
}
//SET TABEL BARIS
for (int i = 0; i < row; i++) {
    List<Object> al = new ArrayList<>();
    al.add(i);
    model_v.addRow(al.toArray());
}
//SET ISI TABEL
for (int i = 0; i < row; i++) {
    for (int j = 0; j < col; j++) {
        model_v.setValueAt(VM[i][j], i, j + 1);
    }
}
} catch (Exception e) {
    System.out.println("err");
}
}

```

B.10 Matriks U

```

public void U(double matriks_tfidf[][]) {
    RealMatrix n = new Array2DRowRealMatrix(matriks_tfidf);
    SingularValueDecomposition svss = new
SingularValueDecomposition(n);
    try {
        //GET U
        RealMatrix U = svss.getU();
        double                UM[][]                =                new
double[U.getRowDimension()][U.getColumnDimension()];
        System.out.println("Hasil Perhitungan U");
        System.out.println("-----");
        for (int i = 0; i < U.getRowDimension(); i++) {
            for (int j = 0; j < U.getColumnDimension(); j++) {
                System.out.print(U.getEntry(i, j));
                System.out.print(" ");
                UM[i][j] = U.getEntry(i, j);
            }
        }
    }
}

```

```

        System.out.println();
    }
    System.out.println();
    //SET TABEL KOLOM
    int col = U.getColumnDimension();
    int row = U.getRowDimension();
    model_u.addColumn("");
    for (int j = 0; j < col; j++) {
        model_u.addColumn(j);
    }
    //SET TABEL BARIS
    for (int i = 0; i < row; i++) {
        List<Object> al = new ArrayList<>();
        al.add(i);
        model_u.addRow(al.toArray());
    }
    //SET ISI TABEL
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            model_u.setValueAt(UM[i][j], i, j + 1);
        }
    }
} catch (Exception e) {
    System.out.println("err");
}
}

```

B.11 Latent Semantic Indexing (LSI)

```

public void LSI(String krank_vall) {
    TF_2.setText(krank_vall);
    System.out.println("nilai k-rank= " + krank_vall);
    //menggambil nilai S krank
    double[][] sk = SVD_SK(krank_vall);
    //menghitung inverse S krank dengan merubah terlebih dahulu
    kedalam matrix (lib apache)
    RealMatrix SI = new Array2DRowRealMatrix(sk);
    RealMatrix SInverse = MatrixUtils.inverse(SI);
    //NILAI Sk Inverse
    double[][] sk_inverse = SInverse.getData();
    //menggambil nilai U krank
    double[][] uk = SVD_UK(krank_vall);
    //menggambil nilai V krank Transpose
    double[][] vkt = SVD_VTK(krank_vall);
    //menTranspose nilai V krank Transpose
    double[][] vk = Transpose(vkt);
    //menampilkan tabel VK vektor dokumen
    get_Vk_vector_tabel(vk);
    double[][] query = getQueryVal();
    //hitung LSI LSI= query x Uk x Sk_inverse
    double[][] result = multiplicar(query, uk);
    //HASIL PERHITUNGAN LSI
    double[][] resultLSI = multiplicar(result, sk_inverse);
    //ambil vector Query LSI ke single array
    double[] queryvec = new double[resultLSI[0].length];
    for (int i = 0; i < resultLSI.length; i++) {
        for (int j = 0; j < resultLSI[0].length; j++) {

```

```

        double val = resultLSI[i][j];
        queryvec[j] = val;
        System.out.println("queryvec [" + j + "] " +
queryvec[j]);
    }
}
//ambil DOC VECTOR dari matriks Vk
RealMatrix docvec = new Array2DRowRealMatrix(vk);
//inisialisasi matriks
double[][] docvector = new
double[docvec.getRowDimension()][docvec.getColumnDimension()];
for (int i = 0; i < docvec.getRowDimension(); i++) {
    for (int j = 0; j < docvec.getColumnDimension(); j++) {
        docvector[i][j] = docvec.getEntry(i, j);
    }
}
//SET TABLE COSINE
set_Table_cosine();
//inisialisasi single array untuk penempatan nilai docvec
setiap dokumen
double Array[] = new double[Integer.parseInt(krank_vall)];
//GET VECTOR DOC to Single Array
for (int g = 0; g < docvector.length; g++) {
    for (int h = 0; h < docvector[0].length; h++) {
        Array[h] = docvector[g][h];
        System.out.println("docvector[" + g + "][" + h + "]
" + Array[h]);
    }
    //Perhitungan nilai Cosine Similarity
    double cosine = CosineSimilarity(queryvec, Array);
//COUNT COSINE
model_cosine.setValueAt(cosine, g, 2); //set value
cosine to tabel
Arrays.fill(Array, 0); //fill arrays to 0
System.out.println("\n");
}
//Insert COSIN to database for sorting
insert_cosine_similarity();
read_sort_Cosine();
}

```

B.12 Matriks S K-rank

```

public double[][] SVD_SK(String krank_vall) {
    int krank = Integer.parseInt(krank_vall);
    System.out.println(krank);
    double svd_sk[][] = new double[krank][krank];
    for (int i = 0; i <= krank - 1; i++) {
        for (int j = 1; j <= krank; j++) {
            svd_sk[i][j] - 1] =
Double.parseDouble(model_s.getValueAt(i, j).toString());
            System.out.println("SVD S Krank-[" + i + "][" + (j -
1) + "] = " + svd_sk[i][j - 1]);
        }
    }
    RealMatrix n = new Array2DRowRealMatrix(svd_sk);
    for (int k = 0; k < n.getColumnDimension(); k++) {

```



```

        model_sk.addColumn(k);
    }
    for (int i = 0; i < n.getRowDimension(); i++) {
        model_sk.addRow(svd_sk);
        for (int j = 0; j < n.getColumnDimension(); j++) {
            Object vall = n.getEntry(i, j);
            model_sk.setValueAt(vall, i, j);
        }
    }
    return svd_sk;
}

```

B.13 Matriks U K-rank

```

public double[][] SVD_UK(String krank_val) {
    int krank = Integer.parseInt(krank_val);
    System.out.println(krank);
    System.out.println(model_u.getColumnCount() + "\n");
    double svd_uk[][] = new double[model_u.getRowCount()][krank];
    for (int i = 0; i <= model_u.getRowCount() - 1; i++) {
        for (int j = 1; j <= krank; j++) {
            svd_uk[i][j] = Double.parseDouble(model_u.getValueAt(i, j).toString());
            System.out.println("SVD U Krank-" + i + "][" + (j - 1) + "] = " + svd_uk[i][j - 1]);
        }
    }
    RealMatrix n = new Array2DRowRealMatrix(svd_uk);
    for (int k = 0; k < n.getColumnDimension(); k++) {
        model_uk.addColumn(k);
    }
    for (int i = 0; i < n.getRowDimension(); i++) {
        model_uk.addRow(svd_uk);
        for (int j = 0; j < n.getColumnDimension(); j++) {
            Object vall = n.getEntry(i, j);
            model_uk.setValueAt(vall, i, j);
        }
    }
    return svd_uk;
}

```

B.14 Matriks VT K-rank

```

public double[][] SVD_VTK(String krank_val) {
    int krank = Integer.parseInt(krank_val);
    System.out.println(krank);
    System.out.println(model_v.getColumnCount() + "\n");
    double svd_vk[][] = new double[krank][model_v.getColumnCount() - 1];
    for (int i = 0; i < krank; i++) {
        for (int j = 1; j < model_v.getColumnCount(); j++) {
            svd_vk[i][j] = Double.parseDouble(model_v.getValueAt(i, j).toString());
        }
    }
}

```

```

        System.out.println("SVD VT Krank-[" + i + "]"[" + (j
- 1) + "] = " + svd_vk[i][j - 1]);
    }
}
RealMatrix n = new Array2DRowRealMatrix(svd_vk);
for (int k = 0; k < n.getColumnDimension(); k++) {
    model_vk.addColumn(k);
}
for (int i = 0; i < n.getRowDimension(); i++) {
    model_vk.addRow(svd_vk);
    for (int j = 0; j < n.getColumnDimension(); j++) {
        Object vall = n.getEntry(i, j);
        model_vk.setValueAt(vall, i, j);
    }
}
return svd_vk;
}

```

B.15 Cosine Similarity

```

public double CosineSimilarity(double[] queryvec, double[]
docvec) {
    int col = queryvec.length;
    int coldoc = docvec.length;
    double query_sqrt = 0, vector_sqrt = 0, cosine_similarity =
0, dot = 0;
    if (col == coldoc) {
        for (int i = 0; i < col; i++) {
            dot = dot + (queryvec[i] * docvec[i]);
        }
        for (int i = 0; i < col; i++) {
            query_sqrt = query_sqrt + (queryvec[i] *
queryvec[i]);
            vector_sqrt = vector_sqrt + (docvec[i] * docvec[i]);
        }
        query_sqrt = Math.sqrt(query_sqrt);
        vector_sqrt = Math.sqrt(vector_sqrt);
        //Perhitungan Cosine Similarity
        cosine_similarity = dot / (query_sqrt * vector_sqrt);
        System.out.println("COSINE SIMILARITY =" +
cosine_similarity);
    } else {
        System.err.println("Cosine Similarity: queryvec tidak
sama dgn docvec");
    }
    return cosine_similarity;
}

```