THESIS

DETEKSI SPAM TWEET PADA TWITTER DENGAN METODE NAIVE BAYES



Oleh:

Budi Santoso

212210345

PROGRAM STRATA-2

JURUSAN TEKNOLOGI INFORMASI

SEKOLAH TINGGI TEKNIK SURABAYA

SURABAYA

2018

THESIS

DETEKSI SPAM TWEET PADA TWITTER DENGAN METODE NAIVE BAYES

Diajukan Guna Memenuhi Sebagian Persyaratan Untuk Memperoleh Gelar

Magister Komputer

Pada

Sekolah Tinggi Teknik Surabaya

Disetujui oleh Tim Penguji Tugas Akhir:

1. Esther Irawati Setiawan, S.Kom, M.Kom. (Pembimbing)
2. (Penguji I)
3. (Penguji II)
4. (Penguji III)

SURABAYA

APRIL 2018

SURAT PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama : Budi Santoso

Program Studi : Teknologi Informasi

NRP : 212210345

dengan ini menyatakan bahwa isi sebagian maupun keseluruhan Tesis dengan judul:

DETEKSI SPAM TWEET PADA TWITTER

DENGAN METODE NAIVE BAYES

adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diizinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 01-09-2018

Yang Membuat Pernyataan,

Budi Santoso

212210345

ABSTRAK

Dewasa ini sosial media dan microblogging merupakan sarana komunikasi yang banyak digunakan. Selain sebagai sarana untuk komunikasi personal, twitter juga digunakan untuk menawarkan produk barang maupun jasa. Penggunaan seperti ini mengakibatkan berkembangnya spam pada jaringan twitter. Penelitian ini dilakukan untuk mengembangkan filter spam pada twitter dengan mengandalkan pada analisa teks pada twit berbahasa indonesia. Dataset diperoleh dari twitter API dengan melakukan pencarian terhadap twit-twit menggunakan kata kunci yang didapat dari daftar trending topic untuk daerah indonesia.

Klasifikasi dengan metode naive dipilih untuk mendeteksi twit-twit spam. Metode Naive Bayes digunakan karena teruji cukup baik untuk klasifikasi teks dalam beberapa penelitian terkait yang telah ada. Fitur utama yang digunakan diuji adalah unigram/bag-of-words dan bigram dengan teknik smoothing stupid backoff. Hasil klasifikasi dengan teknik smoothing stupid backoff ini dibandingkan dengan klasifikasi Naive Bayes dengan salah satu dari unigram atau bigram dengan teknik smoothing laplace.

Ujicoba dilakukan dengan metode k-fold cross validation, dengan 10 kali uji silang. Pada pengujian dibandingkan klasifikasi dengan fitur unigram, fitur bigram, unigram dan bigram dengan perhitungan term frequency dan inverse doument frequency, serta klasifikasi dengan fitur unigram dan bigram sekaligus dengan menggunakan teknik smoothing stupid backoff. Dari hasil pengujian tampak bahwa klasifikasi dengan fitur unigram mempunyai kemampuan deteksi spam terendah dengan tingkat deteksi 87,5%. Akan tetapi klasifikasi metode ini mempunyai tingkat kesalahan penggolongan ham ke spam yang juga paling rendah yaitu di bawah 1 %. Perhitungan term dan inverse document frequency bisa meningkatkan tingkat deteksi sekitar 4% lebih tinggi. Klasifikasi dengan fitur bigram mempunyai tingkat deteksi spam tertinggi yaitu hingga 95%, akan tetapi juga mempunyai tingkat kesalahan deteksi yang tinggi juga yatu hingga 15%. Klasifikasi terbaik pada penggunaan teknik smoothing stupid backoff, di mana tingkat deteksi spam mendekati klasifikasi dengan fitur model bahasa bigram, yaitu hingga 95%, dan tingkat kesalahan yang rendah, yaitu sekitar 1 %.

ABSTRACT

The communication through social media and microblogging facillities is widely used today. Besides for personal communication, Twitter is also used to offer goods and services. This type of use has resulted in the development of spam on the twitter network. This research was conducted to develop a spam filter on Twitter by relying on text analysis on Indonesian tweets. The dataset is obtained from the Twitter API by searching for tweets using keywords obtained from the trending topic list for the Indonesian region.

Classification with the Naive Bayes method is chosen to detect spam tweets. The Naive Bayes method is used because it has been shown to produce good classification results in several related studies. The main features used are unigram / bag-of-words and bigram with stupid backoff smoothing technique. The classification results with this stupid backoff smoothing technique are compared with the Naive Bayes classification using with one of Unigram or Bigram as feature smoothed with Laplace Smoothing.

The evaluation was carried out using the k-fold cross validation method, with 10 cross tests. In testing compared to the classification with unigram, bigram, unigram and bigram features with calculation of term frequency and inverse document frequency, as well as classification with unigram and bigram features at once by using stupid backoff smoothing technique. From the test results it appears that the classification with the unigram feature has the lowest spam detection capability with a detection rate of 87.5%. However, this method classification has the lowest level of ham to spam classification error which is under 1%. Calculation of term and inverse document frequency can increase the detection rate by about 4% higher. The classification with the Bigram feature has the highest spam detection rate of up to 95%, but also has a high detection error rate of up to 15%. The best classification is the use of stupid backoff smoothing techniques, where the spam detection rate is close to classification with the bigram language model feature, which is up to 95%, and a low error rate, which is around 1%.

KATA PENGANTAR

Assalamu’alaikum wr. Wb

Segala puji syukur penulis panjatkan atas segala rahmat, nikmat dan ridhoNya, Sholawat dan salam penulis sampaikan kepada Nabi Muhammad SAW. Sehingga penulis dapat menyelesaikan penelitian ini dengan baik. Penelitian ini adalah sebagai salah satu syarat memperoleh gelar pasca sarjana pada sekolah tinggi teknik Surabaya dengan judul “Deteksi Spam Twit Pada Twitter Dengan Metode Naive Bayes”.

Ucapan terimakasih penulis sampaikan, atas segala bantuan dan dukungannya kepada :

1. Bapak Ir. Arya Tandy Hermawan, M.T., selaku Ketua Sekolah Tinggi Teknik Surabaya.
2. Bapak Ir. Gunawan, M.Kom., selaku Pembantu Ketua I Sekolah Tinggi Teknik Surabaya sekaligus sebagai dosen pembimbing akademik yang banyak memberikan arahan bagi penulis.
3. Ibu Ir. Hj. Endang Setyati, M.T., selaku Ketua Prodi Pascasarjana Sekolah Tinggi Teknik Surabaya.
4. Ibu Esther Irawati, S.Kom., M.Kom. selaku Dosen Pembimbing yang telah memberikan arahan dan masukkan yang sangat membantu penulis, terimakasih banyak atas waktu dan kesempatan bimbingan mulai dari awal hingga terselesaikannya penelitian ini.
5. Istri dan anak tercinta yang telah memberikan support, waktu dan doa, serta kerelaan atas hak waktu yang terkurangi demi penulis menyelesaikan penelitian ini.
6. Ibunda dan seluruh keluarga atas dukungannya.
7. Teman-teman seperjuangan, rombongan ASIA Malang yang senantiasa menyemangati penulis.
8. Bapak Muhammad Rofiq, atas semangatnya dan Yayasan WEC.
9. Serta teman-teman dosen ASIA yang telah banyak memberikan dukungan, doa dan masukkannya.

Semoga semua kebaikan yang telah diberikan kepada penulis penuh dengan berkah, dan semoga Allah SWT akan membalas dengan amal ibadah. Semoga penelitian ini dapat memberikan manfaat bagi seluruh pihak yang membutuhkan.

Wassalamu’alaikum, Wr.Wb

Surabaya, Oktober 2018

Penulis

DAFTAR ISI

Halaman

HALAMAN JUDUL i

HALAMAN PENGESAHAN ii

SURAT PERNYATAAN KEASLIAN iii

ABSTRAK iv

ABSTRACT v

KATA PENGANTAR vi

DAFTAR ISI vii

DAFTAR GAMBAR viii

DAFTAR TABEL ix

DAFTAR ALGORITMA x

DAFTAR SEGMEN PROGRAM xi

DAFTAR RUMUS xii

BAB I PENDAHULUAN 1

1.1 Latar Belakang 1

1.2 Tujuan 3

1.3 Hipotesis 4

1.4 Ruang Lingkup 4

1.4.1 Batasan Penelitian 4

1.4.2 Dataset 4

1.5 Metode penelitian 5

1.6 Sistematika Pembahasan 6

BAB II TEORI PENUNJANG 8

2.1 Tinjauan Pustaka 8

2.2 Penambangan Teks 17

2.3 Klasifikasi Teks 18

2.4 Tahapan Klasifikasi Teks 19

2.5 Model Bahasa N-grams 24

2.6 Metode Naive Bayes 25

2.7 Teknik Smoothing 26

BAB III METODOLOGI PENELITIAN 30

3.1 Tahapan Penelitian 30

3.2 Deskripsi Sistem 31

3.3 Pengumpulan Data 33

3.4 Praproses Data 36

3.5 Ekstrasi Fitur 38

3.6 Klasifikasi 39

3.7 Evaluasi 39

BAB IV PEMBENTUKAN DATASET 41

4.1 Pengumpulan dan Pelabelan Data 42

4.2 Pembersihan Data 44

4.3 CaseFolding 48

4.4 Konversi Kata 48

4.5 Filtering 50

4.6 Stemming 54

BAB V KLASIFIKASI 56

5.1 Ekstraksi Fitur 56

5.2 Training 60

5.3 Proses Klasifikasi 67

5.4 Smoothing 69

BAB VI UJICOBA 72

6.1 Cross-Validation 72

6.2 Ujicoba Klasifikasi Pertama 74

6.3 Ujicoba Klasifikasi Kedua 75

6.4 Ujicoba Klasifikasi Ketiga 77

6.5 Ujicoba Klasifikasi Keempat 78

6.6 Ujicoba Klasifikasi Kelima 79

6.7 Ujicoba Klasifikasi Keenam 81

6.8 Ujicoba Klasifikasi Ketujuh 82

6.9 Ujicoba Klasifikasi Kedelapan 84

6.10 Ujicoba Klasifikasi Kesembilan 85

6.11 Ujicoba Klasifikasi Kesepuluh 87

6.12 Analisa Hasil Pengujian 88

BAB VI PENUTUP 90

7.1 Kesimpulan 90

7.2 Saran 91

DAFTAR PUSTAKA 92

RIWAYAT HIDUP 94

LAMPIRAN A KUESIONER A-1

LAMPIRAN B TAMPILAN PROGRAM B-1

DAFTAR GAMBAR

Gambar Halaman

2.1 Arsitektur Sistem 9

2.2 Contoh Model bahasa 10

2.3 Hasil Pengujian Klasifikasi 10

2.4 Hasil 14

2.5 Proses Klasifikasi 19

3.1 Blok Sistem deteksi Spam 31

3.2 Diagram Alur Pencarian Twit 34

4.1 Pelabelan dan Praproses Data 41

4.2 Contoh Distribusi Kata Pada Twit Ham 51

4.3 Contoh Distribusi Kata Pada Spam 52

4.4 Contoh Frekuensi Kata Ham Setelah Filtering 53

4.5 Contoh Frekuensi Kata Spam Setelah Filtering 53

5.1 Distribusi Kata Pada Ham 57

5.2 Distribusi Kata Pada Spam 58

5.3 Distribusi Bigram Pada Twit Ham 59

5.4 Distribusi Bigram Pada Twit Spam 59

DAFTAR TABEL

Tabel Halaman

2.1 Contoh posting twitter berisi opini 11

2.2 Karakteristik dataset evaluasi 13

2.3 5 konsep teratas dan entitas terkait 16

2.4 Hasil klasifikasi 16

4.1 Contoh Twit Dengan Hashtag #PersijaDay 41

4.2 Contoh twit tagar #GanjarTaktakutPakDirman 41

4.3 Contoh tweet spam 42

4.4 Contoh hasil pembersihan data 44

4.5 Contoh hasil casefolding 46

4.6 Contoh kata yang disubstitusi 47

4.7 Contoh hasil konversi kata 48

4.8 Contoh Hasil Filtering 52

4.9 Contoh hasil stemming 52

5.1 Contoh frekuensi Unigram 12

5.2 Contoh Nilai Probailitas 12

5.3 Contoh probabilitas dengan tf-idf 12

6.1 Pembagian Data Training dan Data Testing 72

6.2 Kondisi Data Ujicoba Pertama 72

6.3 Hasil Ujicoba klasifikasi pertama 73

6.4 Data Training dan Testing Ujicoba Kedua 74

6.5 Hasil Ujicoba Klasifikasi Kedua 74

6.6 Data Training dan Testing Ujicoba Ketiga 75

6.7 Hasil Ujicoba Klasifikasi Ketiga 76

6.8 Data Training dan Testing Ujicoba Keempat 76

6.9 Hasil Ujicoba Klasifikasi Keempat 77

6.10 Data Training dan Testing Ujicoba Kelima 78

6.11 Hasil Ujicoba Klasifikasi Kelima 78

6.12 Data Training dan Testing Ujicoba Keenam 79

6.13 Hasil Ujicoba Klasifikasi Keenam 80

6.14 Data Training dan Testing Ujicoba Ketujuh 80

6.15 Hasil Ujicoba Klasifikasi Ketujuh 81

6.16 Data Training dan Testing Ujicoba Kedelapan 82

6.17 Hasil Ujicoba Klasifikasi Kedelapan Kedelapan 83

6.18 Data Training dan Testing Ujicoba Kesembilan 83

6.19 Hasil Ujicoba Klasifikasi Kesembilan 84

6.20 Data Training dan Testing Ujicoba Kesepuluh 85

6.21 Hasil Ujicoba Klasifikasi Kesepuluh 86

6.22 Hasil Akhir Pengujian 87

DAFTAR ALGORITMA

Algoritma Halaman

4.1 Data Cleaning 44

5.1 Penghitungan *bag-of-words* 55

5.2 Training Multinomial Naive Bayes 59

5.3 Klasifikasi Dengan Multinomial bayes 59

DAFTAR SEGMEN PROGRAM

Segmen Program Halaman

3.1 Otentikasi Twitter 32

4.1 Data Cleaning 44

4.2 Kode Proses Penggantian Kata 48

4.3 Fungsi Praproses Data 50

5.1 Penghitungan *Bag-of-words* 55

5.2 Menghitung Bigram 56

5.3 Menghitung *Term Frequency* 58

5.4 Menghitung *Inverse Document Frequency* 58

5.5 Penghitungan Probabilitas Pada Klasifikasi Naive Bayes 62

5.6 Penghitungan Probabilitas Klasifikasi Naive Bayes Dengan TF-IDF 64

5.7 Fungsi Klasifikasi Dengan Laplace Smoothing 65

5.8 Fungsi Klasifikasi Dengan Stupid Backoff 67

6.1 Evaluasi Sistem Klasifikasi Bayes 71

DAFTAR RUMUS

Gambar Halaman

2.1 Analisa Statistik Linguistik 12

2.2 Teori Bayes 13

2.3 Probabilitas Sentimen 13

2.4 Akurasi 13

2.5 *Decison* 14

2.6 Semantic Smoothing 15

2.7 Akurasi 22

2.8 Presisi 23

2.9 *Recall* 23

2.10 F1-score 23

2.11 Aturan rantai Probabilitas 24

2.12 Asumsi Markov 24

2.13 Hukum Komutativ Probabilitas 25

2.14 Probabilitas Independen 25

2.15 Teorema Umum Probabilitas 25

2.16 Teorema Umum Probabilitas 25

2.17 Teorema Umum Probabilitas 25

2.18 Teorema Umum Bayes 25

2.19 Klasifikasi Naive Bayes 25

2.21 Laplace Smoothing 27

2.22 Jelinek-Mercer Smoothing 27

2.23 Dirichlet Smoothing 27

2.24 Absolute Discounting Smoothing 27

2.25 Two-stage Smoothing 27

2.26 Katz’s backoff Smoothing 27

2.27 Stupid backoff Smoothing 27

2.25 Probabilitas Unigram 27

3.1 Term Frequency Inverse Document Frequency 38

3.2 Likelihood 39

5.1 Laplace Smoothing 60

5.2 Probabilitas dengan TF-IDF 62

5.3 Probabilitas dengan TF-IDF Dengan Laplace Smoothing 62

5.3 Inverse Document Frequency 62

# BAB I

PENDAHULUAN

1. Latar Belakang

Twitter merupakan salah satu dari media jaringan sosial yang penting saat ini. Dengan twitter orang bisa menyampaikan pemikirannya kepada publik dengan pesan yang tidak lebih dari 140 karakter. Mulai 2017, twitter meningkatkan batas pesan hingga 280 karakter[[1]](#footnote-1). Pesan yang singkat dan sederhana ini justru menjadi kekuatan twitter sehingga mencapai posisi 10 besar dalam sarana social networking dewasa ini. Sejak didirikan tahun 2006, pengguna twitter mencapai lebih dari 100 juta orang di dunia. Sejak 201, setiap hari lebih dari 500 juta twit dikirimkan[[2]](#footnote-2). Penggunaan twitter meliputi berbagai hal, ada yang hanya sekedar untuk mengekspresikan perasaan, sarana untuk membangun hubungan dengan penggemar bagi selebriti, sebagai sarana untuk menyebarkan informasi secara realtime, sebagai untuk menyampaikan opini terkait isu terkini, hingga sebagai sarana untuk menyampaikan pandangan bagi kalangan politisi. Untuk mengidentifikasi topik tertentu pengguna menggunakan tagar (#) pada kata kunci tertentu. Jika tagar yang Asama dikirimkan oleh banyak pengguna, maka tagar tersebut *menjadi trending topic*, yaitu topik yang sedang hangat dibicarakan. Salah satu perangkat yang populer yang ditawarkan oleh Twitter adalah daftar topik yang sedang naik daun. Dengan fitur ini pengguna Twitter bisa dengan cepat mendapatkan berita pada topik tertentu yang sedang trend.

Potensi pembaca Twitter yang besar ini menarik perhatian penyebar spam. Spam digunakan untuk berbagai tujuan, misalnya dengan kata-kata yang mengundang rasa ingin tahu, tentang informasi olahraga atau video yang menarik untuk mengarahkan pembaca untuk meng-klik daring yang berisi malware (Villeneuve, Nart, Koobface,2010)[[3]](#footnote-3) atau informasi produk tertentu. Sebagian lagi ada twit spam yang berisi penawaran produk tertentu yang disebarkan secara agresiv, baik oleh pengirim manusia, ataupun bot-nets. Daftar *trending topic* yang pada awalnya disediakan Twitter untuk memudahkan orang mengetahui topik yang sedang hangat akhirnya menjadi sasaran penyalahgunaan oleh penyebar spam. Penyebar spam memasukkan tagar yang menjadi *trending topic* pada twit spamnya sehingga tweetnya dibaca banyak orang. Hal ini tentu sangat mengganggu bagi orang yang membaca twit pada trending topik untuk mengetahui hal-hal yang berkenaan dengan topik itu saja, karena akan menghabiskan resources pada perangkat pengguna dan keseluruhan sistem dan tentunya juga waktu dari pembaca.

Twitter menyediakan beberapa cara bagi pengguna untuk melaporkan spam, dan digunakan oleh twitter untuk melakukan investi terhadap akun yang menyebarkannya. Jika terbukti sebagai penyebar spam maka akun akan diblokir. Akan tetapi cara ini kurang efektif karena sangat lambat, sedangkan trending topic sifatnya sangat dinamis dan hanya bertahan beberapa jam hingga hari saja (Martinez-Romo, & Araujo, 2013)[[4]](#footnote-4). Pendekatan tradisional untuk penanggulangan spam pada twitter sulit diimplementasikan, hal ini dikarenakan karakteristik twitter yang berbeda dengan aplikasi internet biasa seperti email atau website biasa. Pesan spam bisa berasal dari pengguna yang legitimate, sehingga blacklist terhadap asal pesan spam tidak mungkin dilaksanakan. Di samping itu pengguna twitter sudah terbiasa munggunakan layanan penyingkat URL untuk link yang ada di tweetnya. Hal tersebut menyulitkan untuk membedakan link spam yang berbahaya, atau link biasa. Ada beberapa usulan untuk mefilter spam pada jaringan twitter ini, di antaranya dengan menganalisa hubungan pengirim dan penerima (Song et al., 2011)[[5]](#footnote-5) . Yardi dkk. menganalisa struktur network akun untuk membedakan akun spammer dan non-spammer (Yardi, Romero, Schoenebeck, & Boyd, 2010)[[6]](#footnote-6). Demikian juga Benefuto dkk mendeteksi spammer dan non-spammer dengan melihat karakteristik twit dan perilaku akun pengirim (Benevenuto, Magno, Rodrigues, & Almeida, 2010)[[7]](#footnote-7). Pendekatan dengan menganalisa twit dilakukan oleh Martinez-Romo dan Araujo (Martinez-Romo & Araujo, 2013), yaitu dengan membandingkan isi twit dan isi tautan yang tertulis di dalam twit.

Pendeteksian spam dengan melakukan analisa terhadap akun pengirim lebih relevan terhadap akun yang sudah ada dalam waktu beberapa lama. Setelah akun diidentifikasi sebagai spam, langkah yang dilakukan adalah dengan memblokir akun tersebut, atau memblokir alamat IP-nya (Benevenuto et al., 2010). Akan tetapi penyebar spam bisa dengan mudah membuat akun baru untuk menyebarkan spamnya. Sedangkan analisa spam dengan membaca isi tautan juga mengalami kendala kemungkinan lambatnya koneksi. Selain itu, tidak semua twit spam berisi tautan untuk mengarahkan pembaa twit. Untuk itu pada penelitian ini dilakukan pendekatan dengan menganalisa isi twit untuk mendeteksi twit spam. Dengan pendekatan ini bisa dilakukan penyaringan twit spam secara real time.

1. Tujuan

Tujuan dari penelitian ini adalah:

* 1. Mendeteksi spam pada twit berbahasa Indonesia berdasar analisa terhadap isi pesan.
  2. Memfilter spam pada twit secara realtime
  3. Menghasilkan modul untuk filter spam yang efektif menggunakan klasifikasi Naive Bayes dalam bahasa pemrograman python.

1. Hipotesis

Dengan menggunakan metode kalasifikasi Naïve Bayes terhadap fitur yang diekstrak berdasar analisa statistik bahasa diharapkan bisa dilakukan pemilahan twit yang bukan spam dan yang termasuk spam. Diharapkan metode ini bisa mengklasifikasikan twit non-spam dan spam dengan benar hingga 85% dan kesalahan klasifikasi dari twit non-spam ke twit spam bisa ditekan hingga di bawah 5 %.

1. Ruang Lingkup

Pada bagian ruang lingkup akan dijelaskan mengenai batasan penelitian dan dataset yang akan digunakan dalam proses training dan testing untuk memberikan gambaran yang cukup menegenai pembahasan pada bab-bab berikutnya.

* + 1. Batasan Penelitian

Batasan pada penelitian tesis ini adalah sebagai berikut :

* 1. Klasifikasi hanya berdasarkan pada analisa teks twit dengan metode multinomial Naive Bayes
  2. Dataset diambil dari twitter berdasarkan pencarian dengan menggunakan kata kunci dari list trending topik pada akhir Mei hingga awal Juni 2018.
  3. Fitur diambil berdasarkan model bahasa unigram dan bigram.
  4. Smoothing yang digunakan adalah stupid backoff smoothing. Sedangkan untuk perbandingan digunakan laplace smoothing.
     1. Dataset

Dataset yang digunakan dalam penelitian ini adalah data pesan twitter mulai tanggal 19 Mei hingga 4 Juni 2018. Pengambilan data dilakukan dengan melakukan pencarian terhadap twit dari Twitter API berdasarkan kata kunci yang diambil dari daftar trending topic untuk wilayah Indonesia pada durasi waktu tersebut. Ada lima trending topic yang dipilih, yaitu #GanjarTakTakuPakDirman, #PersijaDay, #NasionalismeZamanNow, #TarawihInstagramable, #HarikebangkitanNasional.

1. Metode Penelitian

Metodologi yang digunakan dalam penelitian ini meliputi:

1. Studi Literatur

Dilakukan studi terhadap beberapa jurnal dan literatur pada topik terkait. Di antaranya yang menjadi referensi utama adalah “Detecting malicious tweets in twitter network using statstical analysis” oleh Martinez-Romo J, dan Araujo L., “Twitter as a Corpus for Sentiment Analysis and Opinion Mining” oleh S.Alexander Pak dan Patrick Paroubek[[8]](#footnote-8) menjadi rujukan terkait dengan penggunaan teks twit sebagai korpus, sedangkan referensi terkait smoothing pada klasifikasi naïve bayes di antaranya terdapat pada “Smoothing techniques for Adaptive Online Language Models: Topic Tracking in Twit Streams” oleh Lin J., Snow R., dan Morgan W[[9]](#footnote-9). Di samping itu dipelajari juga literatur terkait spam filtering dan Naive Bayes.

1. Pengumpulan Data

Pengumpulan data dilakukan dengan melakukan pencarian twit dengan menggunakan kata kunci yang didapatkan dari daftar topik yang sedang banyak dibicarakan pada wilayah Indonesia. Data kemudian dilabeli dalam spam dan ham secara manual.

1. Praproses Data

Data lalu dianalisa dan dilakukan praproses agar menjadi dataset yang siap untuk penelitian. Ada beberapa tahapan dalam praproses yaitu: pembersihan data, *casefolding*, *konversi*, *filtering*, dan *stemming*

1. Klasifikasi

Setelah praproses, lalu dilakukan klasifikasi. Tahapan di sini meliputi ekstraksi fitur, training atau pembelajaran, dan proses klasifikasi. Untuk klasifikasi digunakan metode Naive Bayes. Untuk mengatasi data sparsity digunakan teknik smoothing Laplace dan stupid backoff..

1. Pengujian

Pengujian dilakukan untuk melihat apakah data yang telah menjadi input akan diproses sesuai dengan output yang diharapkan. Hal ini juga dilakukan untuk mengevaluasi apakah metode yang diusulkan mampu menjawab tujuan yang telah usulkan.

1. Dokumentasi

Merupakan langkah akhir, penyusunan laporan mulai dari latar belakang permasalahan sampai dengan pengambilan kesimpulan akan dijelaskan dalam tahap dokumentasi ini.

1. Sistematika Pembahasan

Sistematika penulisan laporan pada tesis ini adalah sebagai berikut:

* BAB I : PENDAHULUAN

Dalam bab ini akan dijelaskan tentang latar belakang penelitian, tujuan dan manfaat yang diharapkan, hipotesis, ruang lingkup yang membatasi permasalahan yang dibahas, metode penelitian dan sistematika pembahasan yang dijelaskan secara detail dan terarah.

* BAB II : TINJAUAN PUSTAKA

Pada bab ini akan dijelaskan mengenai klasifikasi dalam data mining, metode klasifikasi, khususnya Naive Bayes, dan pengolahan bahasa natural

* BAB III : METODOLOGI PENELITIAN

Dalam bab ini akan menyajikan tentang metodologi yng digunakan dalam penelitian ini.

* BAB IV : PEMBENTUKAN DATASET

Dalam bab ini akan membahas tentang tahapan prapengolahan dari dataset.

* BAB V : KLASIFIKASI

Dalam bab ini akan dibahas tentang tahapan klasifikasi spam dan ham.

* BAB VI : PENGUJIAN DAN ANALISA HASIL PENELITIAN

Pada bab ini akan dibahas tentang pengujian dan analisa terhadap hasil yang didapat.

* BAB VII : PENUTUP

Dalam bab ini akan dijelaskan tentang kesimpulan dari tujuan penelitian didasarkan pada hasil uji coba, disertakan pula saran untuk perbaikan penelitian selanjutnya.

# BAB II

TEORI PENUNJANG

Pada bab ini akan dijelaskan tentang teori-teori dasar digunakan untuk menunjang penelitian ini. Penjelasan diawali dengan literatur penunjang, dilanjutkan dengan beberapa teori dasar penambangan teks. Setelah itu dibahas tentang langkah-langkah praproses terhadap data dan permasalahan umum tentang klasifikasi

1. Tinjauan Pustaka

Sejak twitter meraih peran yang signifikan dalam kehidupan masyarakat, banyak penelitian yang menjadikan pesan twitter obyek untuk diteliti. Beberapa literatur berikut merupakan seabgian contoh yang digunakan sebagai referensi dalam penelitian ini:

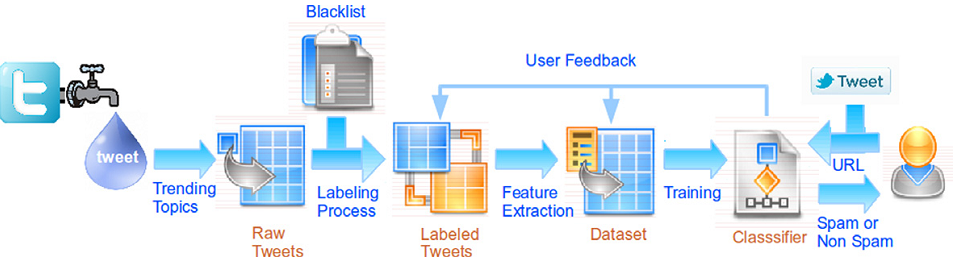
A. Detecting malicious Tweets in Twitter network Using Statistical Analysis

Penulis : Martinez-Romo, J., & Araujo, L.

Penerbit : Expert Systems with Applications

Tahun : 2013

Penelitian ini bertujuan untuk mendeteksi spam pada trending topik twitter. Berbeda dengan pendekatan yang dilakukan pada penelitian-penelitian sebelumnya yang melakukan pendekatan dengan identifikasi user dan berbasis honeypot. Metodologi yang dilakukan pada penelitian ini yaitu pendeteksian spam secara terisolir, tanpa sebelumnya mengetahui informasi mengenai akun pengirim dan penggunaan analisa statistik bahasa untuk mendeteksi spam dalam trending topik. Pendekatan yag dilakukan adalah dengan memeriksa bahasa pada topik, twit, dan halaman web yang terhubung pada twit. Pada twit spam model bahasanya cenderung berbeda sangat signifikan: spammer biasanya berusaha mengarahkan ke trafik ke halaman web yang dia inginkan yang tidak punya keterkaitan secara semantik dengan topik. Perbedaan model bahasa tadi digunakan untuk membedakan twit yang spam dan non-spam.



Gambar 2.1

Arsitektur Sistem

Sistem terbagi dalam 5 proses:

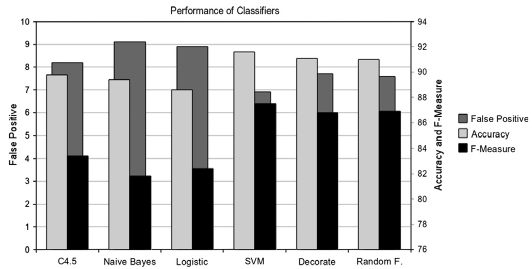
1. Pengumpulan trending topik, pertama kali sistem mengambil sejumlah twit yang terkait dengan sebuah trend tertentu. Pengembalian twit dilakukan secara periodik untuk mendapatkan sekelompok twit yang heterogen
2. Pelabelan spam. . Langkah kedua dengan melakukan pelabelan spam terhadap trending topik, di mana sistem menggunakan beberapa blaklist yang tersedia untuk mendeteksi spam URL pada twit. Hasilnya digunakan untuk melatih sistem dan untuk mendeteksi twit spam baru.
3. Ekstraksi fitur. Ekstraksi fitur dilakukan untuk merepresentasikan twit yang dilabeli menggunakan pengolahan bahasa natural dan teknik analisa isi terhadap isi twit dan isi halaman target yang terdapat pada tautan yang tertera.
4. Pembelajaran komponen sistem klasifikasi. Dataset akhir, berupa twit yang dilabeli dan setiap twit direpresentasikan dengan sejumlah fitur, digunakan untuk melatih komponen klasifikasi terhadap model bahasanya dan mendapatkan pengetahuan untuk mendeteksi spam
5. Pendeteksian spam. Algoritma klasifikasi mendapatkan twit dari user sebagai input dan memberikan notifikasi kepada user apakah spam atau tidak. Pengguna sistem dapat memberikan umpan balik koreksi jika sistem dianggap salah melakukan klasifikasi terhadap twit. Dengan demikian komponen klasifikasi terus melakukan proses pembelajaran.



Gambar 2.2

Contoh model bahasa

Bebeapa algorithma klasifikasi diuji dalam penelitian ini, meliputi C4.5, Naive Bayes, support vector machine, decorate, dan random forest.



Gambar 2.3

Hasil pengujian klasifikasi

Sistem yang dikembangkan pada penelitian ini bisa melakukan klasifikasi secara benar pada twit non-spam dan twit spam dengan tingkat akurasi mencapai masing-masing 89,3% dan 93,7%, dan hanya 6,3 % twit non spam yang terklasifikasikan secara salah ke dalam twit spam.

B. Twitter as a Corpus for Sentiment Analysis and Opinion Mining

Penulis : S Alexander Pak, Patrick Paroubek

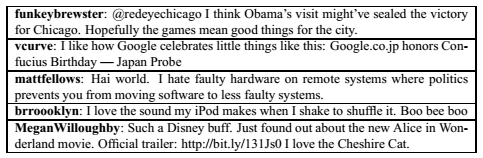
Institusi : Universit´e de Paris-Sud, Laboratoire LIMSI-CNRS, Bˆatiment 508,

Pada paper ini, peneliti memfokuskan pada twitter, platform microblogging yang paling populer saat ini, untuk melakukan analisa sentimen. Di sini dilakukan dilakukan pengumpulan twit secara otomatis dan digunakan sebagai korpus untuk analisa sentimen dan penambangan opini. Peneliti melakukan analisa bahasa terhadap korpus yang telah dikumpulkan dan dilakukan analisa terhadap fenomena yang didapatkan. Dengan menggunakan korpus tersebut, peneliti membangun sistem klasifikasi sentimen, yang bisa menentukan sentimen positif, negatif, dan netral dari suatu dokumen. Hasil pengujian menunjukkan bahwa teknik yang diajukan bisa bekerja dengan efisien dan menunjukkan hasil yang lebih baik daripada metode yang telah ada sebelumnya. Penelitian ini dilakukan pada obyek berbahasa inggris. Korpus dibangun dari 300.000 pesan twitter, yang dikelompokkan secara otomatis ke dalam 3 jenis:

1. Teks yang berisi emosi positif, misalnya perasaan senang, menikmati, dan bahagia.
2. Teks yang berisi emosi negatif, di antaranya adalah kesedihan, marah, dan kekecewaan.
3. Teks obyektiv atau netral yang hanya menyampaikan fakta tanpa mengekspresikan suatu emosi.

Tabel 2.1

Contoh posting twitter berisi opini



Hal yang dilakukan dalam penelitian ini meliputi:

1. Mengumpulkan korpus untuk sentimen positif, negatif, dan netral. Prosedur yang dilakukan adalah sama sebagaimana digunakan oleh (Read, 2005; Go et al., 2009)[[10]](#footnote-10) yaitu memilah twitter berdassar emoticon:

a. Happy emoticons: “:-)”, “:)”, “=)”, “:D”, dll

b. Sad emoticons: “:-(“, “:(“, “=(“, “;(“, dll

1. Melakukan analisa statistik linguistik terhadap corpus yang dikumpulkan

(2.1)

Di mana dan adalah jumlah kemunculan Tag T pada set 1 dan 2.

1. Corpora yang dikumpulkan digunakan untuk membangun sistem klasifikasi sentimen pada microblogging. Dataset yang dikumpulkan digunakan untuk ekstraksi fitur yang digunakan untuk pembelajaran sistem klasifikasi sentimen. Fitur yang digunakan untuk klasifikasi adalah model bahasa n-gram sebagai fitur biner. Peneliti melakukan eksperimen terhadap unigram, bigram, dan trigram. Langkah-langkah yang dilakukan mendapatkan n-gram meliputi
2. Filtering, untuk menghilangkan URL, nama user twitter ( kata yang diawali @), spesial word dari twitter (misal “RT”), dan emoticons.
3. Tokenization, membagi teks dengan memisahkan berdasar spasi dan tanda titik dan membentuk bag-of-words. Kata-kata bahass Inggris yang dipendekkan seperti “don’t”, “I’ll”, “she’d” tetap dijadikan sebagai kata tunggal.
4. Menghilangkan stopwords seperti “a”, “an”, “the” dari bag-of-words yang dibentuk
5. Mengkonstruksi n-gram. Sejumlah pasangan urutan kata dibentuk. Pernyataan negasi seperti “no” atau “not” digabungkan dengan kata yang mendahului atau mengikuti. Hal ini bisa meningkatkan akurasi karena pernyataan negasi sangat penting dalam pernyataan opini.
6. Pengujian untuk membuktikan efektifitas dari sistem yang dikembangkan.

Sistem klasifikasi sentimen yang dibangun menggunakan metode multinomial Naive Bayes. Disamping itu dilakukan eksperimen dengan sistem klasifikasi SVM (Alpaydin, 2004)[[11]](#footnote-11) dan CRF (Lafferty et al., 2001)[[12]](#footnote-12), tetapi hasil terbaik didapatkan dengan Naive Bayes.

Naive Bayes berdasar pada Teori Bayes (Anthony J, 2007)[[13]](#footnote-13)

(2.2)

Di mana s menunjukkan sentimen, dan M adalah pesan Twitter. Karena terdapat jumlah yang sama untuk dataset positif, negatif, dan netral, rumus disederhanakan menjadi

(2.3)

Dataset yang digunakan untuk evaluasi ada pada tabel berikut:

Tabel 2.2

Karakteristik dataset evaluasi

|  |  |
| --- | --- |
| Sentiment | Jumlah sampel |
| Positif | 108 |
| Negatif | 75 |
| Netral | 33 |
| Total108 | 216 |

Untuk menghitung akurasi digunakan rumus berikut:

(2.4)

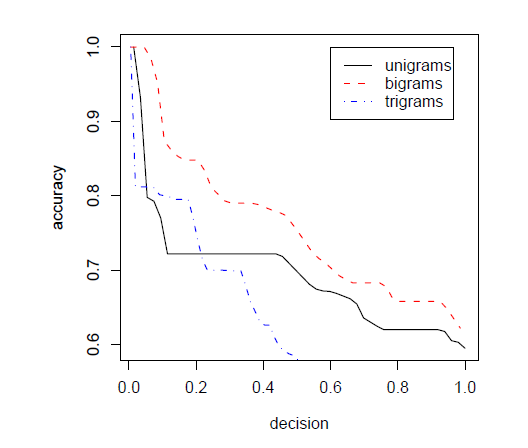
Sedangkan untuk pengukuran decision dari sistem klasifikasi:

(2.5)

Keterangan :

* Dokumen diterima adalah dokumen diterima yang cocok dengan topik
* Semua klasifikasi adalah jumlah secara keseluruhan data klasifikasi

Hasil pengukuran akurasi dan pengambilan keputusan dari penelitian ini disajikan seperti gambar dibawah ini.



Gambar 2.4

Hasil

Dari Gambar 2.4, tampak bahwa bigram menghasilkan akurasi terbaik.

C. Semantic Smoothing for Twitter Sentiment[[14]](#footnote-14)

Penulis : Hassan Saif, Yulan He and Harith Alani

Jurnal : 10th International Semantic Web Conference (ISWC).

Tahun : 2011

Penelitian ini ditujukan untuk mengatasi problem data sparsity yang biasa terjadi dalam klasifikasi teks menggunakan Naive Bayes khususnya pada twitter. Dalam pengolahan bahasa natural, data sparsity adalah istilah yang digunakan untuk menggambarkan fenomena tidak cukupnya observasi data di dalam korpus untuk membuat pemodelan bahasa secara akurat[[15]](#footnote-15). Ide utamanya adalah melakukan ekstraksi konsep yang secara semantik tersembunyi di dalam data lalu menggabungkan konsep-konsep ini sebagai fitur tambahan untuk pelatihan sistem klasifikasi.

Hassan Saif, dkk melakukan investigasi terhadap dua cara yang berbeda untuk merealisasikan semantic smoothing. Cara pertama disebut dengan shallow semantic smoothing, yaitu proses smoothing dengan cara mengganti kata dalam data pembelajaran dengan konsep semantik yang sesuai dengannya, misalnya pada twit “downloading apps for my iPhone!So much fun”, entitas “iPhone” diganti dengan konsep “product”. Entitas lain yan gsemacam dengan iPhone juga diganti dengan konsep “produk”. Cara kedua disebut metode interpolasi, di mana dilakukan interpolasi terhadap model bahasa unigram yang asli dalam sistem klasifikasi Naive Bayes dengan model yang dibuat berdasar kata-kata yang terdapat dalam konsep semantik. Perhitungan klasifikasi Naive Bayes dengan semantic smoothing adalah sebagai berikut:

(2.6)

Di mana adalah model kelas unigram dengan semantic smoothing, adalah kelas model unigram dengan perkiraan likelihood maximum, adalah kata konsep w urutan ke j, adalah distribusi dari konsep semantic dalam data training untuk kelas yang ada dan itu bisa dihitung melalui estimasi likelihood maximum. adalah distribusi kata-kata di dalam data training jika untuk suatu konsep, bisa dihitung dengan algoritma expectation-maximization (EM algorithm). Terakhir, koefisien α digunakan untuk mengontrol pengaruh dari pemasangan semantik dalam model kelas yang baru.

Tabel 2.3

5 konsep teratas dan entitas terkait

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Konsep | Person | Company | City | Country | Organisation |
| Jumlah entitas | 4954 | 2815 | 1575 | 961 | 614 |

Eksperimen dilakukan terhadap data yang terdiri dari 1.6 juta twit. Sejumlah eksperimen dilakukan dengan menggunakan shallow semantic dan interpolasi terhadap Naive Bayes. Untuk metode interpolasi, α ditetapkan dengan nilai konstan 0.8. Dari observasi terhadap hasil pengujian di Tabel 2.4 bisa dilihat bahwa Naive Bayes yang dilatih hanya dengan fitur unigram menghasilkan akurasi 81%. Penggunaan shallow smoothing justru menurunkan akurasi hingga mendekati 5% dikarenakan informasi yang hilang dikarenakan penggunaan penggantian konsep semantik. Penggunaan interpolasi bisa kembali menaikkan akurasinya.

Tabel 2.4

Hasil klasifikasi

|  |  |
| --- | --- |
| Metode | Akurasi |
| Tanpa semantic smoothing | 81% |
| Shallow semantic smoothing | 76.3% |
| Metode interpolasi | 81.3% |

1. Penambangan Teks

Penambangan teks adalah proses untuk mendapatkan pengetahuan melalui koleksi dokumen menggunakan sejumlah perangkat untuk analisa. Sebagaimana penambangan data, penambangan teks mengekplorasi dan menganalisis data dalam jumlah besar baik secara otomatis maupun semi otomatis untuk mendapatkan suatu pola yang bermakna dari data, khusus untuk penambangan teks, sumber data adalah kumpulan dokumen, dan pola yang didapat bukan terdapat dalam record formal pada basis data, akan tetapi pada data tekstual yang tidak terstruktur dalam kumpulan dokumen yang dianalisa.

Sebagaimana penambangan data, penambangan teks memiliki beberapa fungsionalitas yang sering digunakan, yaitu:

* 1. Karakterisasi dan diskriminasi: yaitu menggeneralisasi, merangkum, dan mengkontraskan karakteristik data.
  2. Penggalian pola berulang: yaitu pencarian pola asosiasi (association rule) atau pola intra-transaksi, atau pola pembelian yang terjadi dalam satu kali transaksi.
  3. Klasifikasi: yaitu membangun suatu model yang bisa mengklasifikasikan suatu obyek berdasar atribut-atributnya. Kelas target sudah tersedia dalam data sebelumnya, sehingga fokusnya adalah bagaimana mempelajari data yang ada agar klasifier bisa mengklasifikasikan sendiri.
  4. Prediksi: yaitu memprediksi nila yang tidak diketahui atau nilai yang hilang menggunakan model dari klasifikasi.
  5. Penggugusan/analisa klaster: yaitu mengelompokkan sekumpulan objek data berdasarkan kemiripannya. Kelas target tidak tersedia dalam data sebelumnya, sehingga fokusnya adalah memaksimalkan kemiripan intra kelas dan meminimalkan kemiripan antar kelas.
  6. Analisis outlier: yaitu proses pengenalan data yang tidak sesuai dengan perilaku umum dari data lainnya. Contoh: mengenali noise atau spam dan pengecualian dalam data.
  7. Analisa trend dan evousi: meliputi analisis regresi, penggalian pola sekuensial, analisis periodisitas, dan analisis berbasis kemiripan.

Pada data mining, data diasumsikan sudah tersimpan dalam format yang terstruktur, sehingga praproses berfokus pada dua hal pokok, yaitu pembersihan dan normalisasi data dan sejumlah besar tindakan praproses, titik berat praproses terletak pada identifikasi dan ekstraksi fitur yang representatif dari dokumen dengan bahasa natural. Praproses ini bertanggung jawab mengubah data tidak terstruktur yang tersimpan dalam kumpulan dokumen ke format antara yang secara terstruktur untuk diproses lebih lanjut.

Lebih lanjut, karena informasi yang ditambang berasal dari bahasa natural, maka penambangan teks banyak menggunakan teknik pada pemrosesan bahasa natural. Di antara teknik dan metodologi yang digunakan yaitu teknik dan metodologi pada area pengambilan informasi, ekstraksi informasi, dan perhitungan bahasa berbasis korpus, misalnya perhitungan frekuensi kata, mengembangkan maximum likelihood-estimation (MLE) untuk teks, menerapkan teknik smoothing pada MLE, interpolasi data, dan lain-lain.

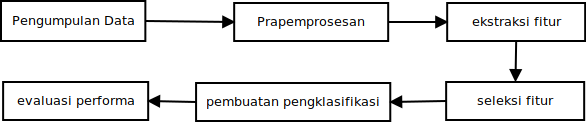
1. Klasifikasi Teks

Klasifikasi teks adalah proses untuk mengklasifikasikan dokumen berdasarkan kata-kata yang terkandung di dalam teks tersebut. Klasifikasi merupakan pencarian sekumpulan model (fungsi) yang menggambarkan dan membedakan kelas atau konsep data dengan maksud menggunakan model tersebut sebagai prediksi terhadap kelas dimana label kelas tersebut tidak diketahui.

Aplikasi dari klasifikasi teks di antaranya adalah pemfilteran spam pada email, di mana email-email diklasifikasikan dalam ham atau email yang relevan dan berguna bagi penggunanya dan spam atau email yang tidak relevan. Teks spam bisa dalam bentuk promosi untuk produk tertentu yang tidak dikehendaki oleh pengguna atau yang lebih berbahaya adalah adanya tautan yang mengarahkan pada halaman web yang berisi malware atau potongan kode program yangdigunakan untuk membajak suatu akun. Aplikasi yang lain adalah mengkategorisasikan artikel berita pada topik tertentu, misalnya berita, olahraga, atau politik. Penggunaan klasifikasi teks berikutnya adalah identifikasi trend topik pada aliran berita, lalu identifikasi bahasa pada suatu artikel. Contoh lain yang bisa dilakukan sebagai pemanfaatan klasifikasi teks adalah analisa sentimen misalnya pada twitter atau ulasan konsumen pada toko online. Teknik klasifikasi teks juga bisa digunakan untuk *authorship attribution*, yaitu menentukan penulis dari suatu teks. Beberapa metode klasifikasi pada data mining yang biasa digunakan adalah decision tree, bayesian, jaringan syaraf tiruan backpropagation (propagasi balik), k-nearest neighbor, Rule-based (berbasis aturan), Support Vector Machine (SVM) dan lain-lain.

1. Tahapan Klasifikasi Teks

Secara umum, proses klasifikasi dibagi dalam enam tahapan, yaitu: pengumpulan data, fase penyiapan data atau praproses, ekstraksi fitur, pemilihan fitur, pembuatan sistem klasifikasi, dan evaluasi performa[[16]](#footnote-16).



Gambar 2.5

Proses klasifikasi

1. Pengumpulan Data

Tahapan awal dalam proses klasifikasi teks adalah pengumpulan data. Data didapatkan dari sumber yang hendak diklasifikasikan, bisa dari teks email, dokumen web, twitter, grup-grup berita dan lain-lain. Dokumen ini adalah sumber untuk membentuk korpus. Untuk klasifikasi, data yang terkumpul kemudian dilabeli sesuai dengan kelas klasifikasi tujuan. Dokumen-dokumen ini akan digunakan dalam proses training dan testing dari sistem klasifikasi

1. Praproses

Praproses adalah tahap penyiapan data agar bisa digunakan dalam training dan testing dari sistem klasifikasi. Langkah pertama dalam praproses adalah merepresentasikan teks dokument dalam himpunan kata yang siap untuk ekstraksi fitur untuk digunakan dalam pembelajaran sistem klasifikasi nanti. Tahapan praproses meliputi:

1. Pembersihan data, yaitu membersihkan data dari simbol-simbol non-ASCII, HTML-escaping, dan lain-lain
2. Tokenisasi: Proses untuk membagi korpus teks ke dalam elemen individual yang akan digunakan untuk menjadi input bagi algoritma pemprosesan bahasa natural. Dokumen diperlakukan sebagai string, kemudian dibagi dalam sejumlah token, yaitu kata-kata penyusunnya.
3. Penghilangan stopword: Stopword adalah kata-kata yang tergolong terlalu umum digunakan, sehingga bisa dikatakan tidak signifikan untuk menjadi fitur klasifikasi. Contoh yang bisa dimasukkan dalam stopword ini di antaranya adalah kata sambung, misalnya dalam bahasa indonesia yaitu “yang”, “dari”, “ke”, dan lain-lain.
4. Stemming: pada tahapan ini setiap kata dikembalikan ke bentuk dasarnya. Dengan demikian maka kata yang sama tidak terhitung secara berbeda, misalnya kata “membaca” akan dikembalikan ke bentuk asal “baca”.
5. Ekstraksi Fitur

Ekstraksi fitur pada teks adalah proses transformasi dari kumpulan kata-kata ke dalam bentuk set fitur yang berguna bagi sistem klasifikasi[[17]](#footnote-17). Untuk mengurangi kompleksitas dari dokumen, dokumen ditransformasikan dari bentuk teks ke dalam bentuk vektor nilai yang merepresentasikan fitur tertentu. Representasi dokumen yang paling umum digunakan adalah vector space model (VSDM), yang mana dokumen direpresentasikan dalam vektor dari kata-kata.

1. Seleksi Fitur

Setelah praproses dan ekstraksi fitur, tahapan penting selanjutnya adalah seleksi fitur untuk membangun vector space, yang akan meningkatkan skalibilitas, efisiensi, dan akurasi dari pengklsifikasi teks. Ide utama dalam tahapan ini adalah memilih sejumlah fitur dari dokumen asal. Seleksi fitur dilakukan dengan memilih kata-kata dengan skor tertinggi menurut pengukuran tingkat kepentingan dari suatu kata yang telah ditentukan. Beberapa teknik evaluasi fitur yang biasa digunakan di antaranya adalah information gain (IG), term frequency, Chi-square, expected cross entropy, odds ratio, the weight evidence of text, mutual information, dan gini index[[18]](#footnote-18).

1. Klasifikasi

Klasifikasi dimulai dengan tahapan training, yaitu pembelajaran untuk mendapatkan pengetahuan dari data training berdasarkan fitur yang telah diekstrak Klasifikasi dokumen bisa dilakukan dalam tiga metode, yaitu: supervised, unsupervised, dan semi-supervised. Supervised learning adalah pembelajaran dari data training yang telah dilabeli ke dalam kelas tujuan yang diketahui. Sedangkan unsupervised learning adalah pembelajaran pada data set yang tidak terlabeli. Biasanya untuk unsupervised learning digunakan istilah pengklasteran. Sedangkan semi-supervised adalah pembelajaran di mana sebagaian data telah dilabeli. Hasil dari pembelajaran adalah model sistem klasifikasi. Model ini merupakan representasi pengetahuan yang akan digunakan untuk prediksi kelas data baru yang belum pernah ada. Beberapa algoritma yang bisa digunakan di antaranya adalah klasifikasi bayes, decision tree, k-nearest neighbour (KNN), support vector machines (SVM), dan neural network

1. Evaluasi Performa

Untuk mengevaluasi sistem klasifikasi, data dibagi menjadi 2 (dua) yaitu training set dan testing set. Training set digunakan oleh algoritma klasifikasi untuk membentuk sebuah model sistem klasifikasi. Testing set digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar. Karena itu, data yang ada pada testing set seharusnya tidak boleh ada pada training set sehingga dapat diketahui apakah model classifier sudah “pintar” dalam melakukan klasifikasi.

Lain lagi halnya dengan validation set. Umumnya beberapa algoritma klasifikasi memerlukan beberapa parameter. Misalnya: jumlah hidden layer dan learning rate pada neural network; parameter kernel pada SVM. Biasanya sebagian dari training set diambil untuk validation set. Validation set ini digunakan untuk mencari parameter yang paling baik untuk sebuah algoritma klasifikasi.

Memisahkan data menjadi training dan testing set dimaksudkan agar model yang diperoleh nantinya memiliki kemampuan generalisasi yang baik dalam melakukan klasifikasi data. Tidak jarang sebuah model klasifikasi dapat melakukan klasifikasi data dengan sangat baik pada training set, tetapi sangat buruk dalam melakukan klasifikasi data yang baru dan belum pernah ada. Hal ini dinamakan overfitting

Performa dari sistem klasifikasi teks bisa dievaluasi dengan menggunakan empat ukuran, yaitu: akurasi, presisi, recall, dan pengukuran F1. Alat pengukuran yang biasa digunakan adalah confusion matrix[[19]](#footnote-19). Hasil klasifikasi dikelompokkan dalam istilah true positif dan true negatif untuk pengelompokan klasifikasi yang benar, dan false positif dan false negatif untuk klasifikasi yang salah.

Tabel 2.5

Confusion matrix

|  |  |  |  |
| --- | --- | --- | --- |
|  | Predicted class | | |
| Actual class |  | Class 1 | Class 2 |
| Class 1 | True positife (tp) | False negative (fn) |
| Class 2 | False positife (fp) | True negative (tn) |

Akurasi digunakan untuk mengukur klasifikasi yang benar dibandingkan dengan keseluruhan hasil klasifikasi. Ukuran ini merupakan ukuran yang paling intuitiv untuk menunjukan keakuratan dari sistem klasifikasi.

(2.7)

Akurasi akan gagal menggambarkan performa sistem klasifikasi jika jumlah false positif dan false negatif tidak berimbang. Untuk lebih memberikan gambaran performa diperlukan ukuran yang lain.

Presisi mengukur rasio klasifikasi yang benar terhadap keseluruhan kelas klasifikasi yang terpilih. Presisi yang tinggi berarti lebih sedikit terjadi kesalahan positif, sedangkan rendahnya presisi berarti lebih banyak terjadi kesalahan positif. Sebagai contoh dalam spam filtering, presisi yang tinggi berarti sedikit teks bukan spam yang tergolong sebagai spam. Sebaliknya jika filter banyak menggolongkan teks bukan spam sebagai spam berarti presisinya rendah.

(2.8)

Recall menunjukkan rasio kelas terpilih yang benar terhadap keseluruhan kelas yang seharusnya terpilih. Misalnya untuk spam filter, dari seluruh spam yang ada, seberapa banyak sistem klasifikasi bisa melabeli spam secara tepat.

(2.9)

Dari *weighted average* antara presisi dan *recall* dibuat satuan ukuran baru yaitu f1 score. Untuk hasil klasifikasi dengan jumlah false positif dan false negatif yang tidak berimbang, lebih baik digunakan f1 score, karena mempertimbangkan baik false positif maupun false negatif.

(2.10)

1. Model Bahasa N-grams

Language Model (LM) atau model bahasa adalah suatu model untuk menentukan probabilitas dari urutan kata-kata. Model bahasa yang paling sederhana adalah n-gram. N-gram merupakan urutan dari N kata: sebuah 2-gram atau bigram adalah dua kata yang berurutan, seperti kalimat “Budi sedang menulis” akan menjadi dua buah bigram: “Budi sedang”, dan “sedang menulis”. Demikian selanjutnya, maka 3-gram atau trigram adalah urutan rangkaian 3 kata, misalnya pada kalimat di atas hanya terdapat satu trigram: “Budi sedang menulis”. 1-gram atau unigram adalah bentuk khusus dari n-gram, di mana hanya menunjukkan kumpulan kata-kata, sehingga sering disebut dengan Bag-of-words (BOW).

Untuk menghitung probabilitas dari keseluruhan urutan kata dalam kalimat digunakan aturan rantai probabilitas:

(2.11)

menunjukkan probablitas urutan N kata dari w1, w2 ....wn. Sedangkan merupakan probablitas dari kata jika didepannya adalah urutan w1, w2 ....wk-1. Untuk menyederhanakan penghitungan digunakan asumsi Markov, di mana probabilitas kemunculan dari suatu kata hanya tergantung dari satu kata di depannya.

(2.12)

1. Metode Naive Bayes

Metode Naive Bayes bersumber dari teori Bayes yang digunakan untuk memudahkan penghitungan probabilitas. Yang mana probabilitas dinyatakan sebagai nilai antar 0 hingga 1 yang menunjukkan tingkat keyakinan terhadap suatu fakta atau prediksi, yang mana nilai satu menunjukkan keyakinan 100 % bahwa suatu fakta memang benar terjadi, dan nol menunjukkan keyakinan 0 %.

Probabilitas dari A dinyatakan dalam P(A). Sedangkan untuk menunjukkan probilitas dari A dalam kondisi B benar dituliskan dengan P(A|B). Untuk menunjukkan probabilitas A da B bernilai benar, dituliskan P(A dan B).

Teori Bayes didasarkan pada hukum komutatif:

(2.13)

Pada kondisi A dan B independen, bisa dituliskan hubungan berikut:

(2.14)

Sedangkan untuk kondisi yang lebih umum, di mana kondisi A dan B saling terkait, maka

(2.15)

Yang berarti, probabilitas A dan B bernilai benar sama dengan nilai probalitas A kali nilai probabilitas B untuk A bernilai benar.

Jika sebaliknya dituliskan |B) maka didapatkan

(2.16)

Sehingga bisa didapatkan rumus untuk mencari probabilitas B untuk kondisi A sebagai berikut:

(2.17)

Atau

(2.18)

Teori Bayes ini diterapkan untuk klasifikasi yang biasa disebut Naive Bayes. untuk menghitung probabilitas suatu kelas C dengan fitur yang diketahui, misalnya F1 dan F2 sebagai berikut:

(2.19)

Dengan ketentuan:

* P(C) adalah *prior probability* dari kelas C tanpa mengetahui data. Nilai ini bisa didapatkan dengan menghitung data yang termasuk kelas C dari keseluruhan data training
* P( adalah *evidence* atau nilai probabilitas dari fitur F1 dan F2.
* Sedangkan P( adalah nilai yang menunjukkan seberapa mungkin didapatkan fitur F1 dan F2 jika diketahui data termasuk dalam kelas C.

1. Teknik *Smoothing*

Smoothing pada klasifikasi Naive Bayes dilakukan untuk penyesuaian terhadap *Maximum Likelihood-Estimation* (MLE) dari model bahasa sehingga akan lebih akurat. Permasalahan yang sering dihadapi oleh klasifikasi Naive Bayes adalah *data sparsity*, di mana observasi data tidak cukup sehingga informasi yang didapatkan dari korpus kurang akurat. Smoothing berperan untuk meningkatkan akurasi model bahasa serta mengakomodasi pembangkitan kata-kata yang umum dan tidak informatif.

Pembangkit maximum likelihood pada umumnya mengabaikan probabilitas kata-kata yang tidak muncul. Oleh karena itu tujuan utama dari smoothing adalah menyediakan probabilitas di atas nol bagi kata-kata yang tidak terdapat dalam dataset sehingga bisa meningkatkan akurasi dari estimator probabilitas[[20]](#footnote-20). Bentuk umum dari model yang mengalami smoothing adalah:

(2.20)

Di mana Ps(w|d) adalah probabiltas dari kata yang terdapat di dalam dokumen, dan P(w|d) adalah kumpulan dari model bahasa dan αd adalah koefisien dari probabiltas yang diberikan kepada kata-kata yang belum ada sedemikian sehingga jumlahnya 1.

Secara umum metode smoothing dibedakan pada pilihan untuk menentukan Ps(w|d). Suatu metode smoothing bisa sesederhana hanya menambahkan suatu nilai saja, atau bisa lebih kompleks di mana kata-kata dengan jumlah berbeda dipeerlakukan secara berbeda. Untuk c(w,ci) sebagai menunjukkan frekuensi dari kata w dalam kategori ci, dan p(w|C) merupakan estimasi maksmum likelihood dari kata w dalam C, beberapa metode smoothing berikut bisa digunakan:

1. Laplace smoothing

(2.21)

1. Jelinek-Mercer smoothing

(2.22)

1. Dirichlet smoothing

(2.23)

1. Absolute discounting smoothing

(2.24)

1. Two-stage smoothing

(2.25)

1. Katz’s backoff smoothing

(2.26)

Dengan d adalah estimasi good-touring.

Pada penelitian ini digunakan stupid backoff smoothing yang merupakan penyederhanaan dari katz’s backoff, di mana tidak menggunakan nilai discounting good-touring, dan langsung menggunakan frekuensi relativ.

(2.27)

Secara umum faktor backoff bisa tergantung k, pada penelitian ini menggunakan nilai konstan 0,4 sebagaimana pada yang dilakukan Brants dkk[[21]](#footnote-21). Untuk unigram model rumus ni setelah penghitungan rekursiv menjadi

(2.28)

Di sini N adalah ukuran korpus training, dan f(wi) adalah frekuensi dari kemunculan kata wi.

# BAB III

METODOLOGI PENELITIAN

Bab ini membahas tentang tahapan penelitian dan metodologi yang digunakan mulai dari studi literatur, pengumpulan data, analisis permasalahan, pembentukan dataset, pembelajaran sistem klasifikasi, implementasi, hingga penyusunan laporan. Selain itu dibahas juga tentang deskripsi sistem yang dikembangkan

1. Tahapan Penelitian

Penelitian yang dilakukan melalui beberapa tahapan sebagai berikut:

1. Studi Literatur

Dilakukan studi terhadap beberapa literatur berupa buku dan jurnal penelitian yang terkait dengan penelitian yang dilakukan. Topik bahasan yang mendukung penelitian ini adalah yang mengenai Naive Bayes classifier, smoothing, klasifikasi multilabel, pemprosesan bahasa natural.

1. Pengumpulan Data

Data diambil dengan cara melakukan pencarian pada twitter dengan menggunakan keyword berdasar trending topik di Indonesia. Di samping itu dilakukan juga pencarian dengan keyword pada topik yang menghasilkan banayak twit dan diperkirakan dimanfaatkan pula oleh penyebar spam.

1. Pembentukan Dataset

Dari data yang terkumpul, kemudian dilakukan pelabelan secara manual terhadap twit-twit yang diperoleh untuk menggolongkan twit ke dalam kelas spam dan ham. Setelah proses pelabelan, lalu dilakukan praproses terhadap data, yaitu menyiapkan data agar bisa diekstrak fiturnya dengan maksimal.

1. Ekstraksi dan Seleksi Fitur

Ekstraksi fitur adalah pengambilan fitur yang menjadi petunjuk klasifikasi dari dokumen. Dalam ekstraksi fitu ini dokumen dalam dataset dijadikan vektor fitur yang akan digunakan dalam proses pembelajaran sistem klasifikasi.

1. Proses klasifikasi

Proses klasifikasi diawali dengan pembelajaran terhadap sistem klasifikasi untuk mendapatkan pengetahuan dari kumpulan data dalam dataset. Klasifikasi menggunakan sistem klasifikasi Naive Bayes.

1. Pengujian

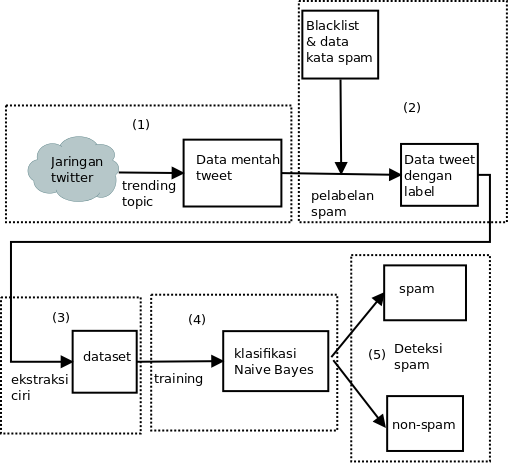
Dataset dibagi menjadi data training dan data testing. Data training digunakan untuk pembelajaran sistem klasifikasi, dan data testing untuk menguji sistem terhadap data yang belum pernah dilihat. Performa diukur menggunakan *confusion matrix*.

1. Penyusunan Laporan

Pada tahap akhir ini kemudian dilakukan penyusunan laporan.

1. Deskripsi Sistem

Blok sistem untuk deteksi spam yang diusulkan adalah sebagai berikut:



Gambar 3.1. Blok Sistem Deteksi Spam

Sistem bekerja melalui proses berikut:

1. Pengambilan data twit dari trending topik twitter.

Sistem mendapatkan data dari pencarian melalui Twitter API menggunakan keyword dari daftar trending topik di Indonesia. Output dari pencarian twit ini adalah tabel yang berisi twit yang diunduh. Sedangkan untuk aliran twit secara realtime, digunakan streaming API yang disediakan Twitter dengan output twit yang diperoleh untuk langsung diidentifikasi sebagai spam atau ham oleh sistem klasifikasi.

1. Pelabelan twit

Twit yang didapatkan kemudian diperiksa secara manual untuk dilabeli sebagai spam atau non-spam. Hasilnya adalah dataset yang berisi kumpulan twit yang sudah ditandai dengan spam dan tidak spam (ham). Sebelum ekstraksi fitur, dilakukan praproses terhadap data. Praproses dilakukan untuk menyiapkan data sehingga bisa menjadi bahan yang siap untuk ekstraksi fitur yang nanti digunakan untuk pembelajaran sistem klasifikasi. Praproses meliputi pembersihan data, penyesuaian tipe karakter atau casefolding, konversi kata, lalu tokenisasi, yaitu mengurai twit ke dalam kata. Setelah itu dilakukan stemming, yaitu mengubah kata-kata bentukan ke dalam bentuk dasar.

1. Ekstraksi Ciri atau Fitur.

Eksraksi fitur dilakukan terhadap twit yang berlabel menggunakan analisa linguistik bahasa untuk menghasilkan dataset akhir yang akan dipelajari oleh sistem klasifikasi.

1. Training.

Sebelum sistem klasifikasi bisa melakukan deteksi spam, harus dilakukan dilakukan training terlebih dahulu, yaitu proses pembelajaran dengan menyerap pengetahuan berdasarkan fitur dari dataset yang tersedia. Dataset yang diperoleh dari proses di atas kemudian dibagi dalam training set dan dataset. Training set digunakan untuk melatih komponen sistem klasifikasi, sehingga mendapatkan pengetahuan yang diperlukan untuk mendeteksi spam.

1. Deteksi spam. Komponen pendeteksi spam, menerima input dari twit yang didownload user, dan memberikan informasi kepada user, jika twit tersebut dicurigai sebagai spam. Sistem klasifikasi akan memberikan nilai True apabila twit yang masuk diidentifikasi sebagai spam, dan memberikan nilai False jika tidak teridentifikasi sebagai spam.
2. Pengumpulan Data

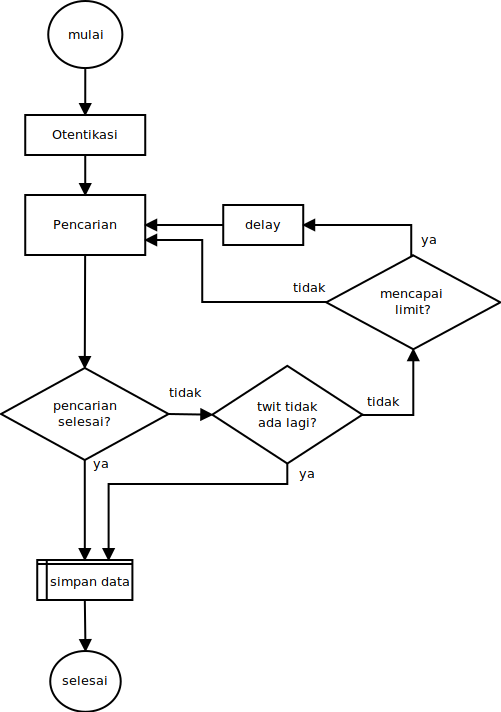
Untuk penyiapan dataset, dilakukan pengumpulan data berupa pesan-pesan twitter pada beberapa topik yang berpotensi dimasuki banyak spam. Untuk mendapatkan dataset digunakan twitter API melalui akses developer. Kemudian dilakukan pecarian pada topik-topik yang sedang hangat pada trending topik di indonesia. Untuk otentikasi dan pencarian twit digunakan script python. Selanjutnya pembuatan skript untuk pencarian pesan twitter dan penyimpanannya dalam file excel untuk pemrosesan selanjutnya.

Untuk mendapatkan akses adalah melakukan otentikasi. Hal ini diperllukan karena untuk menggunakan Twitter REST API pengguna harus menyertakan credential pada permintaan pencarian. Twitter menyediakan Oauth API untuk akses yang terotorisasi. Untuk melihat twit-twit yang bisa diakses publik digunakan otentikasi khusus untuk aplikasi.

Segmen Program 3.1 Otentikasi

1. auth = tweepy.AppAuthHandler(consumer\_key,consumer\_secret)
2. api = tweepy.API(auth, wait\_on\_rate\_limit=True, \
3. wait\_on\_rate\_limit\_notify=True)

Pengambilan data dilakukan dengan menggunakan search API standard dari twitter. Penggunaan search API standard memungkinkan untuk melakukan pencarian menggunakan keyword tertentu. Search API standard melakukan pencarian terhadap sampling dari twit yang dipublikasikan hingga 7 hari ke belakang. Search API standard difokuskan pada relevansi twit yang dicari, sehingga hasilnya tidak komplet keseluruhan twit pada database twitter. Fitur ini hanya terdapat dalam search API premium[[22]](#footnote-22). Untuk menghindari penyalahgunaan, twitter melakukan pembatasan seberapa sering kita melakukan pencarian dari satu alamat. Untuk itu sebelum batas pencarian tercapai, dilakukan tenggang waktu untuk pencarian berikutnya. Proses pencarian twit untuk mendapatkan dataset bisa dilihat pada gambar berikut.



Gambar 3.2

Diagram alur pencarian twit

Berikut script untuk pencarian di twitter:

Segmen Program 3.2 Pencarian twit

1. while tweetCount < maxTweets:
2. try:
3. if (max\_id <= 0):
4. if (not sinceId):
5. new\_tweets = api.search(q=searchQuery, \
6. count=tweetsPerQry,lang='id')
7. else:
8. new\_tweets = api.search(q=searchQuery, \
9. count=tweetsPerQry, \
10. since\_id=sinceId,lang='id')
11. else:
12. if (not sinceId):
13. new\_tweets = api.search(q=searchQuery, \
14. count=tweetsPerQry,
15. max\_id=str(max\_id - 1),lang='id')
16. else:
17. new\_tweets = api.search(q=searchQuery, \
18. count=tweetsPerQry, \
19. max\_id = str(max\_id - 1), \
20. since\_id=sinceId,lang='id')
21. if not new\_tweets:
22. print("No more tweets found")
23. break
24. for twit in new\_tweets:\
25. #f.write(jsonpickle.encode(twit.\_json, \
26. unpicklable=False) +'\n')
27. if 'RT @' not in twit.text:
28. row = [twit.id, twit.text, \
29. twit.user.screen\_name,\
30. twit.created\_at,strftime("%Y-%m-%d %H:%M:%S", gmtime())]
31. csv\_writer.writerow(row)
32. tweetwrited +=1
33. #else:
34. # print(twit)
35. tweetCount += len(new\_tweets)
36. print("Downloaded {0} tweets, writed {1}".format(tweetCount,tweetwrited))
37. time.sleep(60)
38. max\_id = new\_tweets[-1].id
39. except tweepy.TweepError as e:
40. # Just exit if any error
41. print("some error : " + str(e))
42. break

Dataset berupa tabel berisi text-text twitter. Teks twitter berisi banyak sekali atribut. Untuk penelitian ini atribut yang disimpan adalah atribut utama saja yaitu id, created\_at, text. Twit diterima dalam format JSON, lalu disimpan dalam bentuk tabel excel. Untuk akses twitter digunakan library tweepy. Untuk pengolahan data digunakan library pandas. Untuk komputasi ilmiah digunakan numpy.

1. Praproses Data

Untuk pengolahan lebih lanjut, harus dilakukan praproses terhadap data yang didapatkan. Dari twit-twit yang diterima, dipisahkan antara spam dan ham, kemudian disimpan dalam file yang berbeda. Tahapan praproses ini meliputi:

1. Data cleaning, yaitu menghilangkan beberapa informasi yang tidak relevan, di antaranya adalah: alamat url pada akhir twit, hashtag, @, tanda retweet
2. Casefolding, yaitu dengan mengubah semua huruf menjadi huruf kecil dengan tujuan mengatasi kesalahan ortographic model pada tahapan preprocessing selanjutnya.
3. Konversi kata, yaitu mengganti kata-kata yang tidak baku ke dalam bentuk baku. Pada pesan singkat, pengguna biasa menggunakan singkatan. Untuk itu dibuat kamus untuk substitusi ini. Dengan ini diharapkan bisa mengurang dimensionalitas data, dan mengurangi kesalahan penghitungan frekuensi kata.
4. Filtering, yaitu menghapus kata-kata yang tidak relevan untuk pembedaan spam dan ham. Kata-kata ini adalah kata yang umum terpakai, biasa disebut stopwords. Penghilangan ini akan menyederhanakan dimensi data dan mempercepat pemprosesan kata. Penghilangan stopword ini menggunakan library python untuk pemprosesan bahasa natural indonesia yaitu sastrawi.
5. Stemming, yaitu mengembalikan kata-kata bentukan ke bentuk kata dasar. Langkah ini bertujuan sebagaimana proses substitusi. Proses ini juga menggunakan modul pada library python sastrawi. Stemmer pada sastrawi menggunakan algoritma nazief dan Adriani[[23]](#footnote-23). Tahapan algoritma tersebut meliputi:
   1. Langkah pertama adalah dengan melihat pada daftar kata dasar di kamus apakah kata yang diperiksa merupakan kata dasar. Jika kata tersebut terdapat dalam daftar kata dasar, maka proses selesai.
   2. Menghilangkan inflection suffixes (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”). Jika kata berupa particles (“-lah”, “-kah”, “-tah” atau “-pun”) maka langkah ini diulangi lagi untuk menghapus Possesive Pronouns (“-ku”, “-mu”, atau “-nya”).
   3. Menghilangkan derivational suffix (imbuhan turunan). Hilangkan imbuhan -i, -kan, -an.
   4. Menghilangkan derivational prefix (awalan turunan). Hilangkan awalan be-, di-, ke-, me-, pe-, se- dan te-.
   5. Bila dari langkah 4 di atas belum ketemu juga. Maka lakukan analisis apakah kata tersebut masuk dalam tabel diambiguitas kolom terakhir atau tidak.
   6. Bila dari langkah 4 di atas belum ketemu juga. Maka lakukan analisis apakah kata tersebut masuk dalam tabel diambiguitas kolom terakhir atau tidak.
   7. Bila semua proses di atas gagal, maka algoritma mengembalikan kata aslinya.

Kamus akar kata yang digunakan sastrawi berbasis pada kamus di kateglo.com

Tahapan praproses akan dijelaskan lebih detil pada bab berikutnya.

1. Ekstraksi Fitur

Sebelum dilakukan training terhadap algoritma klasifikasi, harus dilakukan ekstraksi fitur terlebih dahulu. Dengan tahap ini, dokument teks yang tidak terstruktur diubah menjadi sejumlah atribut yang seragam sehingga algoritma bisa belajar darinya. Fitur yang digunakan untuk penelitian ini berbasis pada model bahasa n-gram. Untuk perbandingan performa dari sistem klasifikasi dipilih beberapa alternatif, yaitu:

1. Unigram/*bag-of-words*; fitur ini menghitung berapa kali setiap kata muncul dalam korpus ham dan spam.
2. Bigram; Jika pada unigram hanya dilakukan penghitungan terhadap kemunculan tiap kata, tanpa memperhatikan kemungkinan munculnya frasa atau kata majemuk. N-gram menghitung kemunculan kata yang berurutan. Untuk teks twitter yang pendek, hanya dilakukan penghitungan kemunculan dua kata yang berurutan atau bigram.
3. Term-frequency/inverse document frequency (tf-idf); N-gram hanya menghitung kemunculan kata atau frasa, tanpa memperhitungkan panjang pendek dokumen. Hal ini berarti tidak mempertimbangkan seberapa besar pengaruh kata atau frasa tersebut terhadap dokumen di mana dia muncul. Term frequency memperhitungkan hal ini. Di samping itu untuk mengurangi noise yang disebabkan oleh kata yang umum terpakai akan tetapi tidak memberi informasi penting untuk pembedaan spam dan ham digunakan inverse document frequency atau IDF.

(3.1)

Di mana tf(t,i) adalah term frequency untuk term t pada contoh training ke i, M adalah jumlah total contoh training, dan mt adalah jumlah contoh trining yang mengandung term t.

1. Klasifikasi

Untuk klasifikasi diimplementasikan multinomial Naive Bayes. Multinomial Naive Bayes menggunakan model dokumen multinomial yang merepresentasikan sebuah dokumen sebagai vektor fitur dengan elemen integer bernilai frekuensi dari kata –kata di dalam dokumen. Model klasifikasi teks multinomial bisa diestimasikan sebagai berikut[[24]](#footnote-24):

1. Kosakata V didefinisikan, jumlah kata-kata di dalam kumpulan kosakata menunjukkan dimensi dari vektor fitur
2. Pada data training dihitung:
   1. N , adalah jumlah total dokumen
   2. adalah jumlah dokumen pada kelas C=k, (dalam hal ini ham atau spam)
   3. Estimasikan likelihood dengan rumus:

(3.2)

* 1. Estimasikan prior P(C=k) dengan

1. Evaluasi

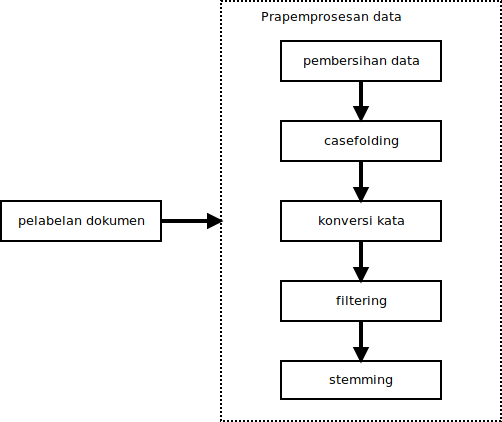
Untuk evaluasi performa digunakan metode *k-fold cross validation*. Dengan metode ini, dataset dibagi dalam k bagian, satu bagian digunakan untuk testing, bagian lain digabungkan untuk training, proses ini diulang sebanyak k kali, dengan bagian yang berbeda sebagai testing set. Hasilnya dirata-ratakan sehingga mendapatkan estimasi yang lebih akurat terhadap performa model sistem klasifikasi dibandingkan dengan pembagian dataset set menjadi dua bagian saja secara sederhana F1-score untuk setiap fold, lalu dihitung rata-ratanya. Validasi silang dalam penelitian ini dilakukan sebanyak 10 kali.

Penilaian terhadap sistem klasifikasi dilakukan dengan menggunakan *confusion matrix*. Masing-masing pilihan fitur akan dibandingkan performanya berdasar F1-score dan confusion matrix ini.

# BAB IV

PEMBENTUKAN DATASET

Proses awal untuk pembentukan dataset adalah mengunduh pesan-pesan twitter dari database twitter. Setelah data terkumpul, kemudian dilakukan pelabelan atau pemisahan antara spam dan ham. Kualitas dataset sangat menentukan performa sistem klasifikasi yang dibuat, dalam hal ini adalah filter untuk spam. Sebelum dataset bsa digunakan untuk pembelajaran bagi sistem klasifikasi, dataset harus disiapkan terlebih dahulu. Penyiapan dataset ini biasa disebut dengan data preprocessing atau praproses data. Praproses data ini meliputi data cleaning, casefolding atau penyeragaman tipe huruf, konversi kata atau substitusi, filtering, dan stemming.



Gambar 4.1

Pelabelan dan praproses Data

1. Pengumpulan Dan Pelabelan Data

Twitter menyediakan standard API bagi developer untuk mengakses data publik. Setelah melakukan otentikasi dengan Oauth, kemudian dilakukan pencarian menggunakan keyword yang menjadi trending topik. Ada 5 trending topik yang dipilih sebagai keyword pencarian dengan rentang waktu akhir mei hingga awal Juni 2018. Trending topik yang dipilih adalah “#GanjarTakTakutPakDirman”, “#PersijaDay”, “#HariKebangkitanNasional”, “#NasionalismeZamanNow”, dan “#TarawihInstragramable”. Tabel 4.1 dan Tabel 4.2 menunjukkan contoh twit yang didapatkan dari pencarian dengan keyword #PersijaDay dan #GanjarTakTakutPakDirman yang sedang trending pada tanggal 29 Mei 2018 hingga 6 Juni 2018. Pada Tabel 4.1, twit nomer 3, 8, dan 9 jelas merupakan spam. Sedangkan twit nomer 7 agak sulit untuk menggolongkan apakah termasuk spam atau ham. Twit selainnya adalah ham. Pada Tabel 4.2 twit nomer 4, 6, dan 9 merupakan spam. Twit spam dan ham disimpan dalam tabel yang berbeda untuk proses selanjutnya, sedangkan twit yang sulit dimasukkan baik sebagai spam maupun ham diabaikan.

Tabel 4.1

Contoh twit dengan hashtag #PersijaDay

|  |  |  |
| --- | --- | --- |
| nr | text | label |
| 1 | @Persija\_Jkt Menang can ! #COYT 🐯🔴 #PersijaDay 👆🍻 #ElClassico | 0 |
| 2 | RT @JakmaniaBoeloes: Ciptakan momen paling indah malam ini. LO HARUS MENANG @Persija\_Jkt !! #PersijaDay | 0 |
| 3 | Vibrator Kapsul Double Alat Bantu Sex Wanita CALL/WA:081321061991 Hrg 400rb #pemuaswanita #RecehkanTwitter #memek… https://t.co/aAEGbBuoq2 | 1 |
| 4 | @infosuporter Indonesia 3 - 0 Thailand #TimnasDay #PersijaDay | 0 |
| 5 | RT @DikySoemarno: Setelah sekian lama, hari ini bertemu kembali.. Macan kembali mengamuk.. korban hari ini, Persebaya.. #PersijaDay #HariIn… | 0 |
| 6 | RT @jak\_srengseng: #PersijaDay Hari ini wajib 3 poin | 0 |
| 7 | RT @SupportPersija: Kalo emang mau usik anak the jak pakai atribut asli! Jangan provokasi! Jangan sok paling berani! Pada intinya kalian ga… | 0 |
| 8 | Yuhu yg Kuat puasanya ya #wanita #cewekbispak #sholehah #viral #wanitasholehah #recehkantwitter #TrendingTopics… https://t.co/5ajqNtPeg7 | 1 |
| 9 | HATIPOKER kini Hadir Membagi Hadiah Free Tshirt JERSEY + FREECHIP Menyambut Piala Dunia 2018 #hatipoker… https://t.co/Stqoj49caJ | 1 |
| 10 | Pemanasan Buat Tar malem @superglad\_band ... Sambil streaming @Persija\_Jkt #PersijaDay #SupergladDay RADIO NET SHO… https://t.co/HzbIeiPuUL | 0 |

Tabel 4.2

Contoh twit tagar #GanjarTaktakutPakDirman

|  |  |  |
| --- | --- | --- |
| nr | text | label |
| 1 | RT @JalalHusin: ganjar tak takut pak dirman..takutnya sama kasus E- KTP dan pergerakan masjid #GanjarTakTakutPakDirman | 0 |
| 2 | RT @firaun\_id: Hastag #GanjarTakTakutPakDirman isinya robot. @ganjarpranowo jangankan Pak Dirman, KPK saja takut sama Ganjar. Sudah jelas-j… | 0 |
| 3 | Hastag #GanjarTakTakutPakDirman isinya robot. @ganjarpranowo jangankan Pak Dirman, KPK saja takut sama Ganjar. Suda… https://t.co/h3njhhL2hy | 0 |
| 4 | SUKACAPSA(.)COM ADALAH AGENT DOMINO QQ AGEN DOMINO 99 DAN POKER ONLINE AMAN DAN TERPERCAYA #recehkantwitter selama… https://t.co/BaQKyFxzcD | 1 |
| 5 | iki toh pemimpine wkwkwk gak ono masa depane, #GanjarTakTakutPakDirman #GanjarKalahPrabowoMenang #GanjarYasin https://t.co/vrZD5oa9vb | 0 |
| 6 | Yuk Gabung aja di bolafinal Situs Betting Online Terpercaya dan Terlengkap di Indonesia minimal Deposit dan withdr… https://t.co/O6NUxYpaU9 | 1 |
| 7 | #GanjarTakTakutPakDirman Next kacung jateng lengserkan....! | 0 |
| 8 | RT @negativisme: melihat hasil berbagai survey, testimoni warga dan bukti kemajuan diberbagai bidang, wajar kalo Pak Ganjar kembali memim… | 0 |
| 9 | RT @JalalHusin: ganjar tak takut pak dirman..takutnya sama kasus E- KTP dan pergerakan masjid #GanjarTakTakutPakDirman | 0 |
| 10 | JADWAL PERTANDINGAN 31/05/2018 DAN PROMO TERBARU KUIS TEBAK JUARA PIALA DUNIA 2018 BERSAMA GALAXY88 TOTAL HADIAH PU… https://t.co/8ywbvK6O4q | 1 |

1. Pembersihan Data

Pembersihan data dilakukan agar teks bersih dari simbol atau karakter yang tidak standard, sehingga memudahkan pemprosesan selanjutnya. Simbol-simbol tersebut hanya akan menambah kompleks data dan mengurangi performa dari sistem klasifikasi.

Berdasarkan tabel twit yang ditampilkan sebelum ini dan tabel twit spam pada

Tabel 4.3, ada beberapa karakteristik data yang bisa dijadikan acuan untuk pembersihan data sebagai berikut:

* Ada simbol-simbol tertentu, misalnya # untuk hashtag, @ untuk mention ke pengguna lain
* Adanya emoji dan simbol non-ASCII yang lain
* Istilah dalam Twitter yang tidak relevan untuk analisa konten, misalnya RT untuk retweet.
* Adanya tanda spasi yang lebih dari satu berurutan, ganti baris.
* Banyak twit yang mengandung spasi di awal dan akhir kalimat.
* Adanya entitas html dalam twit, misalnya “&amp”.
* Adanya banyak angka, baik itu berupa nomer telepon, nilai uang, pin, kata perulangan, dan lain-lain.
* Adanya tautan alamat web yang sudah mengalami kode penyingkatan

Tabel 4.3

Contoh tweet spam

|  |  |
| --- | --- |
| No | Twit spam |
| 1 | TITAN GEL ASLI PEMBESAR DAN PANJANG PENIS ALAMI TLP/WA:081321061991 Hrg 450rb #UCLfinal #Retwet #ViralVideosâ€¦ https://t.co/raIuh9mdpu |
| 2 | Yuk ilangin #jerawat bareng2 pake #MaskerSpirulina. Kulit wajah sehat, bersih, kencang sdh b terasa lgs Info 089653534745 pin:52023DE5 |
| 3 | 4 Rekor Cristiano Ronaldo Ini Sulit Dipecahkan  https://t.co/LPRH9r8VwG  #NasionalismeZamanNow  #KARMABAIKANTV126â€¦ https://t.co/BgURzq8f5P |
| 4 | https://t.co/sGhs4hkfpa  Mohon likenyaa yaaaa  #NasionalismeZamanNow |
| 5 | game khusus 18 tahun keatas - https://t.co/JU2my4wvCn kalah buka baju ampe bugil #WannaOneinJKT #PersijaDay… https://t.co/ujaetHep8t |
| 6 | Hai kak, @rockymengko Follow @InfoSehatID yuk.. untuk dapat info paling update seputar Kesehatan.., pasti di Folback ! |
| 7 | Jasa Tambah Visitor Web, Blog dan Toko Online | 100% real human &amp; uniqe IP Address | Info sms 0823.1212.2323  ~ [#NasionalismeZamanNow] ~ 8 |
| 8 | #OpenBooking  @AdelliaBeauty  #Batam  WA: 0822 8798 4055  #bobatam #bisyarbatam #KulinerLendir  #PersijaDay  https://t.co/Jl49PuAifJ |

Dari beberapa karakteristik di atas perlu dianalisa lagi, agar penghapusan tidak menghilangkan informasi-informasi penting yang justru diperlukan untuk identifikasi spam. Pada tabel 4.3 tampak bahwa spam cenderung menggunakan banyak tagar. Hal ini karena penyebar spam menginginkan twitnya tampil pada beberepa trending topik. Di samping itu juga ada pengggunaan alamat url dan nomor telepon ataupu whatsapp. Karena pesan twitter yang sangat pendek, diusahakan seminimal mungkin informasi yang hilang. Untuk itu diusahakan untuk melakukan penggantian beberapa simbol atau terminologi yang khas pada topik tertentu menjadi suatu istilah yang bisa digunakan untuk ekstraksi fitur setelah proses ini. Penggantian dilakukan pada proses konversi kata setelah ini.

Beberapa langkah pada tahap *cleaning* yang dilakukan adalah:

* Menghapus kata bertagar yang menjadi kata pencarian. Hal ini karena kata muncul baik pada spam maupun ham. Sedangkan tagar yang lain cukup dihilangkan tanda tagarnya. karena bisa menjadi petunjuk untuk spam.
* Menghilangkan tanda baca, simbol dan karakter yang tidak standard.
* Spasi yang dobel atau lebih diganti menjadi spasi tunggal
* Menghilangkan spasi di awal dan akhir teks
* Menghapus kata “RT” yang menunjukkan retweet

Penghapusan dilakukan dengan memanfaatkan manipulasi string yang disediakan oleh bahasa pemrograman.

Algoritma 4.1 Data Cleaning

Fungsi: memberihkan data dari html escaping dan simbol non-ASCII

1. Html-unescaping
2. Ganti tanda baca menjadi spasi
3. Hapus kata RT
4. Trim kalimat
5. Ganti tanda ganti baris dengan spasi
6. Ganti spasi yang dobel atau lebih menjadi spasi tunggal

Segmen Program 4.1. Data Cleaning

1. hams['text']= hams['text'].str.lower()
2. hams['text']= hams['text'].str.replace('#persijaday','')
3. hams['text']= hams['text'].str.replace('#ganjartaktakutpakdirman','')
4. hams['text']= hams['text'].str.replace('#harikebangkitannasional','')
5. hams['text']= hams['text'].str.replace('#nasionalismezamannow','')
6. hams['text']= hams['text'].str.replace('#tarawihinstagramable','')
7. hams['text']= hams['text'].str.replace(r'(Rp)|(\$)', 'matauang')
8. hams['text']= hams['text'].str.replace(r'(\d+)((\+)|)(\d+)(\')','menitke')
9. hams['text']= hams['text'].str.replace(r'\brt\b',' ')
10. hams['text']= hams['text'].str.replace(r'\b(0|[1-9]\d\*)-(0|[1-9]\d\*)-(0|[1-9]\d\*)\b','formasibola')
11. hams['text']= hams['text'].str.replace(r'\b(\d+)(\s+|)(-|(vs))(\s+|)(\d+)\b','skor')
12. hams['text']= hams['text'].str.replace(r'@.\*?(?=\s|$)','mentionke')
13. hams['text']= hams['text'].str.replace(r'[^a-zA-Z0-9 :]', '')
14. hams['text']= hams['text'].str.replace(r'\b[\w\-.]+?@\w+?\.\w{2,4}\b','emailaddr')
15. hams['text']= hams['text'].str.replace(r'(http[s]?\S+)|(\w+\.[A-Za-z]{2,4}\S\*)','urladdr')
16. hams['text']= hams['text'].str.replace( r'\b(\+\d{1,2}\s)?\d?[\-(.]?\d{3}\)?[\s.-]?\d{3}[\s.-]?\d{4}\b', 'telp')
17. hams['text']= hams['text'].str.replace(r'(\b\d+\,\d+\b)|(\b\d+\b)', 'nilaiangka')
18. hams['text']= hams['text'].str.replace(r'[^\w\d\s]', ' ')
19. hams['text']= hams['text'].str.replace(r'\s+', ' ')
20. hams['text']= hams['text'].str.replace(r'^\s+|\s+?$', '')

Kode baris 1 adalah untuk casefolding, dilakukan di awal sehingga penggantian kata bisa lebih mudah dilakukan.

Contoh hasil proses pembersihan data bisa dilihat pada tabel 4.4

.

Tabel 4.4

Contoh hasil pembersihan data

|  |  |
| --- | --- |
| Sebelum Pembersihan | Setelah Pembersihan |
| "Tarawih day 5 done" judul sgnya gitu,gataunya kebanyakam duduk buat boomerangin org yg lagi sholat:) #TarawihInstagramable | tarawih day nilaiangka done judul sgnya gitu gataunya kebanyakam duduk buat boomerangin org yg lagi sholat |
| Terobati dg kemenangan persija #PersijaDay | terobati dg kemenangan persija |
| Alhamdulillah lancar, indahnya berbagi di bulan penuh berkah. #PersijaDay https://t.co/iHzMUsKyl6 | alhamdulillah lancar indahnya berbagi di bulan penuh berkah urladdr |
| Lebih dari 30 squad Persija, hampir setengah nya pemain uzur dan gak jelas,mending squad nya di kurangi dan gajinya… https://t.co/HBMPMCWC9P | lebih dari nilaiangka squad persija hampir setengah nya pemain uzur dan gak jelasmending squad nya di kurangi dan gajinya urladdr |
| Kalian terlalu heboh membicarakan #RoyalWedding sampe kalian lupa kalo hari ini #HariKebangkitanNasional | kalian terlalu heboh membicarakan royalwedding sampe kalian lupa kalo hari ini |
| Sebagai inspektur upacara #HariKebangkitanNasional hari ini adalah Direktur Utama #KantorBeritaIndonesia Meidyatamaâ€¦ https://t.co/geyhOZQSZ1 | sebagai inspektur upacara hari ini adalah direktur utama kantorberitaindonesia meidyatama urladdr |
| Selamat Hari Kebangkitan Nasional Jadikan momen ini sebagai simbol persatuan Indonesia Bangkit, Bersatu Indonesiakuâ€¦ https://t.co/WOpAfXgJkJ | selamat hari kebangkitan nasionaljadikan momen ini sebagai simbol persatuan indonesiabangkit bersatu indonesiaku urladdr |
| Ga boleh nge-DM imam saat traweh sdag berlngsung #tarawihinstagramable https://t.co/xbGN83VjJy | ga boleh ngedm imam saat traweh sdag berlngsung urladdr |
| Hadiri Acara Santri, Ganjar Ditanya Soal Anggaran untuk Ponpes https://t.co/9l0N58SWpL #GanjarTakTakutPakDirman | hadiri acara santri ganjar ditanya soal anggaran untuk ponpes urladdr |
| Angka Kemiskinan Turun 2,21 Persen pada Pemerintahan Ganjar, Ini Penjelasan Taj Yasin #GanjarTakTakutPakDirman | angka kemiskinan turun nilaiangka persen pada pemerintahan ganjar ini penjelasan taj yasin |
| Kiper Persija lagi seneng ngelawak yah? #PersijaDay | kiper persija lagi seneng ngelawak yah |

1. Casefolding

Tidak ada standard dalam penulisan pesan di twitter, sehingga pengguna sering berbeda dalam menuliskan pesannya. Ada yang huruf besar, ada yang huruf kecil. Kata yang berada di awal kalimat sering diawali dengan huruf besar. Perbedaan penulisan ini bisa menimbulkan bias dalam penghitungan kemunculan kata pada dokumen. Untuk menghindari problem ini, maka dilakukan penyeragaman huruf untuk keseluruhan kata di dalam dokumen, dalam hal ini seluruh isi dokumen diubah ke dalam huruf kecil.

Tabel 4.5

Contoh hasil casefolding

|  |  |
| --- | --- |
| Sebelum casefolding | Sesudah casefolding |
| Hadiri Acara Santri, Ganjar Ditanya Soal Anggaran untuk Ponpes | hadiri acara santri ganjar ditanya soal anggaran untuk ponpes |
| Kemenangan ganjar di jateng menjadi "HARGA MATI" | kemenangan ganjar di jateng menjadi "harga mati" |
| Selamat Hari Kebangkitan Nasional Jadikan momen ini sebagai simbol persatuan Indonesia Bangkit, Bersatu Indonesiaku | selamat hari kebangkitan nasionaljadikan momen ini sebagai simbol persatuan indonesiabangkit bersatu indonesiaku |

1. Konversi kata

Konversi kata adalah proses penggantian kata atau istilah dalam twit yang bertujuan meningkatkan kualitas model bahasa yang dibentuk. Karakteristik bahasa pada twit adalah banyak digunakannya kata-kata pergaulan yang tidak baku, banyaknya singkatan , adanya tautan web yang dikode singkat sehingga tidak ada yang sama, penulisan untuk menekankan emosi, misalnya “gooolll”, “mantappp”, dan lain-lain. Untuk itu yang dilakukan dalam konversi kata ini meliputi:

* Penggantian semua alamat web dengan kata pengganti “urladdr”
* Penggantian semua nomer telepon dengan kata tunggal “telnr”
* Penggantian semua alamat email dengan “emailnr”
* Mengganti beberapa simbol angka dengan istilah yang sesuai dengan topik, di antaranya:
  + skor sepakbola, yaitu 3-0, atau 1 – 0 atau 2 vs 0 dengan istilah “skor”
  + formasi bola, 4-3-3 atau 4-4-2, diganti dengan “formasibola”
  + angka menit pada pertandingan bola, biasa tertulis 1’, 3’, 90’, atau 90+3’ diganti dengan “playtime”
  + Angka yang diawali dengan tanda mata uang diganti dengan “jumlahuang”
  + angka selainnya yang berdiri sendiri diganti dengan kata”nilaiangka”
* Kata yang diawali dengan @ (mention) diganti dengan kata “mention”
* Menggant kata perulangan dengan kata tunggal, misal “teman2” menjadi “teman”
* Mengganti kata tidak baku menjadi kata baku

Tabel 4.6

Contoh kata yang disubstitusi

|  |  |
| --- | --- |
| Kata yang diganti | Kata pengganti |
| ga, nga, ngga, kaga | tidak |
| temen | teman |
| ka, kaka, kak | kakak |
| gool, goolll | gol |
| kudu | harus |
| inget | ingat |
| brenti | berhenti |
| sdh, dah | sudah |
| cepet | cepat |

Kata-kata yang hendak dikonversi dan kata penggantinya disimpan dalam data teks untuk memudahkan pemutakhiran data. Sistem membaca daftar kata tersebut, menyimpannya dalam kamus kosakata untuk penggantian kata.

Segmen Program 4.2 Kode proses penggantian kata

1. hams['text']= hams['text'].str.replace(r'\bg+o+l+\b', 'gol')
2. hams['text']= hams['text'].str.replace(r'\b(n|ng|ka|)ga+(k+|)\b', 'tidak')
3. hams['text']= hams['text'].str.replace(r'\b(temen)\b', 'teman')
4. hams['text']= hams['text'].str.replace(r'\b(cepet)\b', 'cepat')
5. hams['text']= hams['text'].str.replace(r'\b(kudu)\b', 'harus')
6. hams['text']= hams['text'].str.replace(r'\b(kq)|(ko)\b', 'kok')
7. hams['text']= hams['text'].str.replace(r'\b(menang+)\b', 'menang')
8. hams['text']= hams['text'].str.replace(r'\b(gw)|(gue)\b', 'saya')
9. hams['text']= hams['text'].str.replace(r'\b(s|)(u|)d(a|)(h|)\b', 'sudah')
10. hams['text']= hams['text'].str.replace(r'\b(brenti)\b', 'berhenti')
11. hams['text']= hams['text'].str.replace(r'\bg+(e+|a+)(i|e)s+\b', 'guys')
12. hams['text']= hams['text'].str.replace(r'\b(ha|a|)y+(u+|o+)(k+|)(s+|)\b', 'ayo')

Tabel 4.7

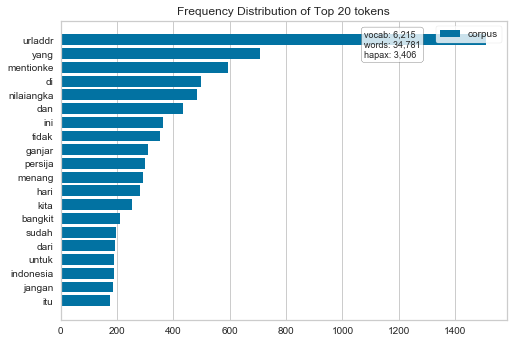
Contoh hasil konversi kata

|  |  |
| --- | --- |
| Teks awal | Teks setelah konversi |
| loh loh kq kaga offside yaa | loh loh kok tidak offside yaa |
| goooooooll persijaa skor | gol persijaa skor |
| hayoo can raih poin nilaiangka poin nilaiangka harga mati can nilaiangka | ayo can raih poin nilaiangka poin nilaiangka harga mati can nilaiangka |

1. Filtering

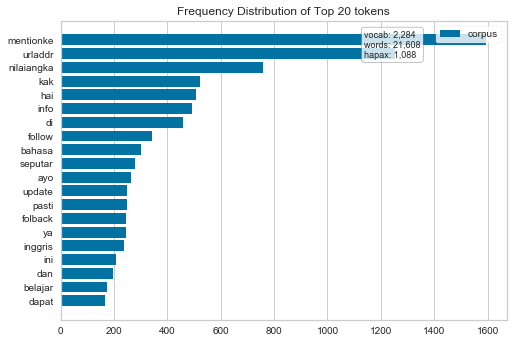
Dalam pesan-pesan pada media sosial sering muncul kata-kata yang biasa digunakan, sehingga frekuensi kemunculan sangat tinggi baik di pesan spam maupun ham. Kata-kata yang demikian ini tidak memberikan informasi yang bisa membedakan apakah pesan twitter yang mengandungnya merupakan spam atau ham. Di pihak lain untuk memprosesnya membutuhkan waktu sehingga mengurangi performa klasifikatornya. Kata-kata ini biasa disebut stopword. Filtering menghapus stopword ini sehingga bisa mempercepat proses training. Karena pesan twitter yang sangat pendek, maka filtering dilakukan seminimal mungkin.

Gambar 4.2 dan Gambar 4.3 menunjukkan 20 kata dengan distribusi tertinggi di spam dan ham. Tampak bahwa urladdr, mention, dan nilaiangka mempunyai frekuensi tertinggi baik di spam maupu ham. Untuk urladdr, berdasarkan berdasarkan analisa terhadap spam dan ham, didapatkan bahwa urladdr pada akhir twit umumnya ada baik pada spam maupun ham. Sedangkan urladdr pada tengah twit, cenderung pada spam. Oleh karena itu urladdr pada akhir pesan bisa menjadi kandidat untuk dihilangkan. Demikian juga mention dan nilaiangka bisa dihilangkan karena tidak memberi informasi lebih untuk pembedaan spam maupun ham. Stopword lain yang bisa dipilih untuk dihilangkan adalah di dan ini



Gambar 4.2

Contoh distribusi kata pada twit ham



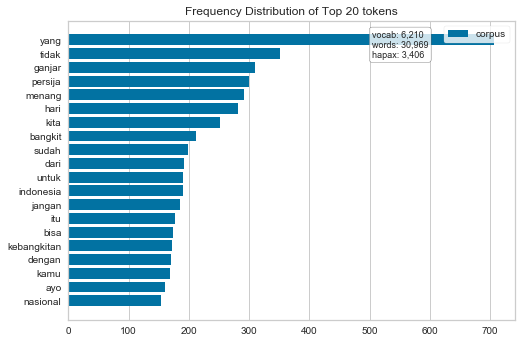
Gambar 4.3

Contoh Distribusi Kata Pada Spam

Segmen Program 4.3 Fungsi Praproses Data

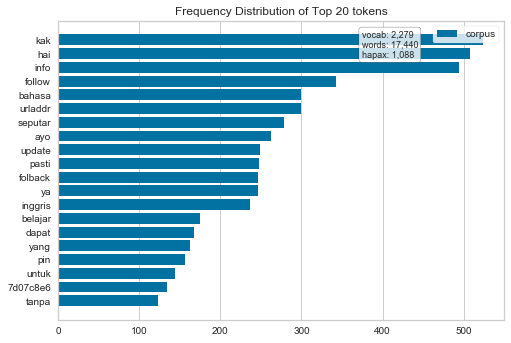
1. def sub\_clean(text,pengganti):
2. text = html.unescape(text)
3. text = text.lower()
4. text = text.strip()
5. for k,v in pengganti.items():
6. text=re.sub(k,v,text)
7. return text
8. def wordfilter(text):
9. text = text.lower()
10. text = text.strip()
11. for k,v in hapus.items():
12. text=re.sub(k,v,text)
13. return text
14. def praproses(text,stemmer,stop,gram=1):
15. text = sub\_clean(text,pengganti)
16. text = stemmer.stem(text)
17. text = stop.remove(text)
18. text = wordfilter(text)
19. words = text.split()
20. if gram > 1:
21. word = []
22. for i in range(len(words) - gram + 1):
23. word += [' '.join(words[i:i + gram])]
24. return word
25. return words

Setelah penghilangan urladdr pada akhir pesan twit, bisa dilihat pada Gambar 4.4 bahwa pada ham urladdr tidak masuk lagi pada 20 kata dengan distribusi tertinggi.



Gambar 4.4

Contoh Frekuensi Kata Setelah Filtering



Gambar 4.5

Contoh Frekuensi Kata Spam Setelah Filtering

Tabel 4.8

Contoh Hasil Filtering

|  |  |
| --- | --- |
| Sebelum Filtering | Sesudah Filtering |
| openbooking mentionke malang wa nilaiangka nilaiangka nilaiangka kulinerlendir urladdr | openbooking malang wa kulinerlendir |
| calon gubernur jawa tengah ganjar pranowo di temu kangen pecinta unggas di kaliwungu kendal dok tim media ganj urladdr | calon gubernur jawa tengah ganjar pranowo temu kangen pecinta unggas kaliwungu kendal dok tim media ganj |
| jangan encer maen lu kek bubur diaduk ye mentionke kalo mao juara wajib menang tidak pake seri we lope you | jangan encer maen lu kek bubur diaduk ye kalo mao juara wajib menang tidak pake seri we lope you |
| mainkan sekarang urladdr dan menangkan jackpotnya rasakan sensasi bermain yang berbeda bersama urladdr | mainkan sekarang urladdr menangkan jackpotnya rasakan sensasi bermain yang berbeda bersama |
| cuma di urladdr kalian bisa mendapatkan thr dengan total ratusan juta rupiah wannaoneinjkt urladdr | cuma urladdr kalian bisa mendapatkan thr dengan total ratusan juta rupiah wannaoneinjkt |

1. Stemming

Stemming adalah pengubahan kata-kata bentukan ke dalam bentuk dasarnya. Tanpa stemming, kata dasar yang sama bisa dihitung secara berbeda, hal ini akan meningkatkan dimensi vektor fiturnya yang akan berpengaruh terhadap performa.

Sebelum dilakukan stemming, terlebih dahulu diakukan tokenisasi. Tokenisasi adalah proses pemotongan string input berdasarkan tiap kata penyusunnya dan dan membedakan karakter-karakter tertentu yang dapat diperlakukan sebagai pemisah kata.

Tabel 4.9

Contoh hasil stemming

|  |  |
| --- | --- |
| Sebelum Stemming | Sesudah Stemming |
| ganjar mengatakan sesuai dengan mimpi para pendiri bangsa bung karno hatta ia juga berjanji akan mensejahtera | ganjar kata sesuai dengan mimpi para diri bangsa bung karno hatta ia juga janji akan sejahtera |
| gampang kok tinggal maen aja tidak perlu klaim syarat ketentuan anda harus mendapatkan kartu kombinasi sudah | gampang kok tinggal maen aja tidak perlu klaim syarat tentu anda harus dapat kartu kombinasi sudah |
| peluang untuk persija jakarta melalui tendangan bebas belum bisa dimaksimalkan persija skor persipura | peluang untuk persija jakarta lalu tendang bebas belum bisa maksimal persija skor persipura |
| maaf bos saya bolos kerjanya hari persija maen | maaf bos saya bolos kerja hari persija maen |
| daftarkan id kamu sekarang juga urladdr mainkan live casino bersama dengan ribuan player lai | daftar id kamu sekarang juga urladdr main live casino sama dengan ribu player lai |
| hari gini masih percaya hestek yang tiba2 nangkring tt yang isinya twit sama semua wkwk | hari gin masih percaya hestek yang tiba2 nangkring tt yang isi twit sama semua wkwk |
| hai kak ayo follow rame2 gt gt untuk dapat info ter update seputar bisnis update banget pasti folback | hai kak ayo follow rame2 gt gt untuk dapat info ter update putar bisnis update banget pasti folback |
| cerita seks wanita bersuami urladdr ceritabokep ceritaseks ceritadewasa bintangbola agenbola | cerita seks wanita suam urladdr ceritabokep ceritaseks ceritadewasa bintangbola agenbola |
| lengkap sudah daryono makin tenang mainnya persija tinggal beli playmaker yang handal | lengkap sudah daryono makin tenang main persija tinggal beli playmaker yang handal |
| hai kak izin share info ya inilah cara unik mahir ngomong bahasa inggris tanpa kursus | hai kak izin share info ya ini cara unik mahir ngomong bahasa inggris tanpa kursus |
| hai kak follow ayo paling update seputar makassar menarik menambah wawasan pasti folback | hai kak follow ayo paling update putar makassar tarik tambah wawas pasti folback |
| nusantarannium semangat hari kebangkitan nasional diperingati tiap tahunnya supaya tidak luntur kita tidak lupa | nusantarannium semangat hari bangkit nasional ingat tiap tahun supaya tidak luntur kita tidak lupa |

Dari Tabel 4.9 tampak bahwa masih ada beberapa kata yang tidak berhasil diubah menjadi kata dasar. Hal ini bisa dikarenakan kamus kosakatanya yang kurang atau penulisan yang tidak standar. Walaupun demikian tujuan untuk mengurangi dimensi vektor fiturnya tetap bisa dicapai, sehingga dataset siap untuk diproses selanjutnya.

# BAB V

KLASIFIKASI

Pada bab ini akan dibahas lebih detail tentang proses klasifikasi. Sebelum proses training, dilakukan terlebih dulu ekstraksi fitur, yaitu menjadikan dokumen teks menjadi vektor yang bisa diproses untuk pembelajaran sistem klasifikasi.

1. Ekstraksi Fitur

Untuk ekstraksi fitur digunakan model bahasa n-gram. Langkah-langkah yang dilakukan adalah sebagai berikut:

1. Membentuk *bag-of-words*, setiap dokumen dipecah menjadi kumpulan kata-kata.
2. Menggabungkan setiap kata dalam urutan kata sesuai dengan gram yang dibentuk.
3. Menghitung term frequency dan inverse document frequency

* Term frequency adalah frekuensi dari sebuah kata di dalam dokumen. Jika di dalam dokumen berisi 100 kata terdapat kata kucing sebanyak 5 kali, maka tf(“kucing”)=0,05
* Inverse document frequency (idf) dari suatu istilah adalah ukuran seberapa penting istilah tersebut dalam keseluruhan korpus. Jika di dalam 1000 dokumen muncul 20 kali kata “kucing” yang tersebar di 10 dokumen, maka idfnya adalah 1000:10 = 100.

Pada penelitian ini dipilih tiga model bahasa yang akan dibandingkan, yaitu bag-of words, bigram, dan bigram dengan term frekuensi dan document frequency.

1. Bag-of-words atau unigram

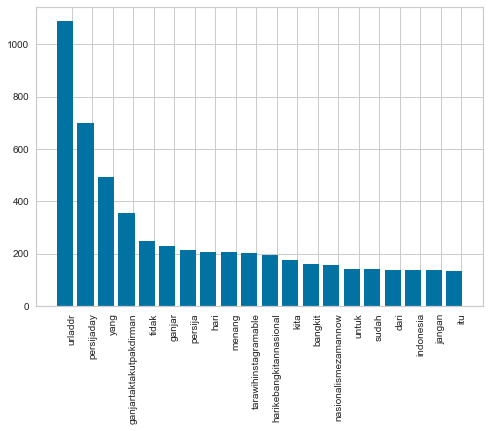
Model ini mengubah teks yang tidak terstruktur menjadi representasi vektor yang bisa dipelajari oleh sistem klasifikasi dengan menghitung kemunculan tiap kata. Langkah awal adalah dengan membuat kosakata dari kata-kata yang muncul pada training set, dan setiap kata dihitung kemunculannya[[25]](#footnote-25).

Algoritma 5.1 Penghitungan bag-of-words

1. split teks perkata
2. hitung frekuensi kemunculan kata pada spam
3. hitung frekuensi kemunculan kata pada ham

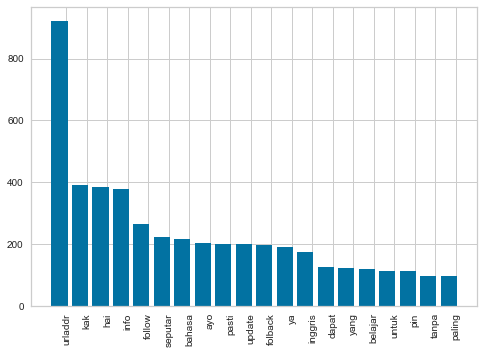
Segmen Program 5.1 Penghitungan bag-of-words

1. for i in range(noOfMessages):
2. processed = process\_text(self.tweets[i])
3. count = list() #
4. for word in processed:
5. if labels[i]: # membedakan kelas : 1 = spam, 0 = ham
6. tf\_spam[word] = self.tf\_spam.get(word, 0) + 1
7. spam\_words += 1
8. else:
9. tf\_ham[word] = self.tf\_ham.get(word, 0) + 1
10. ham\_words += 1
11. if word not in count:
12. count += [word]



Gambar 5.1

Distribusi kata pada ham



Gambar 5.2

Distribusi kata pada spam

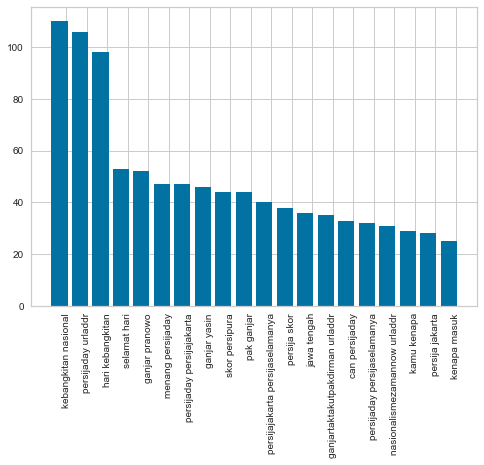
1. Bigram

*Bag-of-words* hanya menghitung jumlah kemunculan kata tanpa mempertimbangkan arti yang mungkin timbul dari urutan kemunculannya. Agar kemunculan frasa juga bisa diperhatikan maka digunakan n-gram model. Sebuah n-gram bisa dianggap sebagai sebuah frasa yang terdiri dar n kata. Sebagai contoh, pada kalimat “ Jangan mencari masalah bro” didapatkan bag-of-words atau 1-grams: “Jangan”, “mencari”, “masalah”, dan “bro”. 2-gram atau bigram dari kalimat tadi adalah: “Jangan mencari”, “mencari masalah”, “masalah bro”.

Segmen Program 5.2 Menghitung bigram

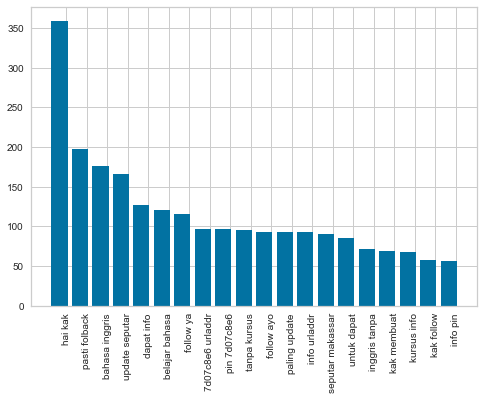
1. words = regTokenize(texts)
2. if gram > 1:
3. word = []
4. for i in range(len(words) - gram + 1):
5. word += [' '.join(words[i:i + gram])]
6. count = list() #
7. for word in processed:
8. if labels[i]: # membedakan kelas : 1 = spam, 0 = ham
9. tf\_spam[word] = self.tf\_spam.get(word, 0) + 1
10. spam\_words += 1
11. else:
12. tf\_ham[word] = self.tf\_ham.get(word, 0) + 1
13. ham\_words += 1
14. if word not in count:
15. count += [word]

Untuk membuat fitur bigram, kalimat dipisahkan perkata, kemudian setiap kata digabung dengan kata berikutnya, hingga akhir kalimat. Kemudian masing-masing bigram dihitung frekuensi kemunculannya pada daftar bigram di spam maupun ham.



Gambar 5.3

Distribusi bigram pada ham



Gambar 5.4

Distribusi 20 frasa tertinggi pada spam

1. Term frequency dan inverse document frequency (tf-idf).

Unigram atau bigram saja tidak memperhitungkan signifikansi dari suatu istilah di dalam dokumen. Dalam hal ini baik suatu dokumen berukuran besar atau kecil, akan dianggap sama saja. Di samping itu ada beberapa kata yang hampir dipastikan muncul di semua dokumen, misalnya kata “yang”. Walaupun sering muncul, tapi kata ini tidak penting. Segmen Program 5.4 menunjukkan proses penghitungan inverse document frequency ini.

Segmen Program 5.3 Menghitung term frequency

1. noOfMessages = self.tweets.shape[0]
2. for i in range(noOfMessages): processed = prosestext(self.tweets[i], stem=self.stem, gram=self.gram)
3. count = list() #
4. for word in processed:
5. if self.labels[i]:
6. self.tf\_spam[word] = self.tf\_spam.get(word, 0)+1
7. self.spam\_words += 1
8. else:
9. self.tf\_ham[word] = self.tf\_ham.get(word, 0) + 1
10. self.ham\_words += 1
11. if word not in count:
12. count += [word]

Segmen Program 5.4 Menghitung inverse document frequency

1. for word in count:
2. if self.labels[i]:
3. self.idf\_spam[word] = self.idf\_spam.get(word,0)+1
4. else:
5. self.idf\_ham[word] = self.idf\_ham.get(word,0) + 1
6. Training

Pada fase training dihitung prior dan Conditional Probability. Pertama dilakukan ekstraksi seluruh kosakata dari keseluruhan dokumen. Prior probability dari setiap kelas, dalam hal ini adalah spam dan ham, dihitung dari jumlah keseluruhan dokumen di dalam kelas dibagi jumlah dokumen keseluruhan. Conditional probability dihitung dari frekuensi kemunculan kata atau istilah dibagi jumlah keseluruhan kata per kelas ditambah jumlah keseluruhan istilah yang terdapat dalam training data. Algoritma 5.2 menunjukkan proses training dengan multinomial Naive Bayes[[26]](#footnote-26). Untuk menghindari nilai nol pada conditional probability digunakan teknik smoothing. Pada thesis ini digunakan Laplace-Smoothing.

Algoritma 5.2 Training multinomial Naive Bayes

1. V ← EkstrakKosakata(D)
2. N ← HitungDokumen(D)
3. for each c∈C
4. do Nc ← HitungDokumenDalamKelas(D,c)
5. prior[c] ← Nc/N
6. textc ← GabungkanTeksSemuaDokumenDiKelas(D,c)
7. for each t∈V
8. do Tct ← HitungTokenTerm(textc,t)
9. for each t∈V
10. do conditionalprob[t][c]
11. return V,prior,conditionalprob

Algoritma 5.3 Klasifikasi dengan Multinomial bayes

1. W ← EkstrakTokenDariDokumen(V,d)
2. for each c∈C
3. do score[c] ← log prior[c]
4. for each t ∈ W
5. do score[c] += log conditionalprop[t][c]
6. return argmaxc∈Cscore[c]

Sedangkan untuk training dengan metode tf-idf ditunjukkan pada Segmen Program 5.3, Segmen Program 5.4, dan Segmen Program 5.6

Untuk demonstrasi proses training dan klasifikasi, digunakan 10 % twit dari 4459 twit dataset sebagai data training. Proses training sebagaimana pada Algoritma 4.1 untuk fitur unigram adalah sebagai berikut:

1. Data Training

Total data training adalah 4019 twit, terdiri dari 2729 ham dan 1290 spam. Sehingga bisa dihitung:

* Prior probabilitas untuk kelas spam:

P(spam) = = = 0,32

* Prior probabilitas untuk kelas ham:

P(ham) = = = 0,68

1. Penghitungan frekuensi dan probabilitas setiap kata

Langkah selanjutnya adalah menghitung frekuensi kemunculan setiap kata pada ham maupun spam dan probabilitas kondisionalnya atau likelihood yang menjadi fitur dari setiap kata. Penghitungan dilakukan dengan: untuk setiap kemunculan kata w dari dokumen d dihitung kemunculan w di dalam kelas spam atau ham. Lalu nilainya dibagi dengan semua kata dalam dokumen d yang termasuk ke dalam kelas spam atau ham. Untuk menghindari probabilitas nol pada kata yang tidak muncul *digunakan Laplace Smoothing*. Laplace smoothing dilakukan dengan penambahan nilai 1.

(5.1)

Dengan V adalah jumlah kosakata keseluruhan. Sebagai contoh, probabilitas kata “bangkit” dan “menang” pada ham dan “kak” dan “info” dengan total kosakata ham sebanyak 18146 dan total kosakata spam 12056 menggunakan laplace smoothing adalah sebagai berikut :

= 0,0175

Tabel 5.1

Contoh frekuensi Unigram

|  |  |  |  |
| --- | --- | --- | --- |
| Ham (Non-Spam) | | Spam | |
| kata | Jumlah kemunculan | kata | Jumlah kemunculan |
| bangkit | 395 | kak | 466 |
| menang | 311 | info | 436 |
| persija | 256 | follow | 310 |
| ganjar | 240 | urladdr | 304 |
| jkt | 221 | bahasa | 265 |
| indonesia | 174 | putar | 254 |
| ayo | 150 | ayo | 239 |
| nasional | 138 | folback | 225 |
| selamat | 132 | inggris | 209 |
| main | 129 | update | 188 |
| bangsa | 117 | ajar | 154 |
| tarawih | 113 | main | 135 |
| persipura | 110 | makassar | 107 |
| urladdr | 100 | kursus | 104 |
| skor | 99 | telp | 102 |
| can | 98 | pin7d07c8e6 | 91 |
| persijaselamanya | 97 | keren | 85 |
| semangat | 89 | game | 72 |
| imam | 80 | online | 70 |
| alhamdulillah | 79 | persibday | 69 |

Tabel 5.2

Contoh Nilai Probailitas

|  |  |  |  |
| --- | --- | --- | --- |
| Ham (Non-Spam) | | Spam | |
| kata | Probabilitas | kata | Probabilitas |
| bangkit | 0,0175 | kak | 0,0330 |
| menang | 0,0138 | info | 0,0309 |
| persija | 0,0114 | follow | 0,0220 |
| ganjar | 0,0107 | urladdr | 0,0216 |
| jkt | 0,0098 | bahasa | 0,0188 |
| indonesia | 0,0077 | putar | 0,0180 |
| ayo | 0,0067 | ayo | 0,0170 |
| nasional | 0,0061 | folback | 0,0160 |
| selamat | 0,0059 | inggris | 0,0148 |
| bangsa | 0,0058 | update | 0,0134 |
| urladdr | 0,0052 | ajar | 0,0110 |
| main | 0,0050 | main | 0,0096 |
| tarawih | 0,0049 | makassar | 0,0076 |
| skor | 0,0045 | kursus | 0,0074 |
| persipura | 0,0044 | telp | 0,0073 |
| can | 0,0044 | pin7d07c8e6 | 0,0065 |
| persijaselamanya | 0,0043 | keren | 0,0061 |
| semangat | 0,0040 | game | 0,0052 |
| pimpin | 0,0036 | online | 0,0050 |
| alhamdulillah | 0,0035 | persibday | 0,0049 |

Segmen Program 5.5 Penghitungan probabilitas pada Naive Bayes classifier

1. for word in self.tf\_spam:
2. prob\_spam[word] = (tf\_spam[word] + 1) /(spam\_words + len(list(tf\_spam.keys())))
3. for word in self.tf\_ham:
4. prob\_ham[word] = (tf\_ham[word] + 1) / (ham\_words + len(list(tf\_ham.keys())))
5. prob\_spam\_twit = spam\_tweets/total\_tweets
6. prob\_ham\_twit = ham\_tweets / total\_tweets

Nilai tf-idf dihitung dengan rumus berikut, dengan TF(w) adalah *term frequency* dari kata w, dan IDF(w) adalah *inverse document frequency* dari kata w:

(5.2)

Dengan menggunakan Laplace Smoothing menjadi

(5.3)

Sedangkan nilai *inverse document frequency* IDF(w) dihitung sebagai nilai logaritmik keseluruhan dokumen dibagi dengan dokumen yang berisi kata w sebagai berikut:

(5.4)

Tabel 5.3

Contoh probabilitas dengan tf-idf

|  |  |  |  |
| --- | --- | --- | --- |
| Ham (Non-Spam) | | Spam | |
| kata | Jumlah kemunculan | kata | Jumlah kemunculan |
| bangkit | 0,000204 | kak | 0,000892 |
| menang | 0,000150 | info | 0,001021 |
| persija | 0,000114 | follow | 0,000861 |
| ganjar | 0,000272 | urladdr | 0,000279 |
| jkt | 0,000202 | bahasa | 0,000619 |
| indonesia | 0,000169 | putar | 0,000752 |
| ayo | 0,000127 | ayo | 0,000305 |
| nasional | 0,000146 | folback | 0,000696 |
| selamat | 0,000144 | inggris | 0,000543 |
| bangsa | 0,000121 | update | 0,000919 |
| urladdr | 0,000117 | ajar | 0,000900 |
| main | 0,000124 | main | 0,000285 |
| tarawih | 0,000177 | makassar | 0,000768 |
| skor | 0,000226 | kursus | 0,000400 |
| persipura | 0,000225 | telp | 0,000484 |
| can | 0,000201 | pin7d07c8e6 | 0,000799 |
| persijaselamanya | 0,000200 | keren | 0,000557 |
| semangat | 0,000267 | game | 0,000687 |
| pimpin | 0,000598 | online | 0,000472 |
| alhamdulillah | 0,000184 | persibday | 0,000259 |

Segmen Program 5.6 Penghitungan probabilitas Naive Bayes dengan tf-idf

1. for word in tf\_spam:
2. prob\_spam[word] = (tf\_spam[word])\* log(total\_tweets/ (idf\_spam[word] + idf\_ham.get(word, 0)))
3. sum\_tf\_idf\_spam += prob\_spam[word]
4. for word in tf\_spam:
5. prob\_spam[word] = (prob\_spam[word] + 1) / (sum\_tf\_idf\_spam + len(list(prob\_spam.keys())))
6. for word in tf\_ham:
7. prob\_ham[word] = (tf\_ham[word]) \* log((spam\_tweets + ham\_tweets) / (idf\_spam.get(word, 0) + idf\_ham[word]))
8. sum\_tf\_idf\_ham += prob\_ham[word]
9. for word in tf\_ham:
10. prob\_ham[word] = (prob\_ham[word] + 1) / (sum\_tf\_idf\_ham + len(list(prob\_ham.keys())))
11. prob\_spam\_twit, prob\_ham\_twit = spam\_tweets/ total\_tweets, ham\_tweets / total\_tweets
12. Proses Klasifikasi

Contoh klasifikasi untuk twit berbunyi “Ciptakan momen paling indah malam ini. LO HARUS MENANG @Persija\_Jkt !! #PersijaDay” :

* Pembersihan data dan casefolding: “ciptakan momen paling indah malam lo harus menang jkt persijaday”
* Stemming : “cipta momen paling indah malam lo harus menang jkt persijaday”
* Filtering: “cipta momen indah malam lo menang jkt”
* Perhitungan skor probabilitas:
  + P(twit|ham) = P(cipta|ham)\*P(momen|ham)\*P(indah|ham)\*P(malam|ham)\*P(lo|ham)\*P(menang|ham)\*P(jkt|ham)\*(P(ham))7

=

=

* + P(twit|spam) = P(cipta|spam)\*P(momen|spam)\*P(indah|spam)\*P(malam|spam)\*P(lo|spam)\*P(menang|spam)\*P(jkt|spam)\*(P(spam))7

=

=

* + P(twit|ham) > P(twit|spam) maka twit termasuk ham. Pada perhitungan P(twit|spam), kata-kata “cipta”, “momen”, “indah”, dan “jkt” tidak muncul dalam data training spam, maka nilainya menggunakan nilai smoothing

Segmen Program 5.7 Fungsi Klasifikasi

1. def classify(self, processed\_text):
2. pSpam, pHam = 0, 0
3. for word in processed\_text:
4. if word in prob\_spam:
5. pSpam += log(prob\_spam[word])
6. else:
7. if method == 'tfidf':
8. pSpam -= log(sum\_tf\_idf\_spam \
9. + len(list(prob\_spam.keys())))
10. else:
11. pSpam -= log(spam\_words \
12. + len(list(prob\_spam.keys())))
13. if word in prob\_ham:
14. pHam += log(prob\_ham[word])
15. else:
16. if method == 'tfidf':
17. pHam -= log(sum\_tf\_idf\_ham \
18. + len(list(prob\_ham.keys())))
19. else:
20. pHam -= log(ham\_words
21. + len(list(prob\_ham.keys())))
22. pSpam += log(prob\_spam\_twit)
23. pHam += log(prob\_ham\_twit)
24. return pSpam >= pHa

Contoh perhitungan dengan klasifikasi menggunakan fitur yang diambil dari model bigram adalah sebagai berikut:

* Twit: “Ciptakan momen paling indah malam ini. LO HARUS MENANG @Persija\_Jkt !! #PersijaDay”
* Pembersihan data dan casefolding: “ciptakan momen paling indah malam lo harus menang jkt persijaday”
* Stemming : “cipta momen paling indah malam lo harus menang jkt persijaday”
* Filtering: “cipta momen indah malam lo menang jkt”
* Bigram: “cipta momen”, “momen indah”, “indah malam”, “malam lo”, “lo menang”, “menang jkt”
* P(twit|ham) = P(“cipta momen”|ham)\*P(“momen indah”|ham)\*P(“indah malam”|ham)\*P( “malam lo”|ham)\*P( “lo menang”|ham)\*P( “menang jkt”|ham)\*(P(ham))6

Karena semua frasa tidak terdapat dalam data training maka dimasukkan nilai smoothing

Maka P(twit|ham) = =

* P(twit|spam) = P(“cipta momen”|spam)\*P(“momen indah”|spam)\*P(“indah malam”|spam)\*P( “malam lo”|spam)\*P( “lo menang”|spam)\*P( “menang jkt”|spam) \*(P(spam))6

Sebagaimana papa perhitungan skor ham, semua frasa tidak terdapat dalam data training spam maka dimasukkan nilai smoothing

Maka P(twit|ham) = =

Dari perhitungan di atas, P(twit|ham) > P(twit|spam) sehingga twit masuk ke dalam ham

1. Smoothing

Pada penelitian ini dilakukan ujicoba peggunaan smoothing dengan menggunakan *stupid backoff*[[27]](#footnote-27). Penghitungan *Smoothing* *stupid backoff* dengan melakukan pencarian berdasar gram yang tinggi, dan apabila tidak ditemukan, maka nilainya dicari pada gram yang lebih rendah dengan perkalian faktor *backoff* α. Di sini dimulai dari bigram, dengan nilai α = 0,4.

Segmen Program 5.8 Klasifikasi dengan smoothing stupid backoff

1. hamscore = 0.0
2. spamscore = 0.0
3. for word in processed\_text:
4. wordtoken = word.split()
5. tokenprev = wordtoken[0]
6. tokennext = wordtoken[1]
7. if word in tf\_ham2:
8. bicount = tf\_ham2[word]
9. bi\_unicount = tf\_ham1[tokenprev]
10. hamscore += math.log(bicount)
11. hamscore -= math.log(bi\_unicount)
12. else:
13. if tokennext in tf\_ham1:
14. unicount = tf\_ham1[tokennext]
15. else:
16. unicount = 0
17. hamscore += math.log(0.4)
18. hamscore += math.log(unicount + 1)
19. hamscore -= log(ham\_words + vocabsize)
21. for word in processed\_text:
22. wordtoken = word.split()
23. tokenprev = wordtoken[0]
24. tokennext = wordtoken[1]
25. if word in tf\_spam2:
26. bicount = tf\_spam2[word]
27. bi\_unicount = tf\_spam1[tokenprev]
28. spamscore += log(bicount)
29. spamscore -= log(bi\_unicount)
30. else:
31. if tokennext in tf\_spam1:
32. unicount = tf\_spam1[tokennext]
33. else:
34. unicount = 0
35. spamscore += math.log(0.4)
36. spamscore += math.log(unicount + 1)
37. spamscore -= log(spam\_words + vocabsize))
38. return spamscore >= hamscore

Contoh klasifikasi twit dengan menggunakan stupid backoff:

* Twit: “Ciptakan momen paling indah malam ini. LO HARUS MENANG @Persija\_Jkt !! #PersijaDay”
* Pembersihan data dan casefolding: “ciptakan momen paling indah malam lo harus menang jkt persijaday”
* Stemming : “cipta momen paling indah malam lo harus menang jkt persijaday”
* Filtering: “cipta momen indah malam lo menang jkt”
* Bigram: “cipta momen”, “momen indah”, “indah malam”, “malam lo”, “lo menang”, “menang jkt”
* Unigram = “cipta”, “momen”, “indah”, “malam”, “lo”, “menang”, “jkt”
  + P(twit|ham) = dimulai dari bigram, karena semua frase tidak ada di training, maka dihitung dengan unigram dengan backoff faktor 0,4
  + P(twit|ham)= (0,4)7\* P(cipta|ham)\*P(momen|ham)\*P(indah|ham)\*P(malam|ham)\*P(lo|ham)\*P(menang|ham)\*P(jkt|ham)\*(P(ham))7

=

=

* + P(twit|spam) = dimulai dari bigram, karena semua frase tidak ada di training, maka dihitung dengan unigram dengan backoff faktor 0,4
  + P(twit|spam)= (0,4)7\* P(cipta|spam)\*P(momen|spam)\*P(indah|spam)\*P(malam|spam)\*P(lo|spam)\*P(menang|spam)\*P(jkt|spam)\*(P(spam))7

=

=

* + P(twit|ham) > P(twit|spam) maka twit tergolong ham

# BAB VI

UJICOBA

Pada bab ini dibahas tentang pengujian sistem klasifikasi yang telah dibangun. Pengujian dilakukan untuk mengetahui performa dari sistem klasifikasi. Hendak diketahui juga pengaruh pilihan fitur berbeda terhadap performa klasifikasi. Di samping itu juga akan dilihat perbandingan antara klasifikasi dengan stupid backoff smoothing terhadap klasifikasi dengan Laplace Smoothing.

1. Cross-Validation

Untuk memvalidasi Sistem klasifikasi Naive Bayes yang telah dibangun terhadap data yang belum pernah dilihat, dilakukan pembagian dataset menjadi training dan dataset. Pembagian yang sederhana adalah dengan membagi langsung menjadi dua bagian dengan rasio tertentu. Pengacakan pada data sebelum pembagian menjamin distribusi yang seimbang pada kedua bagian. Bagian yang besar dijadikan training set, dan bagian yang kecil sebagai testing set. Kekurangan metode ini adalah sistem klasifikasi tidak dilatih dan divalidasi pada keseluruhan contoh di dataset.

Agar bisa dilakukan pelatihan dan validasi pada semua bagian digunakan metode k-fold cross-validation. Dengan metode ini data set dibagi menjadi k bagian. Dilakukan pelatihan dan validasi secara berulang, dengan setiap perulangan menjadikan satu bagian sebagai testing set sedangkan bagian yang lain menjadi training set. Testing set dirotasi pada setiap perulangan. Skor pada setiap perulangan dijumlah kemudian diambil rata-ratanya.

Pada penelitian ini digunakan validasi silang sebanyak 10 kali ujicoba klasifikasi, atau biasa disebut 10-fold cross validation. Jumlah data sebanyak 4459 yang terdiri dari 3022 twit ham dan 1437 twit spam. Untuk ujicoba, data ini dibagi dalam 10 bagian. Pada setiap satu putaran ujicoba klasifikasi, 10 % digunakan sebagai data testing dan sisanya sebagai data training. Pada pengujian berikutnya, 10 % data testing diambil dari bagian berikutnya. Demikian seterusnya, sehingga pengujian dilakukan pada keseluruhan data. Hasi akhir merupakan rata-rata dari keseluruhan putaran ujicoba klasifikasi. Sebelum pembagian dalam data training dan testing dilakukan pengacakan terlebih dahulu terhadap dataset, agar didapatkan distribusi yang merata.

Segmen Program 6.1 Evaluasi Sistem klasifikasi Bayes

1. from sklearn.model\_selection import KFold
2. kf = KFold(n\_splits=10)
3. print(kf)
4. for train\_index, test\_index in kf.split(tweets,labels):
5. print('\n== 1. Unigram/Bag-of\_Words ==')
6. clf = SpamClassifier(tweets[train\_index],labels[train\_index],method='bow',gram=1)
7. clf.train()
8. prediksi = clf.predict(tweets[test\_index])
9. metrics(labels[test\_index],prediksi)
11. print('\n== 2. Unigram/Bag-of\_Words with tfidf ==')
12. clf = SpamClassifier(tweets[train\_index],labels[train\_index],method='tfidf',gram=1)
13. clf.train()
14. prediksi = clf.predict(tweets[test\_index])
15. metrics(labels[test\_index],prediksi)
17. print('\n== 3. Bigram ==')
18. clf = SpamClassifier(tweets[train\_index],labels[train\_index],method='bow',gram=2)
19. clf.train()
20. prediksi = clf.predict(tweets[test\_index])
21. metrics(labels[test\_index],prediksi)
23. print('\n== 4. Bigram with tfidf ==')
24. clf = SpamClassifier(tweets[train\_index],labels[train\_index],method='tfidf',gram=2)
25. clf.train()
26. prediksi = clf.predict(tweets[test\_index])
27. metrics(labels[test\_index],prediksi)
29. print('\n== 5. Stupid Backoff ==')
30. clf = SpamClassifier(tweets[train\_index],labels[train\_index],method='stbo',gram=2)
31. clf.train()
32. prediksi = clf.predict(tweets[test\_index])
33. metrics(labels[test\_index],prediksi)

Pada penelitian ini dilakukan perbandingan untuk klasifikasi dengan 5 variasi:

* 1. Fitur berupa bag-of-words atau unigram tanpa mempertimbangkan term frequency dan inverse document frequency.
  2. Fitur berupa bag-of-words dengan term frequency dan inverse document frequency
  3. Fitur bigram tanpa term frequency dan inverse document frequency
  4. Fitur bigram dengan term frequency dan inverse document frequency
  5. Fitur bigram dengan smoothing stupid backoff

Tabel 6.1

Pembagian Data Training dan Data Testing

|  |  |  |
| --- | --- | --- |
| No Fold | Index Data Training | Index Data testing |
| 1 | 456 - 4458 | 0 - 445 |
| 2 | 0 – 445 dan 892 - 4458 | 445 - 891 |
| 3 | 0 – 891 dan 1338 - 4458 | 892 - 1337 |
| 4 | 0 – 1337 dan 1784 - 4458 | 1338 - 1783 |
| 5 | 0 – 1783 dan 2230 - 4458 | 1784 - 2229 |
| 6 | 0 – 2229 dan 2676 - 4458 | 2230 - 2675 |
| 7 | 0 – 2675 dan 3122 - 4458 | 2676 - 3121 |
| 8 | 0 – 3121 dan 3568 - 4458 | 3122 - 3567 |
| 9 | 0 – 3567 dan 4014 - 4458 | 3568 - 4013 |
| 10 | 0 – 4013 | 4014 - 4458 |

1. Ujicoba Klasifikasi Pertama

Pada ujicoba klasifikasi pertama, data training adalah data dengan indeks 456 s.d. 4458, sedangkan data testing adalah data dengan indeks nomer 0 s.d. 445.

Tabel 6.2

Kondisi Data Ujicoba Pertama

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1298 | 2715 |
| Testing | 139 | 307 |

Hasil ujicoba pertama bisa dilihat pada tabel berikut:

Tabel 6.3

Hasil Ujicoba klasifikasi pertama

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | klasifikasi | True spam | False spam | True ham | False Ham | F1 score | Akurasi |
| unigram | MultinomialNB laplace smoothing | 123 | 0 | 307 | 16 | 0.939 | 0.964 |
| Unigram TF-IDF | MultinomialNB laplace smoothing | 130 | 0 | 307 | 9 | 0.967 | 0.980 |
| Bigram | MultinomialNB laplace smoothing | 139 | 37 | 270 | 0 | 0.883 | 0.917 |
| Bigram TF-IDF | MultinomialNB laplace smoothing | 138 | 16 | 291 | 1 | 0.942 | 0.962 |
| Bigram & unigram | MultinomialNB Stupid Backoff smoothing | 135 | 1 | 306 | 4 | 0.982 | 0.989 |

Pada ujicoba pertama ini, F1-score tertinggi ada pada klasifikasi menggunakan smoothing stupid backoff, tingkat pengenalan terhadap spam cukup tinggi, tapi masih ada twit bukan spam yang digolongkan sebagai spam. Klasifikasi dengan fitur bigram menggunakan laplace smoothing mempunyai tingkat pengenalan spam tertinggi, akan tetapi juga terlalu agresiv sehingga tinggi juga tingkat penggolongan ham sebagai spam. Klasifikasi dengan fitur unigram, tidak ada kekeliruan memasukkan twit ham sebagai spam. Tapi paling rendah dalam mengenali spam.

1. Ujicoba klasifikasi kedua

Ujicoba kedua menggunakan data dengan indeks nomer 446 hingga 891 sebagai data testing, sedangkan sisanya sebagai data training.

Tabel 6.4

Data training dan testing pada ujicoba kedua

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1303 | 2710 |
| Testing | 134 | 312 |

Sebaran spam dan ham tidak jauh beda dengan sebaran spam dan ham pada ujicoba pertama. Hal ini menunjukkan pengacakan data cukup berhasil.

Tabel 6.5

Hasil klasifikasi pada ujicoba kedua

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Klasifikasi | True Spam | False Spam | True Ham | False Ham | F1 Score | Akurasi |
| Unigram | Multinomialnb Laplace Smoothing | 115 | 0 | 312 | 19 | 0.924 | 0.957 |
| Unigram TF-IDF | Multinomialnb Laplace Smoothing | 118 | 0 | 312 | 16 | 0.937 | 0.964 |
| Bigram | Multinomialnb Laplace Smoothing | 131 | 50 | 262 | 3 | 0.832 | 0.881 |
| Bigram TF-IDF | Multinomialnb Laplace Smoothing | 131 | 31 | 281 | 3 | 0.885 | 0.924 |
| Bigram  & Unigram | Multinomialnb Stupid Backoff Smoothing | 129 | 1 | 311 | 5 | 0.977 | 0.987 |

Unigram tanpa tf dan idf mempunyai tingkat recall yang paling rendah, yaitu kemampuan deteksi spam yang paling rendah, dan masih banyak spam yang digolongkan sebagai ham. Penggunaan tf dan idf meningkatkan sedikit pengenalan terhadap spamnya. Bigram mempunyai tingkat recall tertinggi. baik menggunakan metode tf-idf maupun tidak, tingkat deteksi spamnya cukup tinggi. Sayang sekali banyak pula twit ham yang digolongkan sebagai ham, sehingga f1 scorenya menjadi terendah di antara metode lain yang diujicobakan. Klasifikasi dengan smoothing stupid backoff mempunyai tingkat pengenalan spam yang cukup tinggi dan di pihak lain penggolongan ham sebagai spam jauh lebih rendah dibandingkan dengan penggunaan bigram saja sebagai fitur.

1. Ujicoba Klasifikasi Ketiga

Untuk ujicoba ketiga data testing digeser ke indeks nomer 892 hingga 1337. Sebaran twit spam pada data testing agak lebih tinggi dibandingkan pada data training yang sebelumnya, yaitu 152 spam, dengan total ham sejumlah 294. Di pihak lain pada data training sebaran spamnya lebih rendah dibandingkan dengan dua ujicoba sebelumnya, dengan total spam sebanyak 1285 twit dan twit ham berjumlah 2728.

Tabel 6.6

Data training dan testing pada ujicoba ketiga

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1285 | 2728 |
| Testing | 152 | 294 |

Hasil ujicoba ketiga bisa dilihat pada tabel 6.6. Walaupun ada sedikit perbedaan kondisi pada data training dan data testing, tapi kecenderungan hasil masih bisa dikatakan sama. Pilihan fitur unigram baik dengan tf-idf maupun tidak mempunyai tingkat pengenalan spam terendah. hal ini berarti juga lebih banyak spam yang tergolong ke ham, di pihak lain tidak ada twit ham yang terdeteksi sebagai ham. Spam filter menggunakan bigram, baik dengan tf-idf maupun tidak, bisa mendeteksi lebih banyak spam. Akan tetapi kekeliruan menggolongkan ham sebagai spam juga jauh lebih tinggi dibandingkan denga spam filter menggunakan unigram. Penggunaan smoothing stupid backoff menunjukkan perbaikan yang signifikan terhadap spam filter menggunakan bigram dengan laplace smoothing. Dari 151 spam yang ada pada data testing, hanya satu spam yang tidak berhasil dideteksi. Di pihak lain tidak ada twit ham yang digolongkan sebagai spam.

Tabel 6.7

Hasil ujicoba klasifikasi ketiga

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Klasifikasi | True Spam | False Spam | True Ham | False Ham | F1 Score | Akurasi |
| Unigram | Multinomialnb Laplace Smoothing | 138 | 0 | 294 | 14 | 0.952 | 0.969 |
| Unigram TF-IDF | Multinomialnb Laplace Smoothing | 142 | 0 | 294 | 10 | 0.966 | 0.978 |
| Bigram | Multinomialnb Laplace Smoothing | 151 | 49 | 245 | 1 | 0.858 | 0.888 |
| Bigram TF-IDF | Multinomialnb Laplace Smoothing | 150 | 27 | 267 | 2 | 0.912 | 0.935 |
| Bigram Unigram | Multinomialnb Stupid Backoff Smoothing | 151 | 0 | 294 | 1 | 0.997 | 0.998 |

1. Ujicoba Klasifikasi Keempat

Pada ujicoba keempat, data testing dimulai dari data indeks 1338 s.d. 1783 dan selainnya sebagai data training. Sebaran spam dan ham pada ujicoba ini adalah, total spam pada data testing berjumlah 135 twit dan total ham sebanyak 311 twit. Sedangkan untuk data training nya total spam sebanyak 1302 dan total ham berjumlah 2711.

Tabel 6.8

Data training dan data testing ujicoba keempat

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1302 | 2711 |
| Testing | 135 | 311 |

Tabel 6.9

Hasil ujicoba keempat

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Klasifikasi | True Spam | False Spam | True Ham | False Ham | F1 Score | Akurasi |
| Unigram | Multinomialnb Laplace Smoothing | 118 | 1 | 310 | 17 | 0.930 | 0.960 |
| Unigram TF-IDF | Multinomialnb Laplace Smoothing | 125 | 1 | 310 | 10 | 0.960 | 0.975 |
| Bigram | Multinomialnb Laplace Smoothing | 134 | 53 | 258 | 1 | 0.832 | 0.879 |
| Bigram TF-IDF | Multinomialnb Laplace Smoothing | 133 | 44 | 267 | 2 | 0.853 | 0.897 |
| Bigram  Unigram | Multinomialnb Stupid Backoff Smoothing | 133 | 8 | 303 | 2 | 0.964 | 0.978 |

Pada ujicoba keempat ini, kembali ditunjukkan bahwa fitur unigram dan Naive Bayes menggunakan laplace smoothing mempunyai tingkat pengenalan spam terendah. F1-scorenya masih tinggi yaitu 0.93. Penghitungan tf-idf sedikit meningkatkan kemampuannya untuk mendeteksi spam. Spam filter dengan pemodelan bahasa bigram mempunyai tingkat pengenalan spam yang tinggi, akan tetapi juga banyak memasukkan twit ham sebagai spam. F1-score pada spam filter dengan fitur berdasar model bahasa bigram lebih rendah dibandingkan dengan penggunaan bag-of-words, yaitu sekitar 0.8. Penghitungan term frequency dan inverse document frequency agak mengurangi tingkat kekeliruan menggolongkan twit ham sebagai spam. Klasifikasi dengan stupid backoff smoothing menunjukkan pengenalan spam yang tinggi dan tingkat kekliruan penggolongan ham sebagai spam yang jauh lebih rendah dibandingkan dengan penggunaan bigram saja sebagai fitur.

1. Ujicoba klasifikasi kelima

Untuk ujicoba kelima, data testing menggunakan data set dengan indeks dari 1784 dingga 2229, dengan sebaran spam dan ham adalah 139 twit sebagai spam dan 307 twit adalah ham. Sedangakan jumlah spam dan ham pada data training meliputi 1298 twit spam dan 2715 twit ham.

Tabel 6.10

Data training dan data testing ujicoba kelima

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1298 | 2715 |
| Testing | 139 | 307 |

Hasil ujicoba kelima bisa dilihat pada Tabel 6.11

Tabel 6.11

Hasil ujicoba kelima

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Klasifikasi | True Spam | False Spam | True Ham | False Ham | F1 Score | Akurasi |
| Unigram | Multinomialnb Laplace Smoothing | 116 | 0 | 307 | 23 | 0.910 | 0.949 |
| Unigram TF-IDF | Multinomialnb Laplace Smoothing | 121 | 0 | 307 | 18 | 0.960 | 0.975 |
| Bigram | Multinomialnb Laplace Smoothing | 134 | 57 | 250 | 5 | 0.812 | 0.860 |
| Bigram TF-IDF | Multinomialnb Laplace Smoothing | 133 | 35 | 272 | 6 | 0.867 | 0.910 |
| Bigram  Unigram | Multinomialnb Stupid Backoff Smoothing | 131 | 1 | 306 | 8 | 0.967 | 0.980 |

Hasil ujicoba kelima tampak bahwa spam filter dengan penggunaan fitur unigram mendeteksi spam sebanyak 116 twit spam dari total spam sebanyak 139 Sedangkan keseluruhan twit ham bisa dikenali sebagai ham. Penghitungan term frequency dan inverse document frequency sedikit meningkatkan pengenalan terhadap spam. Klasifikasi dengan fitur bigram bisa mendeteksi hingga 96% spam yang ada, akan tetapi tingkat kesalahan penggolongan ham menjadi spam juga tinggi, yaitu hingga mencapai 18% dari total ham yang ada. Penghitungan term frequency dan inverse document frequency bisa mengurangi tingkat kesalahan deteksi ham sebagai spam sebanyak 6% menjadi 11%. F1-score pada metode ini tetap di bawah 0.9. Klasifikasi bayes dengan smoothing stupid backoff bisa mengenali spam hingga 94% dari total keseluruhan spam yang diperiksa. Sedangkan tingkat kesalahan penggolongan twit ham sebagai spam sangat rendah, yaitu 0.3% dari keseluruhan twit ham yang diperiksa.

1. Ujicoba klasifikasi Keenam

Pada ujicoba keenam, yang digunakan sebagai data testing adalah data set dengan indeks 2230 hingga 2674. Pada 446 data tersebut terdapat 145 spam dan 301 ham. Sedangkan pada data training terdapat 2721 ham dan 1292 spam.

Tabel 6.12

Data training dan data testing pada ujicoba keenam

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1292 | 2721 |
| Testing | 145 | 301 |

Hasil ujicoba keenam bisa dilihat pada Tabel 6.13. Dari 145 spam yang diujikan, 128 bisa dideteksi oleh klasifikasi dengan fitur unigram. hanya 1 dari 301 twit ham yang salah dideteksi sebagai spam. Penghitungan term dan inverse document frequency sedikit meningkatkan pengenalan terhadap spam. Klasifikasi berdasar fitur bigram bisa mengenali 143 dari 145 spam yang diuji. Akan tetapi di pihak lain jumlah twit ham yang salah digolongkan ke dalam spam juga tinggi, hingga mencapai 46 twit dari total 301 twit ham yang diuji. Penghitungan term dan inverse document frequency sebagai tambahan fitur untuk bigram bisa menurunkan tingkat kesalahan penganalan ham sebagai spam sebanyak 34%. Klasifikasi dengan stpid backoff smoothing sedikit di bawah bigram dengan lapace smoothing, yaitu bisa mendeteksi 139 spam dari 145 spam yang diperiksa. Akan tetapi di pihak lain tingkat kesalahan penggolongan baik spam maupun ham cukup rendah.

Tabel 6.13

Hasil ujicoba keenam

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Klasifikasi | True Spam | False Spam | True Ham | False Ham | F1 Score | Akurasi |
| Unigram | Multinomialnb Laplace Smoothing | 128 | 1 | 300 | 17 | 0.952 | 0.969 |
| Unigram TF-IDF | Multinomialnb Laplace Smoothing | 131 | 2 | 299 | 14 | 0.942 | 0.941 |
| Bigram | Multinomialnb Laplace Smoothing | 143 | 46 | 255 | 2 | 0.858 | 0.892 |
| Bigram TF-IDF | Multinomialnb Laplace Smoothing | 142 | 30 | 271 | 3 | 0.896 | 0.926 |
| Bigram  Unigram | Multinomialnb Stupid Backoff Smoothing | 139 | 4 | 297 | 6 | 0.997 | 0.998 |

1. Ujicoba klasifikasi Ketujuh

Putaran pengujian ketujuh menggunakan data indeks nomer 2676 hingga 3121 sebagai data testing dan sisanya sebagai data training. Komposisi data training terdiri dari 1288 data spam dan 2725 ham. Sedangkan data testing terdiri dari 149 twit spam dan 297 twit ham. Sebaran spam dan ham pada data training maupun data testing ini tidak begitu beda dengan sebaran spam dan ham pada data training dan testing dari ujicoba sebelumnya.

Tabel 6.14

Data training dan data testing pada ujicoba ketujuh

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1288 | 2725 |
| Testing | 149 | 297 |

Hasil ujicoba ketujuh bisa dilihat pada Tabel 6.15. Klasifikasi dengan unigram/bag-of-words bisa mendeteksi 135 dari total 149 spam di data testing. Tidak ada kesalahan penggolongan twit ham sebagai spam. Penambahan perhitungan term dan inverse document frequency menambah jumlah spam yang dideteksi menjadi 138. Klasifikasi berdasar model bigram bisa mendeteksi hingga 146 dari total 149 spam yang diujicobakan. Akan tetapi klasifikasi dengan metode ini mempunyai tingkat kesalahan deteksi ham menjadi spam yang cukup tinggi yaitu hingga 13 % dari jumlah total ham yang diperiksa. Penambahan term frequency dan inverse document frequency menurunkan tingkat kesalahan penggolongan ham hingga separuhnya. Penggunaan stupid backoff smoothing dalam penghitungan probabilitas mempunyai perimbangan deteksi dan kesalahan yang lebih baik. Klasifikasi dengan metode ini bisa mendeteksi 143 spam, dan hanya dua kalai melakukan kesalahan penggolongan ham sebagai spam.

Tabel 6.15

Hasil ujicoba ketujuh

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Klasifikasi | True Spam | False Spam | True Ham | False Ham | F1 Score | Akurasi |
| Unigram | Multinomialnb Laplace Smoothing | 135 | 0 | 297 | 14 | 0.950 | 0.969 |
| Unigram TF-IDF | Multinomialnb Laplace Smoothing | 138 | 0 | 297 | 11 | 0.962 | 0.975 |
| Bigram | Multinomialnb Laplace Smoothing | 146 | 40 | 257 | 3 | 0.871 | 0.904 |
| Bigram TF-IDF | Multinomialnb Laplace Smoothing | 143 | 20 | 277 | 6 | 0.917 | 0.942 |
| Bigram Stupid Unigram | Multinomialnb Stupid Backoff Smoothing | 143 | 2 | 295 | 6 | 0.973 | 0.982 |

1. Ujicoba klasifikasi Kedelapan

Data testing pada putaran ujicoba kedelapan ini dimulai dari indeks nomer 3122 sampai dengan 3567. Pada bagian tersebut terdapat 151 twit spam dan 295 twit ham. Komposisi untuk data trainingnya adalah terdiri dari 1286 twit spam dan 2727 twit bukan spam atau ham.

Tabel 6.16

Data training dan testing pada ujicoba kedelapan

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1286 | 2727 |
| Testing | 151 | 295 |

Hasil ujicoba kedelapan untuk pendeteksian spam bisa dilihat pada Tabel 6.17. Klasifikasi dengan fitur unigram/bag-of-words mendeteksi 130 twit spam dari total 151 spam yang diperiksa, sehingga ada 21 twit spam yang secara keliru dinyatakan sebagai ham. Di pihak lain, klasifikasi dengan metode ini hanya sekali mengalami kesalahan penggolongan ham sebagai spam, dari total 295 twit ham yang diperiksa. Perhitungan term frequency dan inverse document frequency pada klasifikasi ini meningkatkan deteksi spam menjadi 139 dari total spam yang diujikan. Klasifikasi dengan fitur bigram mendeteksi 149 spam dari keseluruhan twit spam yang diperiksa. Tingginya tingkat deteksi spam ini dibarengi pula dengan tingginya tingkat kesalahan penggolongan twit ham ke dalam kelompok spam, yaitu hingga 41 twit ham digolongkan sebagai ham. Klasifikasi dengan fitur bigram dan tf-idf mempunyai tingkat kesalahan yang lebih rendah, yaitu 25 twit ham dari total 295 twit dideteksi sebagai spam. Klasifikasi dengan smoothing stupid backoff bisa mendeteksi 148 spam dari 151 twit spam yang diperiksa, dengan 2 kali kesalahan penggolongan ham menjadi spam.

Tabel 6.17

Hasil klasifikasi pada ujicoba kedelapan

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Klasifikasi | True Spam | False Spam | True Ham | False Ham | F1 Score | Akurasi |
| Unigram | Multinomialnb Laplace Smoothing | 130 | 1 | 294 | 21 | 0.922 | 0.951 |
| Unigram TF-IDF | Multinomialnb Laplace Smoothing | 139 | 2 | 293 | 12 | 0.952 | 0.969 |
| Bigram | Multinomialnb Laplace Smoothing | 149 | 41 | 254 | 2 | 0.874 | 0.904 |
| Bigram TF-IDF | Multinomialnb Laplace Smoothing | 148 | 25 | 270 | 3 | 0.914 | 0.937 |
| Bigram  Unigram | Multinomialnb Stupid Backoff Smoothing | 148 | 2 | 293 | 3 | 0.983 | 0.989 |

1. Ujicoba Klasifikasi Kesembilan

Untuk ujicoba kesembilan digunakan bagian data nomer 3568 sampai dengan 4013 sebagai data testing. Komposisi data training adalah meliputi 1281 twit spam dan 2732 twit ham. Sedangkan data testing terdiri dari 156 spam dan 290 ham.

Tabel 6.18

Data training dan data testing pada ujicoba kesembilan

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1281 | 2732 |
| Testing | 156 | 290 |

Tabel 6.19 menampilkan hasil klasifikasi pada ujicoba kesembilan. Klasifikasi dengan fitur unigram tanpa mempertimbangkan term dan inverse document frequency mengenali 87% dari total 156 twit spam yang diujikan. Tidak ada kesalahan penggolongan terhadap 290 twit ham yang diperiksa. Penghitungan probabilitas dengan mempertimbangkan term dan inverse document frequency meningkatkan tingkat deteksi spam higga 6 % menjadi 93% pengenalan spam atau 145 spam dari total 156 twit spam yang dibaca. Klasifikasi dengan fitur bigram tanpa term dan inverse document frequency mengenali hingga 154 twit spam, akan tetapi terjadi kesalahan penggolongan ham sebagai spam hingga hampir 20% dari 290 twit ham yang diperiksa. Dengan penambahan penghitungan term dan inverse document frequency tingkat kesalahan ini terkurangi sebanyak 6%, tanpa mengurangi tingkat pengenalan spam secara benar. Klasifikasi dengan bigram dan unigram menggunakan smoothing stupid backoff bisa mendeteksi 153 spam dari total 156 twit spam yang diperiksa, dengan 3 kali kesalahan penggolongan ham sebagi spam dari total 290 twit ham yang dibaca.

Tabel 6.19

Hasil klasifikasi ujicoba kesembilan

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Klasifikasi | True Spam | False Spam | True Ham | False Ham | F1 Score | Akurasi |
| Unigram | Multinomialnb Laplace Smoothing | 135 | 0 | 290 | 21 | 0.928 | 0.952 |
| Unigram TF-IDF | Multinomialnb Laplace Smoothing | 145 | 0 | 290 | 11 | 0.963 | 0.975 |
| Bigram | Multinomialnb Laplace Smoothing | 154 | 52 | 238 | 2 | 0.851 | 0.879 |
| Bigram TF-IDF | Multinomialnb Laplace Smoothing | 154 | 34 | 256 | 2 | 0.895 | 0.919 |
| Bigram  Unigram | Multinomialnb Stupid Backoff Smoothing | 153 | 3 | 287 | 3 | 0.981 | 0.986 |

1. Ujicoba klasifikasi Kesepuluh

Pada ujicoba kesepuluh digunakan 10% bagian akhir dari data set yaitu dari indeks nomer 4014 hingga 4458. Dari 445 data testing ini terdapat 137 spam dan 308 ham. Sedangkan data trainingnya mengandung 1300 spam dan 2714 ham.

Tabel 6.20

Data training dan data testing pada ujicoba kesepuluh

|  |  |  |
| --- | --- | --- |
|  | Jumlah Spam | Jumlah Ham |
| Training | 1300 | 2714 |
| Testing | 137 | 308 |

Pada ujicoba kesepuluh ini klasifikasi dengan berdasar fitur unigram/bag-of-words tanpa term dan inverse document frequency mengenali 122 dari total 137 spam yang diperiksa. Di samping itu tidak ada kesalahan penggolongan ham sebagai spam dari 308 twit yang dibaca. Perhitungan probabilitas menggunakan term dan inverse document frequency meningkatkan tingkat deteksi spam sekitar 4 %. Klasifikasi dengan fitur bigram bisa mengenali keseluruhan spam yang diuji, akan tetapi terjadi kesalahan penggolongan ham sebagai spam sebanyak 15,5% dari total 308 twit ham yang diujikan. Perhitungan term dan inverse document frequency bisa menurunkan tingkat error ini sekitar 7,5%, menjadi 25 kesalahan penggolongan spam terhadap twit ham yang dibaca. Sedangkan untuk deteksi spamnya, ada 1 twit spam yang lolos dari pendeteksian. Deteksi spam dengan fitur bigram dan unigram sekaligus menggunakan smoothing stupid backoff mempunyai tingkat deteksi sedikit lebih rendah daripada klasifikasi dengan bigram, yaitu 133 dari 137 spam, akan tetapi di pihak lain tingkat kesalahan penggolongan ham sebagai spam jauh lebih rendah daripada klasifikasi dengan bigram, yaitu hanya ada 3 kali kesalahan penggolongan ham sebagai spam.

Tabel 6.21

Hasil ujicoba kesepuluh

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Fitur | Klasifikasi | True Spam | False Spam | True Ham | False Ham | F1 Score | Akurasi |
| Unigram | Multinomialnb Laplace Smoothing | 122 | 0 | 308 | 15 | 0.942 | 0.967 |
| Unigram TF-IDF | Multinomialnb Laplace Smoothing | 128 | 0 | 308 | 9 | 0.966 | 0.980 |
| Bigram | Multinomialnb Laplace Smoothing | 137 | 48 | 260 | 0 | 0.851 | 0.892 |
| Bigram TF-IDF | Multinomialnb Laplace Smoothing | 136 | 25 | 267 | 1 | 0.913 | 0.942 |
| Bigram | Multinomialnb Stupid Backoff Smoothing | 133 | 3 | 305 | 4 | 0.974 | 0.984 |

1. Analisa Hasil Pengujian

Dari hasil yang didapat dari keseluruhan 10 kali pengujian dilakukan penghitungan rata-rata untuk melihat performa keseluruhan dari metode yang digunakan. Hasil penghitungan rata-rata tersebut bisa dilihat pada Tabel 6.22.

Tabel 6.22

Hasil Akhir Pengujian

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Fitur | Smoothing | True  Spam | False Spam | True Ham | False Ham | F1 Score |
| Unigram | Laplace | 126 | 0 | 302 | 18 | 0.933 |
| Unigram Tf-Idf | Laplace | 132 | 1 | 302 | 12 | 0.954 |
| Bigram | Laplace | 142 | 47 | 255 | 2 | 0.852 |
| Bigram Tf-Idf | Laplace | 141 | 29 | 274 | 3 | 0.899 |
| Bigram& Unigram | Stupid Backoff | 140 | 3 | 300 | 4 | 0.976 |

Dari Tabel 6.22 dilihat bahwa klasifikasi dengan fitur unigram tanpa perhitungan term dan inverse document frequency mengenali rata-rata 87,5% dari total spam yang diujicobakan. Klasifikasi dengan unigram ini memiliki tingkat error penggolongan dari ham ke spam yang sangat rendah. Perhitungan term dan inverse document frequency bisa sedikit meningkatkan kemampuan Naive Bayes classifier-nya dalam mendeteksi spam. Tingkat deteksi spam yang lebih tinggi dilakukan pada klasifikasi dengan fitur model bahasa bigram dengan tingkat pengenalan spam hingga di atas 98% dari total spam yang diujikan. Akan tetapi klasifikasi dengan metode ini terlalu agresiv dalam mendeteksi spam sehingga banyak twit ham yang juga digolongkan sebagai spam. Tingkat kesalahan ini bisa mencapai 15% dari total twit ham yang dibaca. Tentunya hal ini kurang diharapkan untuk informasi-informasi yang krusial. Dengan memperhitungkan term dan inverse document frequency, tingkat kesalahan penggolongan ham sebagai spam bisa dikurangi hingga sekitar 9%. Klasifikasi berdasar fitur unigram dan bigram dengan menggunakan smoothing stupid backoff merupakan kompromi terbaik, dengan tingkat deteksi spam yang tinggi, yaitu sekitar 97%, dan tingkat kesalahan penggolongan twit ham menjadi spam yang sangat rendah, yaitu berkisar 0,9% saja.

Untuk aplikasi pada twitter, di mana informasi di dalamnya tidak begitu krusial, maka pilihan klasifikasi dengan tingkat deteksi spam yang tinggi adalah lebih baik. Sedangkan jika digunakan untuk aplikasi yang banyak informasi-informasi yang sangat krusial, maka menggunakan filter yang lebih hati-hati, seperti pada klasifikasi dengan fitur unigram adalah lebih baik. Secara umum bisa dikatakan dari pengujian ini, performa klasifikasi dengan smoothing stupid backoff mempunyai keunggulan dibandingkan klasifikasi Naive Bayes lain yang diuji.

# BAB VII

PENUTUP

Bab ini berisi kesimpulan dan saran bagi penelitian selanjutnya. Kesimpulan didasarkan pada fakta yang terjadi selama penelitian dan analisa atas hasil penelitian.

1. Kesimpulan

Dari hasil penelitian deteksi spam twit menggunakan metode Naive Bayes dapat disimpulkan sebagai berikut:

1. Sistem klasifikasi yang dibuat terbukti dapat melakukan identifikasi spam di atas 85% dengan tingkat kesalahan klasifikasi ham menjadi spam di bawah 5% sehingga cukup layak untuk digunakan.
2. Klasifikasi dengan fitur unigram dengan F-score 0,933 dan tingkat pengenalan spam sebesar 87,5% dengan kesalahan klasifikasi ham menjadi spam di bawah 1% merupakan klasifikasi yang paling moderat di antara klasifikasi yang diuji.
3. Klasifikasi dengan fitur bigram dengan F-score 0,852 dan tingkat pengenalan spam hingga 98% tingkat kesalahan klasifikasi hingga 15% merupakan filter spam yang paling agresiv.
4. Klasifikasi menggunakan fitur unigram dan bigram sekaligus dengan smoothing stupid backoff dengan F-score 0,976 merupakan klasifikasi yang terbaik dari sistem klasifikasi yang diuji. Penggunaan unigram dan bigram sekaligus ini menghasilkan filter dengan tingkat deteksi tinggi, hingga 97%, dengan tingkat kesalahan yang rendah, yaitu di bawah 1%.
5. Algoritma tf-idf terbukti bisa meningkatkan tingkat deteksi spam hingga beberapa persen.
6. Saran

Untuk perbaikan pada penelitian selanjutnya maka disarankan sebagai berikut:

1. Penggunaan stupid backoff pada penelitian ini baru terbatas pada fitur bigram sebagai gram tertinggi. Akan menarik untuk diteliti lebih jauh klasfikasi dengan gram yang lebih tinggi.
2. Dengan karakteristik twitter yang berisi pesan yang sangat pendek, maka menarik untuk dikembangkan penelitian dengan model bahasa n-gram berbasis karakter.
3. Perlu dikembangkan algoritma stemming yang lebih baik untuk mengatasi bahasa twitter yang tidak baku dan sangat dinamis.

DAFTAR PUSTAKA

Villeneuve, Nart. Koobface: Inside a crimeware network. Munk School of Global Affairs, 2010

Martinez-Romo, Juan, and Lourdes Araujo. "Detecting malicious tweets in trending topics using a statistical analysis of language." Expert Systems with Applications 40, no. 8 (2013): 2992-3000

Benevenuto, F., G. Magno, T. Rodrigues, V. Almeida, and Detecting Spammers on Twitter,” in Collaboration. "electronic messaging, anti-abuse and spam conference (CEAS), vol. 6." Redmond, Washington, July (2010).

Lee, Kyumin, James Caverlee, and Steve Webb. "Uncovering social spammers: social honeypots+ machine learning." In Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval, pp. 435-442. ACM, 2010

Nederhof, Mark-Jan, and Giorgio Satta. "Computation of distances for regular and context-free probabilistic languages." Theoretical Computer Science 395, no. 2-3 (2008): 235-254

Shekar, Chandra, Shruti Wakade, Kathy J. Liszka, and Chien-Chung Chan. "Mining pharmaceutical spam from twitter." In Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on, pp. 813-817. IEEE, 2010

Thomas, Kurt, Chris Grier, Justin Ma, Vern Paxson, and Dawn Song. "Design and evaluation of a real-time url spam filtering service." In Security and Privacy (SP), 2011 IEEE Symposium on, pp. 447-462. IEEE, 2011

Pak, Alexander, and Patrick Paroubek. "Twitter as a corpus for sentiment analysis and opinion mining." In LREc, vol. 10, no. 2010, pp. 1320-1326. 2010

Lin, Jimmy, Rion Snow, and William Morgan. "Smoothing techniques for adaptive online language models: topic tracking in twit streams." In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 422-429. ACM, 2011.

Kaur, Jasleen, and Jatinderkumar R. SAINI. "A Study of Text Classification Natural Language Processing Algorithms for Indian Languages." VNSGU Journal of Scienceand Technology Gujarat, India (2015): 162-167.

Jurafsky, Dan, and James H. Martin. Speech and language processing. Vol. 3. London: Pearson, 2014.

Chen, Stanley F., and Joshua Goodman. "An empirical study of smoothing techniques for language modeling." Computer Speech & Language 13, no. 4 (1999): 359-394.

Yuan, Quan, Gao Cong, and Nadia Magnenat Thalmann. "Enhancing Naive Bayes with various smoothing methods for short text classification." In Proceedings of the 21st International Conference on World Wide Web, pp. 645-646. ACM, 2012.

Brants, Thorsten, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. "Large language models in machine translation." In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). 2007.

Yardi, S., Romero, D. and Schoenebeck, G., 2009. *Detecting spam in a twitter network*. First Monday, 15(1).

1. <https://blog.twitter.com/official/en_us/topics/product/2017/tweetingmadeeasier.html> diakses tanggal 6 Januari 2018 [↑](#footnote-ref-1)
2. [http://www.internetlivestats.com/twitter-statistics/#](http://www.internetlivestats.com/twitter-statistics/), diakses tanggal 20 Juni 2018 [↑](#footnote-ref-2)
3. Villeneuve, Nart. Koobface: Inside a crimeware network. Munk School of Global Affairs, 2010 [↑](#footnote-ref-3)
4. Martinez-Romo, Juan, and Lourdes Araujo, Detecting malicious tweets in trending topics using a statistical analysis of language, 2013 [↑](#footnote-ref-4)
5. Song, Jonghyuk, Sangho Lee, and Jong Kim, Spam filtering in twitter using sender-receiver relationship. Springer, Berlin, Heidelberg, 2011. [↑](#footnote-ref-5)
6. Yardi, Sarita, Daniel Romero, and Grant Schoenebeck, Detecting spam in a twitter network, First Monday 15, no. 1,2009. [↑](#footnote-ref-6)
7. Benevenuto, F., Magno, G., Rodrigues, T., & Almeida, V. Collaboration,

   Electronic messaging. In Anti-Abuse and Spam Conference – CEAS 2010,2010 [↑](#footnote-ref-7)
8. Pak, Alexander, and Patrick Paroubek, Twitter as a corpus for sentiment analysis and opinion mining, 2010 [↑](#footnote-ref-8)
9. Lin, Jimmy, Rion Snow, and William Morgan, Smoothing techniques for adaptive online language models: topic tracking in twit streams, 2011 [↑](#footnote-ref-9)
10. Jonathon Read, Using emoticons to reduce dependency in machine learning techniques for sentiment classification, In ACL. The Association for Computer Linguistics, 2005 [↑](#footnote-ref-10)
11. Ethem Alpaydin, Introduction to Machine Learning (Adaptive Computation and Machine Learning), The MIT Press,2004 [↑](#footnote-ref-11)
12. John D. Lafferty, Andrew McCallum, and Fernando C. N.Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc, 2001 [↑](#footnote-ref-12)
13. Hayter Anthony J, Probability and Statistics for Engineers and Scientists. Duxbury, Belmont, CA, USA, 2007 [↑](#footnote-ref-13)
14. Saif, Hassan; He, Yulan and Alani, Harith, Semantic smoothing for Twitter sentiment analysis., Germany, 2011 [↑](#footnote-ref-14)
15. Ben Allison, David Guthrie, and Louise Guthrie, Another Look at the Data Sparsity Problem, Heidelberg, 2006 [↑](#footnote-ref-15)
16. Klaur & R. Saini, A Study of Text Classification Natural Language Processing rithms for Indian Languages,2015 [↑](#footnote-ref-16)
17. Python 3 text Processing [↑](#footnote-ref-17)
18. [↑](#footnote-ref-18)
19. Han dan Kramber, 2011 [↑](#footnote-ref-19)
20. Gurneet kaur, Er. Neelam Oberai, Naive Bayes Classifier With Modified Smoothing Techniques For Better Spam Classification, 2014 [↑](#footnote-ref-20)
21. Brants, Popat, Xu, Och & Dean, Large Language Models in Machine Translation, Google Inc, [↑](#footnote-ref-21)
22. Twitter Inc, https://developer.twitter.com/en/docs/tweets/search/overview/standard [↑](#footnote-ref-22)
23. Adriani, M., Asian, J., Nazief, B., Tahaghoghi, S. M. M., & Williams, H. E. (2007). Stemming Indonesian. ACM Transactions on Asian Language Information Processing, 6(4), 1–33 [↑](#footnote-ref-23)
24. Hiroshi Shimodaira, Text Classification using Naive Bayes, 2015 [↑](#footnote-ref-24)
25. Sebastian Raschka, naive Bayes and text Classification [↑](#footnote-ref-25)
26. Christopher D. Manning, Prabakhar Raghavan, & Hinrich Schuetze, An Introduction to Information Retrieval, Chambridge UP, 2009 [↑](#footnote-ref-26)
27. T. Brants, dkk, Large language Models in Machine Translation, Google Inc, [↑](#footnote-ref-27)