Pemberian Peringkat Komentar pada *Community Question Answering* dengan Fitur *Soft-Cosine Semantic Similarity* untuk Kasus *Question-External Comment*

Floribertus Yericho Pramudya[1], Ir. Moch. Arif Bijaksana, M.Tech., Ph.D[2]

[1,2]Fakultas Informatika, Universitas Telkom, Bandung
[1]yeyericho@students.telkomuniversity.ac.id, [2]arifbijaksana@telkomuniversity.ac.id

Abstrak

Di era sekarang ini kebutuhan informasi semakin tinggi dengan banyaknya teknologi yang berkembang dengan cepat. Dengan adanya internet yang semakin cepat dan efektif, maka sistem *Community Question Answering* (CQA) sudah dapat dipastikan akan sangat membantu pengguna internet untuk mendapatkan informasi yang dibutuhkan. Dengan masukan berupa dataset berbentuk XML yang berisi pertanyaan baru, pertanyaan relevan dan jawaban. *Output* yang dihasilkan berupa nilai *Mean Average Precision* (MAP) dari sepuluh komentar *good* teratas.. CQA sendiri cukup terbuka untuk umum dan semuanya bebas untuk bertanyajawab, tetapi dengan kebebasan itu pengguna juga disulitkan dengan banyaknya jawaban dan tidak menjamin semuanya benar dan sesuai dengan pertanyaan. Bahkan ada kemungkinan juga jika jawaban yang terbaik ada di pertanyaan lain yang sudah pernah ditanyakan.

Penelitian yang ada sebelumnya menggunakan fitur *Cosine Similarity*. Fitur *Cosine Similarity* hanya mengambil jawaban yang memiliki kesamaan kata dengan pertanyaan yang ada. Sedangkan dengan ditambahkan fitur *Soft-Cosine Semantic Similarity* akan meningkatkan kemungkinan untuk mendapatkan jawaban yang tepat meskipun tidak memiliki kesamaan kata sekalipun.

Pengujian dilakukan menggunakan *dataset* dari SemEval-2017 Task 3 menunjukkan bahwa gabungan fitur *Soft-Cosine Semantic Similarity* dengan algoritma klasifikasi *Support Vector Machine* lebih baik dari kombinasi yang lain. Kombinasi ini menghasilkan nilai MAP sebesar **21.0%** untuk mencari persamaan *Original Question* dengan *Related Comments*.

Kata Kunci : *community question answering*, klasifikasi, pemberian peringkat komentar, *qatar living forum*, *soft-cosine semantic similarity*

Abstract

In this era the needs of information increases with a lot of technology that developed rapidly. With the existence of internet which getting more fast and effective, the Community Question Answering (CQA) system will absolutely help the internet user to get the information that they need. The input is XML dataset with new question, relevance question and comment inside them. The output is result of Mean Average Precision (MAP) from top ten comments.

The CQA itself is quite available for public and all of the internet user are unrestrained to do some question and answer. But with the freedom that they got, user also having some difficulties from getting too much of answers and also there's no guarantee that all of the answers are right and suitable with the question. Even maybe there are some probability that the best answer is from another question that has been asked before.

Some previous research only using Cosine Similarity feature. Cosine similarity feature only take answers which have some letter similarity with the one that is in the question. While the system added by Soft-Cosine Semantic Similarity, that will increasing the probability on getting the right answer with having no word similarity.

The SemEval-2017 Task 3 dataset testing result showed us that the combination between Soft-Cosine Semantic Similarity feature with Support Vector Machine algorithm classification is better than the other combination. The MAP value from this combination is **21.0%** which is used to search the similarity between Original Question and Related Comments.

Keywords : *classification, comment ranking, community question answering, qatar living forum, soft-cosine semantic similarity*

1. Pendahuluan

1.1. Latar Belakang

Di zaman modern ini manusia sudah semakin bergantung kepada internet, serta diikuti dengan kebutuhan akan informasi juga meningkat. Maka sistem *Community Question Answering* (CQA) akan membantu pengguna untuk menemukan informasi yang dibutuhkan, contohnya *Qatar Living Forum* yang merupakan sumber dari *dataset* yang disediakan SemEval-2017 Task 3. Selain itu CQA merupakan tempat umum yang bisa diakses oleh semua orang. Semua orang dapat mengajukan pertanyaan dan memberikan jawaban sesuai keinginan mereka sendiri sehingga untuk mendapatkan jawaban yang sesuai seperti yang diharapkan sangat sulit karena banyaknya jawaban di setiap pertanyaan dan tidak semua jawaban sesuai dengan pertanyaan yang ditanyakan. Terutama jika jawaban yang paling tepat ada di pertanyaan lain yang serupa dengan jumlah yang banyak. Maka dari itu akan

memakan banyak waktu untuk mendapatkan jawaban yang tepat dan hal ini sangat tidak menguntungkan dan membuang-buang waktu.

Dalam sistem *Community Question Answering* maka banyak juga celah dari sistem yang dapat dikembangkan lagi. Terutama pada bagian pemilihan jawaban yang sangat susah untuk dilakukan secara *manual*, sehingga dibutuhkan otomasi yang dapat membantu pengguna memilah jawaban yang berkualitas dengan yang tidak berkualitas. Bantuan yang dapat diberikan berupa pemberian peringkat jawaban.

Pada tugas akhir ini akan mengembangkan sistem yang akan membantu pengguna CQA untuk mendapatkan jawaban yang relevan dari banyak jawaban dari setiap pertanyaan terkait, karena dengan banyaknya jawaban akan menambah waktu yang digunakan pengguna CQA dalam mencari jawaban yang tepat. Tetapi dengan adanya dua metode yang serupa yaitu *Cosine Similarity* dan *Soft-Cosine Semantic Similarity* (SCSS) akan dibandingkan untuk menentukan metode yang lebih membantu pengguna dalam mencari jawaban, karena SCSS merupakan metode yang belum lama muncul dan lebih digunakan untuk menangani kasus *similarity* antar pertanyaan [1] berbeda dengan *Cosine Similarity* yang sudah banyak digunakan untuk menangani berbagai kasus *similarity*. Data yang akan digunakan merupakan *dataset* yang disiapkan SemEval-2017 Task 3 yang berasal dari *Qatar Living Forum* dan berbahasa Inggris.

1.2. Topik dan Batasan

Berdasarkan deskripsi latar belakang yang sudah dipaparkan maka dapat dirumuskan beberapa masalah di tugas akhir ini seperti cara mengidentifikasi komentar yang relevan, membandingkan fitur yang mampu menghubungkan keterkaitan antar pertanyaan dengan komentar relevan, dan memberikan peringkat pada komentar *Good*. Dari semua permasalahan yang ada dibuat batasan dalam melakukan penelitian berupa data yang disiapkan oleh SemEval -2017 Task 3 yang berisi dataset dari *Qatar Living Forum*, lalu fitur yang digunakan hanya *Cosine Similarity* dan *Soft-Cosine Semantic Similarity* dan yang terakhir proses klasifikasi dalam pemberian peringkat dengan mengambil sepuluh kelas *good* teratas untuk menentukan kualitas.

1.3. Tujuan

Tujuan dari tugas akhir ini untuk menjawab permasalahan yang ada yaitu dapat mengidentifikasi komentar yang relevan dengan menggunakan fitur yang ditentukan dilanjutkan dengan mendapatkan fitur yang berdampak besar terhadap performa dan diakhiri dengan memberikan peringkat komentar sesuai hasil klasifikasi.

1.4. Organisasi Tulisan

Jurnal TA ini terdiri dari beberapa bagian setelah pendahuluan yaitu Studi Terkait, Sistem yang Dibangun, Evaluasi, dan Kesimpulan. Bagian studi terkait menjelaskan mengenai teori dasar yang digunakan dalam penanganan *question answering* seperti jenis-jenis kasus pada community question answering, data yang digunakan, penjelasan praproses, ekstraksi fitur, klasifikasi, pemberian peringkat, evaluasi performansi sistem dan *library* yang digunakan dalam penelitian. Bagian sistem yang sibangun menjelaskan mengenasi rancangan sistem yang dibangun berdasarkan mekanisme dan batasan yang digunakan. Bagian evaluasi menjelaskan mengenai data yang digunakan, pengujian dan hasil pengujian. Analisis menggunakan hasil dari ekstraksi fitur dan klasifikasi. Bagian kesimpulan berisi kesimpulan terhadap pengaruh ekstraksi fitur dan klasifikasi terhadap nilai MAP yang dihasilkan. Saran sebagai masukan untuk penelitian lebih lanjut.

2. Studi Terkait

2.1. Questions Answering

Sistem *Question Answering* (QA) merupakan suatu sistem komputer yang memiliki kemampuan untuk mengakses beberapa sumber informasi dan menggunakan sumber informasi yang didapat untuk menjawab pertanyaan yang disampaikan dalam natural language oleh pengguna ke dalam sistem. Sistem *question answering* mirip dengan mesin pencari yang menyediakan antarmuka bagi pengguna untuk mencari informasi. Namun perbedaan mendasar dari sistem question answering dengan mesin pencari ada pada hasil informasi yang dihasilkan. Hasil keluaran dari mesin pencari adalah satu himpunan dokumen yang cocok dengan permintaan pengguna. Mesin pencari menggunakan kata kunci untuk melakukan pencarian hasil dalam sistem. Di sisi lain, *sistem question answering* mengambil pertanyaan dari *natural language* sebagai masukan dan mengembalikan jawaban spesifik sebagai hasil keluaran. [1]

Community Question Answering (CQA) tidak jauh berbeda dengan QA. Perbedaan terletak pada komentar atau jawaban. Pada CQA komentar atau jawaban dari pertanyaan yang diajukan dijawab oleh manusia. Komentar yang diperoleh bervariatif, mulai dari dengan kualitas tinggi hingga rendah.

2.1.1. Question-Comment Similarity

Question-Comment Similarity merupakan jenis CQA yang menilai kesamaan antara pertanyaan dengan komentar yang ada [2]. Dengan adanya kesamaan maka komentar tersebut di posisikan sebagai kelas *Good*, tetapi masuk ke dalam kelas *Bad* di dalam evaluasi kelas *PotentiallyUsefull*. Evaluator performa akan menggunakan *mean average precision*. Fitur yang digunakan dalam CQA ini, yaitu: kemiripan subjek antara pertanyaan dengan komentar, isi pertanyaan dengan komentar, dan subjek beserta isi pertanyaan dengan komentar.

2.1.2. Question-Question Similarity

Question-Question Similarity merupakan jenis CQA yang menemukan pemeringkatan pertanyaan yang berhubungan dengan pertanyaan terbaru [2]. Pertanyaan yang tergolong *Perfect Match* dan *Relevant* akan ditempatkan di atas *Irrelevant*. Evaluator performa akan menggunakan *mean average precision*. Fitur yang digunakan dalam CQA ini, yaitu: hubungan antara subyek dengan subyek, isi pertanyaan dengan isi pertanyaan, dan subyek beserta isi pertanyaan dengan subyek beserta isi pertanyaan.

2.1.3. Question-External Comment Similarity

Question-External Comment Similarity merupakan jenis CQA yang mengurutkan komentar atau jawaban berdasarkan tingkat relevansi terhadap pertanyaan yang asli. Komentar atau jawaban yang memiliki kelas *Good* akan ditempatkan di atas kelas *PotentiallyUsefull* dan *Bad* yang tergolong *Bad*. Sistem hanya akan mengevaluasi sepuluh komentar teratas dengan asumsi pengguna menginginkan sepuluh komentar teratas sebagai komentar *Good* [2].

2.2. Dataset

Dataset yang umum digunakan pada penelitian *natural language processing*.

2.2.1. SemEval-2017 Task 3 (Training Data)

Dataset untuk melakukan proses learning pada sistem yang disediakan oleh SemEval-2017 Task 3 tetap menggunakan *dataset* yang sama dengan SemEval-2016 Task 3. *Dataset* tersebut mengambil dari *Qatar Living Forum* dalam Bahasa Inggris. *Dataset* berisi sejumlah *original question* yang memiliki *related question* dan masing-masing memiliki *related comment* [2].

2.2.2. SemEval-2017 Task 3 (Test Data)

Dataset untuk melakukan proses *testing* yang disediakan oleh SemEval-2017 Task 3 tetap menggunakan *dataset* yang sama dengan SemEval-2016 Task 3. *Dataset* tersebut mengambil dari *Qatar Living Forum* dalam Bahasa Inggris. *Dataset* ini menyediakan dua data *test* berupa data yang tidak memiliki nilai relevansi dan memiliki nilai relevansi yang sesuai [2].

2.3. Extensible Markup Language

Dalam kasus penghitungan peringkat pada sistem *question answering forum*, format data yang digunakan adalah format XML. XML merupakan bahasa yang digunakan untuk mendefinisikan data pada suatu dokumen. XML dikembangkan dari *Standard Generalized Markup Language*.

Dokumen XML tersusun dari kumpulan entitas yang mengandung data. Data tersusun dari karakter, di mana karakter tadi dapat berupa data. Secara fisik, dokumen terdiri dari kumpulan unit bernama entitas. Entitas dapat menyertakan entitas lain di dalamnya. Secara logis, dokumen tersusun atas deklarasi, elemen, komentar, referensi karakter dan instruksi pemrosesan [3].

2.4. Praproses

Teks yang didapatakan dari *dataset* harus diolah terlebih dahulu sebelum dapat diproses. Hal ini ditujukan untuk merapikan pertanyaan dan komentar yang didapatkan sehingga siap diproses oleh sistem. Dalam pengolahan *dataset* yang didapat, ada beberapa teknik dalam praproses dapat digunakan untuk mempermudah pengambilan data dari teks.

2.4.1. Text Extraction

Pertanyaan dan komentar yang terkandung di dalam *dataset* berada dalam format xml. *Tag* dan atribut yang tidak dibutuhkan tidak akan diambil dan tidak diikutkan dalam proses selanjutnya. Tujuan utama tahap ini merupakan pengambilan isi dari pertanyaan dan komentar.

2.4.2. Tokenization

Tokenization adalah proses yang dilakukan untuk mendapatkan setiap kata yang digunakan pada suatu teks. Teks dipisah menjadi kumpulan kalimat atau kata dengan atau tanpa menghilangkan karakter yang tidak diinginkan seperti tanda baca [4].

2.4.3. Stopword Removal

Stopwords dihapus dari teks karena hanya mengandung sedikit dan bahkan tanpa informasi, seperti kata sambung dan kata depan. Dalam Bahasa Inggris contoh *stopword* adalah "*on*", "*and*" dan "*in*". Selain sedikitnya informasi yang terkandung, *stopword* juga dihapus untuk mempersingkat waktu yang diperlukan oleh sistem.

2.4.4. Lemmatization

Lemmatization merupakan tahap untuk menghasilkan lemma atau bentuk dasar dari suatu kata. Sebuah kata dasar dapat memiliki bentuk lampau dan bentuk waktu sekarang. Sebagai contoh *used, user, using* dapat diubah menjadi kata dasarnya yaitu use. Tujuan utama tahap ini adalah mengurangi jumlah kata yang digunakan lebih lanjut dan memudahkan proses pencarian kata.

2.4.5. Stemming

Stemming merupakan proses pemotongan untuk menghilangkan imbuhan pada suatu kata. Sebuah kata dapat memiliki imbuhan seperti "*studies*" dan "*studying*" akan berubah menjadi "*studi*" dan "*study*" setelah adanya pemotongan imbuhan "*-es*" dan "*-ing*". Tujuan dari tahap ini adalah mengurangi imbuhan pada tiap kata untuk mendapatkan kata dasar secara cepat tanpa ada perubahan huruf layaknya *lemma*, tetapi tanpa adanya perubahan huruf mengakibatkan dampak negatif seperti lebih rendahnya nilai *similarity*.

2.4.6. POS Tagging

POS tagging adalah proses pemberian *POS tag* yang sesuai dengan kata berdasarkan konteks kemunculan kata. *POS tag* memisahkan kata ke dalam kelompok berdasarkan peran yang dimainkan kata pada kalimat. *POS tags* memberikan informasi semantik mengenai sebuah kata. Beberapa jenis *tag* yang umum digunakan adalah *noun*, *verb*, *adjective* dan *preposition*.

2.4.7. Doc2vec

Doc2vec merupakan proses untuk mengubah *natural language* menjadi vektor. Tujuan dari adanya *doc2vec* sendiri untuk membantu sistem memproses dokumen, dikarenakan sistem tidak bisa melakukan penanganan lebih lanjut jika dokumen tetap dalam bentuk *natural language*. Setiap dokumen akan menghasilkan seratus nilai vektor tidak peduli berapapun jumlah kata yang terdapat di dalam dokumen tersebut.

2.5. Ekstraksi Fitur

Ekstraksi fitur merupakan tahap untuk menemukan fitur atau ciri yang dapat diukur. Fitur yang diperoleh dari teks dianalisis lebih lanjut pada proses berikutnya. Beberapa fitur dipilih untuk menemukan tingkat relevansi dari pertanyaan dengan komentar.

2.5.1. Word Matching

Word matching merupakan fitur yang diambil saat sebuah kata memenuhi kriteria tertentu. Fitur *word matching* yang di gunakan yaitu *question author*. *Question author* akan menententukan apakah ada kata yang benar-benar sama dengan yang diinginkan. Fitur ini mengidentifikasi apakah sebuah komentar berasal dari pengguna yang mengajukan pertanyaan dengan melihat kesamaan identitas pengguna. Komentar yang diberikan oleh pemberi pertanyaan kemungkinan besar merupakan informasi lebih lanjut mengenai pertanyaan yang diberikan atau berupa *feedback* [5].

2.5.2. Similarity

Similarity merupakan fitur yang diambil dengan mengukur kemiripan dua buah kata atau dokumen.

2.5.2.1. Cosine Similarity

Cosine Similarity merupakan fitur leksikal dalam menghitung kemiripan dua buah dokumen yang telah diubah ke dalam bentuk vektor dengan menghitung sudut kosinus [5]. Dua dokumen yang memiliki banyak kata yang sama kemungkinan memiliki nilai kemiripan yang tinggi.

$$CosineSimilarity = \frac{A.B}{\|A\|.\|B\|} = \frac{\sum_{i=1}^{n} u_i.v_i}{\sqrt{\sum_{i=1}^{n}(u_i)^2}.\sqrt{\sum_{i=1}^{n}(v_i)^2}} \quad \textbf{(2.1)}$$

Dengan *u* dan *v* adalah vektor dari dokumen pertanyaan dan komentar yang didapat dari *doc2vec*, dengan *u* merupakan vektor dari dokumen pertanyaan asli dan *v* merupakan vektor dari dokumen komentar yang berasal dari dokumen pertanyaan yang relevan dengan pertanyaan asli.

2.5.2.2. Soft-Cosine Semantic Similarity

Soft-Cosine Semantic Similarity secara garis besar sama seperti *Cosine Similarity* ditambah dengan relasi akun antar kata. Fitur ini biasa digunakan untuk mencari kesamaan antar pertanyaan [1]. Perbedaan yang mencolok dengan *Cosine Similarity* adalah *Soft-Cosine Semantic Similarity* bisa melakukan penghitungan walaupun tidak ada kata yang sama antar variabel dengan persamaan di bawah.

$$Cosine_M = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} u_i m_{i,j} v_j}{\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} u_i m_{i,j} u_j}.\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} v_i m_{i,j} v_j}} \quad \textbf{(2.2)}$$

Dimana $m_{i,j}$ merupakan relasi antara kata *i* dan kata *j*. Lalu kesamaan antar teks tidak akan *null* setelah berbagi kata yang berelasi, walaupun tidak ada kata yang sama. Jika sebuah kata hanya memiliki relasi ke diri mereka sendiri maka *soft-cosine* akan kembali ke *cosine*. Rumus $m_{i,j}$ terbagi menjadi dua sesuai dengan relasi yang akan digunakan, yaitu: *semantic relations* dan *edit-distance based relations* [1].

2.5.2.2.1. Semantic Relations

Semantic relations yang relevan antar kata dapat didapatkan dengan *simple similarity measure (cosine)* antar representasi vektor dari kata-kata tersebut [1]. Maka persamaan yang akan digunakan seperti di bawah.

$$m_{i,j} = \max(0, CosineSimilarity(v_i, v_j))^2 \quad \textbf{(2.3)}$$

Berdasarkan pada rumus di atas, angka 0 merupakan pengganti jika *CosineSimilarity* menghasilkan nilai negatif yang sulit diinterpretasikan dan sering tidak relevan.

2.5.2.2.2. Edit-Distance Based Relations

Edit-distance based relations menggunakan *Levenshtein Distance* antar kata sehingga *edit-distance based relations* dapat dihitung walaupun ada salah penulisan kata [1]. Dan hal itu didefinisikan sebagai berikut $m_{i,j} = 1$ dan $i \neq j$ sehingga didapatkan persamaan di bawah.

$$m_{i,j} = \alpha * \left(1 - \frac{Levenshtein(w_i, w_j)}{\max(\|w_i\|, \|w_j\|)}\right)^{\beta} \quad \textbf{(2.4)}$$

Dengan $\|w\|$ merupakan jumlah karakter dari suatu kata, α merupakan faktor pembanding relatifitas dengan elemen diagonal dan β merupakan faktor yang membuat nilai lebih dinamis.

Levenshtein merupakan algoritma untuk mencari jarak atau jumlah perbedaan huruf antar kata, seperti contohnya kata "aku" dan "kamu" maka nilai *levenshtein*nya adalah 1 (satu) karena adanya huruf yang berpasangan antara dua kata tersebut yaitu: "a", "k", dan "u" sedangkan huruf yang tidak memiliki pasangan hanya "u" saja.

2.6. Klasifikasi

Klasifikasi merupakan sebuah proses dalam mengelompokkan suatu data ke dalam suatu kelas. Diantara beberapa algoritma klasifikasi yang ada, yang akan digunakan dalam penelitian ini adalah *Logistic Regression* dan *Support Vector Machine*.

2.6.1. Logistic Regression

Regresi merupakan salah satu analisis yang memperkirakan hasil dependen variabel berdasarkan variabel independen. Pada *Logistic Regression*, variabel dependen adalah kategori. *Logistic Regression* dapat memprediksi dengan dua variabel dependen atau lebih [6]. Dua variabel seperti variabel kategori baik dan buruk, tinggi dan rendah atau 0 dan 1. Regresi logistik multinomial digunakan apabila variabel dependen yang digunakan terdiri dari dua atau lebih variabel kategorik. Sedangkan *Logistic Regression* ordinal digunakan apabila variabel dependennya menggunakan skala ordinal seperti sangat baik, baik, buruk dan sangat buruk. *Logistic Regression* multinomial juga digunakan untuk menganalisis hubungan antara satu atau lebih variabel bebas baik yang nominal maupun kategorikal dengan variabel tergantung berupa data outcome kategorikal yang lebih dari 2 kategori.

2.6.2. Support Vector Machine

Support Vector Machine (SVM) adalah sistem pembelajaran yang sangat cepat dan efektif untuk permasalahan klasifikasi teks. SVM menggunakan ruang hipotesis berupa fungsi-fungsi linier dalam sebuah *feature space* dengan dimensi tinggi. SVM dilatih dengan algoritma pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan induktif bias yang berasal dari teori pembelajaran statistik [7].

Binary SVM classifier dapat dianggap sebagai *hyperplane* dalam *feature space* yang memisahkan titik yang merepresentasikan nilai positif dari titik yang merepresentasikan nilai negatif. *Hyperplane* dengan margin tertinggi akan yang akan digunakan untuk memisahkan antar kelas. Margin merupakan jarak dari *hyperplane* ke titik terdekat pada himpunan positif dan negatif [4].

2.7. Evaluasi Perfomansi Sistem

Pada akhir implementasi dilakukan evaluasi hasil keluaran sistem sebagai tahap pengujian. Terdapat dua buah evaluasi yaitu evaluasi klasifikasi dan evaluasi pemberian peringkat.

2.7.1. Evaluasi Klasifikasi

Evaluasi yang melihat nilai F-score dan akurasi sebagai acuan akan kualitas pengambilan keputusan oleh sistem. Ruang lingkup dalam evaluasi klasifikasi yang bisa diperoleh yaitu:

- *True Positive* (TP), saat kelas diprediksi *true* dan faktanya *true*.
- *False Positive* (FP), saat kelas diprediksi *true* dan faktanya *false*.
- *True Negative* (TN), saat kelas diprediksi *false* dan faktanya *false*.
- *False Negative* (FN), saat kelas diprediksi *false* dan faktanya *true*.

2.7.1.1. Precision, Recall dan Accurracy

Precision digunakan untuk mengukur ketepatan sistem dalam menentukan dokumen relevan pada pencarian dari dokumen yang diterima, dengan kata lain hasil *precision* merupakan nilai dari kemampuan sistem dalam mengambil keputusan.

Recall digunakan untuk mengukur ketepatan sistem dalam menentukan dokumen relevan pada pencarian dari keseluruhan dokumen relevan yang ada, dengan kata lain hasil *recall* merupakan nilai dari ketepatan sistem mendapatkan dokumen relevan sesungguhnya.

Accurracy digunakan untuk mendapatkan ketepatan pengambilan keputusan oleh sistem untuk menentukan kerelevanan dokumen.

$$precision = \frac{TP}{TP+FP} \quad \textbf{(2.5)}$$

$$recall = \frac{TP}{TP+FN} \quad \textbf{(2.6)}$$

$$accurracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \textbf{(2.7)}$$

2.7.1.2. F1 Measure

F1 measure digunakan untuk menghitung rata-rata performa harmonis dari *precision* dan *recall*. *F1 measure* dibutuhkan karena dengan hasil dari *precision* dan *recall* masing-masing tidak bisa menentukan ketepatan sistem. Dengan adanya *F1 measure* jika *precision* dan *recall* memiliki bobot yang berbeda tetap didapatkan hasil rata-rata performa yang sudah ternormalisasi.

$$F1 = \frac{2*precision*recall}{precision+recall} \quad \textbf{(2.8)}$$

2.7.2. Evaluasi Pemberian Peringkat

Pemberian peringkat merupakan proses yang berbeda dengan klasifikasi. Perfoma pemeringkatan juga dipengaruhi oleh performa klasifikasi, karena dalam pemeringkatan diperlukan kelas dan urutan yang tepat. Dalam hal ini diperlukan *Mean Average Precision* (MAP). MAP merupakan pengukuran yang menghasilkan satu nilai pada semua tingkat recall yang berbeda. Untuk satu kebutuhan informasi, *Average Precision* (AP) adalah rata-rata dari nilai *precision* untuk k dokumen teratas dengan memperhitungkan dokumen yang relevan dan nilai ini dirata-ratakan terhadap kebutuhan informasi.

$$MAP(Q) = \frac{1}{Q}\sum_{j-1}^{Q}\frac{1}{q_j}\sum_{k=1}^{q_j} precision(doc_k) \quad \textbf{(2.9)}$$

Dengan *Q* adalah jumlah query pencarian, q_j adalah jumlah dokumen relevan untuk *query j* dan $precision(doc_k)$ merupakan nilai precision dokumen relevan pada posisi *k* [2].

2.8. Library

Library merupakan sarana pendukung untuk mempermudah *programmer* untuk menggunakan algoritma tertentu yang bersifat permanen dan memiliki *input* dan *output* yang pasti.

2.8.1. SKLEARN

Scikit-learn atau yang biasa disebut sklearn merupakan modul untuk bahasa pemrograman Python. Modul ini membuat pembelajaran mesin lebih mudah untuk pemula menggunakan bahasa tingkat tinggi. *Scikit-learn* mengimplementasikan banyak algoritma pembelajaran mesin dan tetap mudah digunakan dalam bahasa Python. Hal itu merupakan jawaban untuk banyaknya kebutuhan dari pemula atau *non-specialists* dalam industri perangkat lunak dan *web* [8]. Pada penelitian ini *Scikit-learn* digunakan untuk membantu proses klasifikasi.

2.8.2. NLTK dan GENSIM

Natural Language Toolkit (NLTK) dibuat di Universitas Pennsylvania pada tahun 2001. NLTK didesain dengan tiga aplikasi pembelajaran yaitu: *Assignments*, *Demonstrations*, dan *Projects* [9]. Pada penelitian ini NLTK digunakan untuk membantu proses *preprocessing*.

Gensim merupakan *library* untuk *digital document indexing* dan *similarity search* [10]. Pada penelitian ini gensim digunakan untuk membantu proses *preprocessing* yaitu doc2vec.

3. Sistem yang Dibangun

3.1. Gambaran Umum Sistem

Pada tugas akhir ini akan dibangun sistem yang memberikan peringkat pada setiap komentar dari sebuah pertanyaan. Gambaran umum sistem sebagai berikut diawali dengan sistem menerima masukan *dataset* dengan format XML yang berisi *original question, related question* dan *related comment* dilanjutkan dengan menerapkan praproses pada dataset. Setelah praproses selesai dilakukan ekstraksi fitur berupa *similarity* dan *word matching*. Hasilnya digunakan untuk membangun model pembelajaran dan melakukan klasifikasi dengan menggunakan *Logistic Regression* dan *Support Vector Machine*. Memberikan peringkat berdasarkan nilai probabilitas komentar untuk mendapatkan kelas *Good*. Melakukan evaluasi peringkat dengan *Mean Average Precision* dari hasil klasifikasi yang didapatkan.

3.2. Praproses

Praproses dilakukan menggunakan beberapa teknik secara terurut setelah berkas selesai dimuat berupa: ***Text extraction*** untuk mendapatkan *original question* dan *related comment*; ***Tokenization*** mengubah dokumen menjadi kumpulan *token* untuk diproses. Contohnya ada satu kalimat *"Is there any place I can find scented massage oils in Qatar?"* maka akan menjadi *"Is", "there", "any", "place", "I", "can", "find", "scented", "massage", "oils", "in", "Qatar"*; ***Stopword removal*** untuk menghilangkan *stopword* untuk meningkatkan performa. Contohnya hasil *tokenization* didapat *"Is", "there", "any", "place", "I", "can", "find", "scented", "massage", "oils", "in", "Qatar"* maka akan menjadi *"place", "find", "scented", "massage", "oils", "Qatar"*; ***Lemmatization*** untuk mendapatkan kata dasar dari sebuah kata. Contohnya hasil *stopword removal* didapat *"place", "find", "scented", "massage", "oils", "Qatar"* maka akan menjadi *"place", "find", "scent", "massage", "oil", "Qatar".*

3.3. Ekstraksi Fitur

Proses ekstraksi fitur pada sistem akan menerima masukan berupa teks yang telah melewati praproses. Kelompok fitur similarity diterapkan pada original question, related question dan related comment secara berpasangan.

3.3.1. Question Author

Fitur *question author* memeriksa apakah pertanyaan dan komentar dibuat oleh pengguna yang sama. Contoh: ***Related Question***: *Where is nearby oil shop?* (ditulis oleh Billy, berninai *true*); ***Related Comment***: *Try this one, It is right beside the gate of the market.* (ditulis oleh Tomy, berninai *false*); ***Related Comment***: *What is the name of the shop?* (ditulis oleh Billy, berninai *true*) maka komentar yang ditulis oleh pemberi pertanyaan tidak akan diproses lebih lanjut.

3.3.2. Cosine Similarity

Cosine Similarity memperoleh kemiripan dua teks dengan menghitung sudut kosinus dua vektor terhadap pendekatan pertanyaan dengan komentar dan pertanyaan dengan pertanyaan. Contohnya vektor dari dua kalimat pertanyaan berikut "*Where I can buy good oil for massage*?" dan "*is there any place i can find scented massage oils in qatar*?" menghasilkan nilai *cosine* sebesar 0.03154874795903651.

3.3.3. Soft-Cosine Semantic Similarity

Soft-Cosine Semantic Similarity memperoleh kemiripan dua teks dengan menghitung sudut kosinus dua vektor terhadap pendekatan pertanyaan dengan komentar dan pertanyaan dengan pertanyaan dengan tambahan penghitungan *Edit-Distance Based Relations* atau *Semantic Relations*. *Soft-Cosine Semantic Similarity* sendiri dianggap dapat menemukan *similarity* walaupun tidak ada kata yang sama antar dokumen. Contohnya vektor dari dua kalimat pertanyaan berikut "*Where I can buy good oil for massage*?" dan "*Yes. It is right behind Kahrama in the National area.*" menghasilkan nilai *soft-cosine* sebesar 0.0 untuk yang menggunakan *Edit-Distance Based Relations dan* 1.0000000106759594 untuk yang menggunakan *Semantic Relations*.

3.4. Klasifikasi dan Pemberian Peringkat Komentar

Nilai hasil dari tahap praproses digunakan untuk parameter pembangunan model. Proses ini dilakukan dengan melakukan percobaan kombinasi fitur dan konfigurasi pada parameter tertentu. *Classsifier* yang digunakan yaitu *Logistic Regression* (LR) dan *Support Vector Machine* (SVM) dengan tujuan membandingkan penggunaan klasifikasi dengan *kernel linier* yaitu LR dengan *kernel non-linier* yaitu SVM. Hasil yang akan dikeluarkan berupa nilai penentuan kelas. Dengan nilai tersebut diberikan peringkat hanya pada kelas *Good* karena kelas *Good* harus berada di atas kelas lain.

3.5. Evaluasi

Sistem dievaluasi menggunakan MAP untuk menilai kualitas peringkat yang diberikan. Pengujian untuk memastikan kelas *Good* berada di atas kelas lain. Komentar kelas *Good* dikelompokkan menurut *original question*. Setiap *original question* diambil maksimal 10 peringkat teratas untuk dievaluasi.

4. Evaluasi

4.1. Hasil Pengujian

Dalam proses pengujian ada beberapa fitur yang pada akhirnya harus ditinggalkan, seperti *stemming* dan *pos tagging*. Fitur *stemming* dan *pos tagging* ditinggalkan karena *stemming* cenderung menambah nilai dari *Edit-Distance Based Relations* karena bersifat hanya memotong kata dan tidak mengubah huruf dari kata dasar yang berubah saat diberi imbuhan menjadi kata dasar yang sebenarnya berbeda dengan *pos tagging* yang pada akhirnya tidak berpengaruh apapun pada proses pengujian.

Hasil pengujian utama berupa nilai MAP dan didukung oleh nilai *precision* (P), *recall* (R), *accurracy* (A) dan *F1-Measure* (F1), selain itu klasifikasi yang digunakan tanpa adanya perubahan parameter. MAP menunjukkan nilai rata-rata P pada tiap pertanyaan. P menunjukan nilai positif yang tepat dari semua nilai positif yang diberikan. R menunjukkan nilai positif yang tepat dari nilai nilai positif yang sesungguhnya. A menunjukkan nilai positif dan negatif yang tepat dari nilai positif dan negatif yang sesungguhnya. F1 menunjukkan nilai rata-rata dari P dengan R yang telah ternormalisasi. Hasil pengujian dapat dilihat pada tabel di bawah ini dengan tanda cetak tebal sebagai nilai maksimal yang didapat dari tiap klasifikasi dan metrik evaluasi.

LR	MAP	P	R	A	F1
Cosinus	**18.5%**	**8.8%**	**50.4%**	50.6%	**15.0%**
Soft Cosinus (ED)	15.6%	8.6%	26.1%	69.7%	13.0%
Soft Cosinus (SR)	13.3%	8.3%	24.9%	**69.8%**	12.4%

Tabel 4.1 Performa klasifikasi Logistic Regression untuk OQRC

SVM	MAP	P	R	A	F1
Cosinus	16.3%	**9.5%**	38.8%	**62.6%**	15.2%
Soft Cosinus (ED)	**21.0%**	9.0%	55.2%	48.1%	**15.5%**
Soft Cosinus (SR)	**21.0%**	8.4%	**74.1%**	28.2%	15.1%

Tabel 4.2 Performa klasifikasi SVM untuk OQRC

Hasil pengujian pemberian peringkat untuk kasus *Question-External Comment* menggunakan *Soft-Cosine Semantic Similarity* menghasilkan nilai MAP tidak lebih dari **21.0%** dan kualitas dari hasil klasifikasi yang diambil sepuluh peringkat teratas tergolong buruk, dikarenakan sistem terlalu mudah memberikan nilai positif kepada

komentar sehingga didapatkan rata-rata maksimal nilai kebenaran dari sepuluh komentar *good* teratas hanya **9.5%** dan menunjukkan jika **90.5%** dari nilai positif yang diberikan seharusnya bernilai negatif.

Selanjutnya dilakukan pengujian tambahan untuk kasus *Question-Question* dan *Question-Comment*. Pengujian tersebut menggunakan seluruh data tanpa adanya pemberian peringkat. Hasil pengujian menunjukkan rata-rata nilai MAP, P, R dan F1 lebih baik dibandingkan dengan adanya pemberian peringkat.

LR	MAP	P	R	A	F1
Cosinus	**46.6%**	33.4%	**47.5%**	50.4%	**39.2%**
Soft Cosinus (ED)	41.8%	30.3%	24.2%	55.6%	26.9%
Soft Cosinus (SR)	43.4%	**33.7%**	26.0%	**57.8%**	29.4%

Tabel 4.3 Performa klasifikasi Logistic Regression untuk OQRQ

SVM	MAP	P	R	A	F1
Cosinus	**45.0%**	**34.4%**	**52.5%**	50.2%	**41.6%**
Soft Cosinus (ED)	35.5%	30.8%	51.1%	44.7%	38.4%
Soft Cosinus (SR)	44.2%	33.7%	26.0%	**57.8%**	29.4%

Tabel 4.4 Performa klasifikasi SVM untuk OQRQ

LR	MAP	P	R	A	F1
Cosinus	**56.3%**	44.0%	48.7%	49.6%	46.2%
Soft Cosinus (ED)	54.4%	**52.2%**	31.4%	**56.6%**	39.2%
Soft Cosinus (SR)	53.1%	43.4%	**69.5%**	46.1%	**53.5%**

Tabel 4.5 Performa klasifikasi Logistic Regression untuk RQRC

SVM	MAP	P	R	A	F1
Cosinus	56.4%	44.7%	**97.2%**	45.2%	**61.2%**
Soft Cosinus (ED)	**57.6%**	**50.8%**	60.2%	**56.3%**	55.1%
Soft Cosinus (SR)	49.3%	42.2%	21.1%	52.0%	28.2%

Tabel 4.6 Performa klasifikasi SVM untuk RQRC

4.2. Analisis Hasil Pengujian

Dilihat dari hasil pengujian, nilai MAP tidak selalu menjamin kualitas dari proses penilaian relevansi. Pada kasus utama menunjukkan hasil pengujian kualitas yang buruk untuk semua metode dan klasifikasi. Lalu didapatkan fakta hubungan antar hasil uji P, R, F1 dan A seperti tabel di bawah.

P	R	F1	A
+	+	+	+
+	-	-	+
-	+	+	-
-	-	-	+
(+) = 50%~		(-) = ~50%	

Tabel 4.7 Tabel fakta hasil uji kualitas

Pada kasus *Question-External Coment* dengan mengambil sepuluh komentar kelas *good* teratas, fitur *Soft-Cosine Semantic Similarity* menunjukkan hasil MAP yang lebih baik dari fitur *Cosine Similarity* menggunakan algoritma klasifikasi SVM, tetapi memiliki kualitas yang tidak baik. Dalam segi kualitas lebih baik menggunakan *Cosine Similarity* dan klasifikasi LR, karena dengan nilai MAP yang sedikit lebih rendah dari nilai terbaik tetapi memiliki hasil uji kualitas yang lebih baik dengan nilai R dan A yang sebanding. Hal tersebut menunjukkkan bahwa sistem mampu menyeimbangkan antara menentukan kelas yang positif atau *good* dengan yang negatif atau *bad* dengan seimbang tanpa ada kecenderungan yang berarti.

Penelitian menunjukkan *Cosine Similarity* lebih konsisten dari pada *Soft-Cosine Semantic Similarity* dalam hasil MAP, *Precision*, *Recall*, *Accuracy* dan F1 pada kasus *Question-Question* dan *Question-Comment.* Konsisten yang dimaksud di sini adalah tidak ada perbedaan hasil uji yang besar disetiap uji dan cenderung mendapat hasil positif.

Sedangkan untuk mendapatkan nilai maksimal pada tiap kasus didapatkan kombinasi yang berbeda. Pada kasus *Question-Question* lebih baik menggunakan *Cosine Similarity* dan SVM, meskipun nilai MAP rendah dari *Cosine Similarity* dan LR tetapi nilai *Precision*, *Recall*, *Accuracy* dan F1 lebih konsisten. Sedangkan pada kasus *Question-Comment* lebih baik menggunakan *Soft-Cosine Semantic Similarity* dengan pendekatan relasi e*dit-distance based relations* dan SVM yang mendapatkan rata-rata hasil MAP, *Precision*, *Recall*, *Accuracy* dan F1 dengan perbedaan tidak terlalu jauh. Lalu pada kasus *Question-External Comment* lebih baik menggunakan *Soft-*

Cosine Semantic Similarity dengan pendekatan relasi e*dit-distance based relations* dan SVM karena hasil MAP yang paling tinggi ditambah dengan perbedaan antara *Recall* dan *Accuracy* tidak terlalu jauh.
Untuk fitur word matching digunakan untuk semua data sebagai pembersih data yang hampir pasti tidak berguna.

5. Kesimpulan

5.1. Kesimpulan

Berdasarkan hasil pengujian dan analisis dapat ditarik kesimpulan bahwa fitur *Soft-Cosine Semantic Similarity* masih bisa bersaing dengan *Cosine Similarity* untuk mencari komentar relevan pada CQA, lalu *Soft-Cosine Semantic* Similarity berdampak besar terhadap performa namun kurang efektif karena hasil yang kurang konsisten. Dengan tambahan klasifikasi SVM maka fitur *Soft-Cosine Semantic Similarity* mendapatkan nilai MAP paling besar yaitu 21.0% dengan nilai *recall* lebih dari 50%.

5.2. Saran

Dilihat dari hasil pengujian dan analisa yang didapat maka ada beberapa saran yang diberikan untuk pengembangan yang lebih baik seperti memberikan parameter yang sesuai pada *classifier* sehingga mendapatkan hasil klasifikasi yang lebih baik dan menggunakan cara memberikan peringkat yang berbeda.

Daftar Pustaka

[1] D. Charlet and G. Damnati, "SimBow at SemEval-2017 Task 3: Soft-Cosine Semantic Similarity between Questions for Community Question Answering," *Proceedings of the 11th International Workshop on Semantic Evaluations,* pp. 315-319, 2017.

[2] P. Nakov, D. Hoogeveen, L. Marquez, A. Moschitti, H. Mubarak, T. Baldwin and K. Verspoor, "SemEval-2017 Task 3: Community Question Answering," *Proceedings of the 11th International Workshop,* pp. 27-48, 2017.

[3] T. Bray, P. Jean, C. M. Sperberg-McQueen, E. Maler and F. Yergeau, "Extensible Markup Language," *W3C,* 2006.

[4] R. Feldman and J. Sanger, "The text mining handbook: advanced approaches in analyzing unstructured data," 2007.

[5] Q. H. Tran, V. D. Tran, T. T. Vu, M. L. Nguyen and S. B. Pham, "JAIST: Combining multiple features for Answer Selection in Community Question Answering," *The 9th International Workshop on Semantic Evaluation,* pp. 215-219, 2015.

[6] Pennsylvania State University, "STAT 504 Analysis of Discrete Data," [Online]. Available: https://onlinecourses.science.psu.edu/stat504/node/149/. [Accessed May 2018].

[7] N. Cristianini and J. Shawe-Taylor, "An introduction to support vector machines and other kernel-based learning methods," 2000.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos and D. Cournapeau, "Scikit-learn: Machine Leaning in Python," *Journal of Machine Leaning Research,* 2011.

[9] S. Bird and E. Loper, "NLTK: The Natural language Toolkit".

[10] D. Jurafsky and J. H. Martin, "Question Answering," in *Speech and Language Processing*, 2017.

[11] Y. Yue, T. Finley, F. Radlinski and T. Joachims, "A Support Vector Method for Optimizing Average Precision," *SIGIR,* pp. 271-278, 2007.

[12] R. Rehurek and P. Sojka, "Gensim - Statistical Semantic in Python," Masaryk University, 2011.

[13] D. M. W. Powers, "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation," *Journal of Machine Learning Tecknologies,* vol. 2, no. 1, 2011.

[14] M. Sokolova, N. Japkowicz and S. Szpakowicz, "Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation," 2006.