

Nama : Hariel Hermawan  
Prodi : Pendidikan Teknologi Informasi  
Kelas : E

## Rangkuman

### 1. Natural Language

Natural Language Processing (NLP) merupakan salah satu cabang ilmu AI yang berfokus pada pengolahan bahasa natural. Bahasa natural adalah bahasa yang

Ada berbagai terapan aplikasi dari NLP. Diantaranya adalah Chatbot (aplikasi yang membuat user bisa seolah-olah melakukan komunikasi dengan computer), Stemming atau Lemmatization (pemotongan kata dalam bahasa tertentu menjadi bentuk dasar pengenalan fungsi setiap kata dalam kalimat), Summarization (ringkasan dari bacaan), Translation

Tools (menterjemahkan

bahasa) dan

aplikasi-aplikasi

lain

yang

memungkinkan komputer mampu memahami instruksi bahasa yang diinputkan oleh user.

Contoh Aplikasi NLP

Penelitian yang dikerjakan oleh Suhartono, Christiandy, dan Rolando (2013) adalah merancang sebuah algoritma lemmatization untuk Bahasa Indonesia. Algoritma ini dibuat untuk menambahkan fungsionalitas pada algoritma Stemming yang sudah pernah dikerjakan sebelumnya yaitu Enhanced Confix-Stripping Stemmer (ECS) yang dikerjakan pada tahun 2009. ECS sendiri merupakan pengembangan dari algoritma Confix-Stripping Stemmer yang dibuat pada tahun 2007. Pengembangan yang dikerjakan terdiri dari beberapa rule tambahan dan modifikasi dari rule sebelumnya.

Langkah untuk

melakukan suffix backtracking juga ditambahkan.

Hal ini

untuk

menambah akurasi.

Secara mendasar, algoritma lemmatization ini tidak bertujuan untuk mengembangkan dari metode ECS, karena tujuannya berbeda. Algoritma lemmatization bertujuan untuk memodifikasi ECS, supaya lebih tepat dengan konsep lemmatization.

Namun

demikian, masih ada beberapa kemiripan pada proses yang ada pada ECS. Ada beberapa kasus yang mana ECS belum berhasil untuk digunakan, namun bisa

diselesaikan pada algoritma lemmatization ini.

Pengujian validitas pada algoritma ini adalah dengan menggunakan beberapa artikel yang ada di Kompas, dan diperoleh hasil sebagai berikut:

| Category      | FULL         |              |              |            |                | UNIQUE       |             |             |            |                |
|---------------|--------------|--------------|--------------|------------|----------------|--------------|-------------|-------------|------------|----------------|
|               | T            | V            | S            | E          | P              | T            | V           | S           | E          | P              |
| Business      | 6344         | 5627         | 5550         | 77         | 0.98632        | 1868         | 1580        | 1559        | 21         | 0.98671        |
| Regional      | 6470         | 4802         | 5846         | 81         | 0.98313        | 1213         | 1011        | 995         | 16         | 0.98417        |
| Education     | 4165         | 5927         | 3598         | 32         | 0.99460        | 868          | 637         | 623         | 14         | 0.97802        |
| Science       | 6246         | 5504         | 5398         | 73         | 0.98674        | 874          | 643         | 630         | 13         | 0.97978        |
| Sports        | 6231         | 3242         | 5522         | 42         | 0.98705        | 838          | 608         | 604         | 4          | 0.99342        |
| International | 10953        | 3630         | 9917         | 75         | 0.97934        | 2037         | 1593        | 1575        | 18         | 0.98870        |
| Megapolitan   | 3998         | 5471         | 3214         | 28         | 0.99488        | 610          | 302         | 297         | 5          | 0.98344        |
| National      | 5499         | 5564         | 4764         | 38         | 0.99317        | 559          | 326         | 324         | 2          | 0.99387        |
| <i>Oasis</i>  | 6087         | 9992         | 5462         | 42         | 0.99580        | 820          | 528         | 524         | 4          | 0.99242        |
| Travel        | 8379         | 7502         | 7457         | 45         | 0.99400        | 892          | 611         | 607         | 4          | 0.99345        |
| <b>All</b>    | <b>64372</b> | <b>57261</b> | <b>56728</b> | <b>533</b> | <b>0.99069</b> | <b>10579</b> | <b>7839</b> | <b>7738</b> | <b>101</b> | <b>0.98712</b> |

Hasil dari pengujian menunjukkan bahwa akurasi yang diperoleh sekitar 98.71%.

T = Total data count

V = Valid test data count

S = Successful lemmatization

E = Error / Kegagalan

P = Precision

Aplikasi NLP yang lainnya adalah seperti penerjemah bahasa, chatting dengan komputer, meringkas satu bacaan yang panjang, pengecekan grammar dan lain sebagainya.

## 2. Machine Language

Kode mesin, yang juga dikenal sebagai machine language atau kode asli, adalah bahasa dasar komputer. Kode ini dibaca oleh unit pemrosesan pusat (CPU) komputer, terdiri dari angka biner digital, dan tampak seperti rangkaian angka nol dan satu yang sangat panjang. Kode biner adalah satu-satunya bahasa yang dapat dipahami oleh perangkat keras komputer.

Setiap CPU dikaitkan dengan arsitektur set instruksi (ISA) tertentu yang menentukan serangkaian operasi yang dapat dipahami dan dilakukan CPU. ISA dibangun ke dalam mikroarsitektur prosesor, yang bertindak sebagai antarmuka antara perangkat keras dan perangkat lunak. ISA menentukan instruksi mana yang dapat dikirimkan ke CPU dan bagaimana instruksi tersebut harus diformat.

Prosesor membaca instruksi dan melakukan tugas yang ditentukan oleh instruksi tersebut. Setiap instruksi terdiri dari sejumlah bit tertentu dan mencakup satu opcode dan satu atau lebih operand. Opcode memberi tahu komputer apa yang harus dilakukan, dan operand memberi tahu komputer data apa yang harus digunakan.

Bergantung pada prosesor, ISA mungkin mengharuskan semua instruksi memiliki panjang yang sama, atau mungkin mengizinkan instruksi dengan panjang yang berbeda-beda. Arsitektur prosesor menentukan panjang instruksi dan bagaimana setiap instruksi dipolakan. Eksekusi instruksi dikontrol oleh firmware atau kabel internal CPU.

Contoh penggunaan set instruksi MIPS

ISA yang telah banyak diterapkan adalah Mikroprosesor tanpa Tahapan Pipa yang Terkunci (MIPS), yang didasarkan pada arsitektur komputer set instruksi yang diperkecil (RISC). Semua instruksi MIPS panjangnya 32 bit, dengan operand yang ditentukan dalam enam bit pertama. MIPS mendukung tiga jenis instruksi:

- ▣ Register (tipe R atau format R). Semua nilai data operand dalam instruksi ada dalam register.

- ▣ Langsung (tipe I atau format I). Satu nilai data operand ditetapkan dalam instruksi, dan nilai data lain yang ditetapkan, baik satu atau dua, ada dalam register.

- ▣ Lompat (tipe J atau format J). Instruksi ini digunakan untuk melompat ke lokasi yang ditentukan dalam instruksi.

Setiap jenis instruksi diformat sedikit berbeda dari yang lain karena mengandung jumlah elemen yang berbeda, meskipun semuanya masih memiliki panjang 32-bit.

Misalnya, instruksi tipe-R mencakup enam elemen, yang paling banyak dari ketiganya, seperti yang ditunjukkan dalam sintaks berikut:

op rs rt rd poros fungsi

Setiap elemen sintaks menempati sejumlah bit tertentu dalam keseluruhan instruksi 32-bit. Elemen sintaks juga harus disediakan dalam urutan yang ditentukan. Tabel berikut menjelaskan setiap elemen dan mencantumkan jumlah bitnya.

Contoh arsitektur set instruksi MIPS.

Misalkan instruksi tipe-R

Contoh arsitektur set instruksi MIPS.

| Element | Bits | Value  |
|---------|------|--|
| op      | 6    | Instruction's opcode, which is always 0 when using the R-type. |
| rs      | 5    | First source register (between 0-31).                          |
| rt      | 5    | Second source register (between 0-31).                         |
| rd      | 5    | Destination register (between 0-31).                           |
| shamt   | 5    | Instructions for shifting bits; 0 indicates no shifting.       |
| func    | 6    | Operation to be performed.                                     |

disebutkan di atas, dan fungsi penjumlahan tipe-R adalah 32, seperti yang didefinisikan oleh MIPS. Tabel berikut menunjukkan nilai untuk setiap elemen dalam instruksi. Baris pertama (setelah header) dalam teks biasa, dan baris kedua menyediakan padanan biner untuk setiap nilai.

Nilai setiap elemen dalam contoh arsitektur set instruksi MIPS.

| op     | rs    | rt    | rd    | shamt | func   |
|--------|-------|-------|-------|-------|--------|
| 0      | 17    | 18    | 16    | 0     | 32     |
| 000000 | 10001 | 10010 | 10000 | 00000 | 100000 |

Misalkan instruksi tipe-R digunakan untuk menambahkan dua nilai bersama-sama. Nilai pertama ada di register 17, dan yang kedua ada di register 18. Jumlah dari dua nilai akan disimpan di register 16. Opcode tipe-R adalah 0, seperti yang

Ketika nilai biner pada baris bawah digabungkan, mereka menghasilkan keseluruhan instruksi 32-bit yang dikirimkan ke CPU.

**Nilai biner** berikut menunjukkan instruksi secara keseluruhan, dibagi menjadi empat oktet:

00000010 00110010 10000000 00100000

Setiap instruksi MIPS harus diserahkan ke CPU dalam format 32-bit yang tepat. Operand menunjukkan operasi dan jenis instruksi. Jika operand adalah 000000, maka itu adalah tipe-R, dalam hal ini, tujuan instruksi ditunjukkan oleh enam bit terakhir (func).

### Berpindah dari kode sumber ke kode mesin

Sebagian besar perangkat lunak dikembangkan dalam **bahasa pemrograman** yang dapat dibaca manusia, seperti C++, C#, Java, PHP, Python atau Swift, yang semuanya dianggap sebagai bahasa tingkat tinggi. **Kode sumber** disimpan sebagai berkas teks yang akhirnya diterjemahkan ke dalam kode mesin oleh **kompiler**, **assembler** atau interpreter. Pendekatan yang tepat bergantung pada bahasa pemrograman dan platform target.

Salah satu pendekatan yang umum adalah menggunakan kompiler untuk menerjemahkan kode sumber ke dalam kode rakitan. Kode rakitan tersebut kemudian



Diagram yang mengilustrasikan ikhtisar proses konversi kode sumber

Bahasa seperti **Java** dan **C#** mengambil pendekatan berbeda untuk berpindah dari kode sumber ke kode mesin. Kode tersebut tetap berjalan melalui kompiler, tetapi kompiler tidak menghasilkan kode assembly. Sebaliknya, ia menghasilkan **bytecode** atau jenis bahasa perantara lainnya. Kode perantara tersebut kemudian diserahkan ke penerjemah untuk setiap platform target. Penerjemah menghasilkan kode mesin untuk platform tersebut.

Dalam beberapa kasus, pengembang akan menulis **perangkat lunak** secara langsung dalam kode assembly, sehingga menghilangkan kebutuhan akan kompiler. Bahasa assembly dianggap sebagai bahasa tingkat rendah seperti kode mesin, kecuali bahwa bahasa assembly ditulis dalam teks biasa dan dapat dibaca manusia. Kode assembly dikirimkan langsung ke assembler, yang mengubah bahasa assembly menjadi kode mesin.

Kode assembly lebih sesuai langsung dengan kode mesin daripada bahasa pemrograman tingkat tinggi. Kode assembly menggunakan kata-kata deskriptif, yang dikenal sebagai *mnemonik*, yang sesuai dengan operan ISA. Misalnya, contoh MIPS di atas menambahkan dua nilai. Operan dalam kasus ini adalah 32, atau 100000 dalam biner. Mnemonik kode assembly untuk operasi ini adalah *add*.

Programmer manusia jarang, bahkan mungkin tidak pernah, berurusan langsung dengan kode mesin lagi. Jika pengembang men-debug **program** pada level rendah, mereka mungkin menggunakan cetakan yang menunjukkan program dalam bentuk kode mesin. Cetakan, yang disebut dump, bisa sangat sulit untuk dikerjakan. Beberapa utilitas dump menampilkan data sebagai nilai heksadesimal karena lebih mudah dibaca.

