BAB 1
PENDAHULUAN

1.1 Latar Belakang Masalah

Part-of-speech, atau POS, adalah kelas kata yang diberikan kepada sebuah kata berdasarkan konteks kalimat yang memuatnya. Seperti kata *Apa* pada kalimat *ular apa ini?* memiliki kelas kata *pronoun* dan pada kalimat *apa besok ada ulangan*? memiliki kelas kata *particle*. *Part-of-speech* menjadi komponen penting bagi sistem lain yang berhubungan dengan bahasa karena dapat membantu meningkatkan akurasi [1] seperti NER, *speech preprocessing, information extraction*.

POS Tagging, proses menandai kata dengan *part-of-speech* yang tepat adalah pekerjaan yang memakan waktu dan tenaga apabila dikerjakan manusia [2] sehingga diperlukan otomatisasi. Penelitian terbaru [3] [4] [5] mencoba melakukan otomatisasi tersebut menggunakan algoritma berbasis *neural network* dimana diketahui bi-LSTM mengungguli performa algoritma lain. Namun bi-LSTM memiliki kekurangan dimana algoritma ini memiliki arsitektur yang kompleks sehingga beban komputasi menjadi tinggi terutama ketika diterapkan pada kasus skala besar [6]. Algoritma *neural network* memiliki kelebihan ketimbang algoritma lain [7], antara lain adalah dapat memanfaatkan *word/char embedding* yang bisa mengurangi dampak *unseen word* yang mana diketahui menurunkan performa *POS tagger* [2] serta *neural network* bisa mengurangi beban kerja dalam *feature engineering*. Oleh karena itu masih diperlukan alternatif algoritma *neural network* lain yang bisa menyaingi atau mendekati performa bi-LSTM namun dengan kompleksitas lebih rendah.

Berdasarkan studi literatur, terdapat penelitian yang menunjukkan performa Elman kompetitif dengan LSTM [8]. Elman pun memiliki arsitektur 4 kali lebih rendah dari LSTM [9]. Kemudian terdapat penelitian yang menyatakan Elman unggul dari CRF [10] [11] dimana CRF adalah algoritma peringkat dua setelah bi-LSTM untuk performa tertinggi. Elman adalah algoritma yang termasuk ke dalam keluarga *neural network* dan *deep learning* dimana arsitekturnya menyerupai rantai yang dimaksudkan untuk mengalirkan informasi dari langkah atau data

sebelumnya. Informasi konteks ini dimanfaatkan oleh banyak penelitian untuk digunakan pada kasus data sekuensial seperti *POS tagging*.

Dari masalah dan fakta yang telah disebutkan, penelitian ini akan menggunakan Elman untuk diterapkan pada kasus *POS tagging* bahasa Indonesia untuk diketahui performanya. Dataset diambil dari penelitian Kurniawan dan Aji [5] agar hasil penelitian mudah dibandingkan. Dataset ini merupakan buatan Dinakarami et. al [12] tetapi sudah dilakukan proses *splitting* menjadi *train set, validation set* dan *test set* untuk keperluan *K-cross validation* oleh Kurniawan dan Aji [5].

1.2 Identifikasi Masalah

Dari uraian latar belakang masalah di atas, kekurangan bi-LSTM sebagai algoritma terunggul saat ini adalah kompleksitas arsitektur yang dimiliki sehingga menyebabkan beban komputasi menjadi tinggi. Oleh karena itu masih diperlukan alternatif neural network lain yang lebih rendah kompleksitasnya namun bisa menyaingi atau mendekati performa metode berbasis LSTM mengingat kelebihan yang dimiliki neural network. Karakteristik solusi dari masalah tersebut ada pada Elman namun sampai saat ini performa algoritma ini belum diketahui pada *POS tagging* bahasa Indonesia.

1.3 Maksud dan Tujuan

Maksud dari penelitian ini adalah membuat program POS tagger untuk bahasa Indonesia menggunakan *Elman Recurrent Neural Network*. Sedangkan tujuan penelitian yang ingin dicapai adalah untuk mengetahui performa F1 score, akurasi dan waktu komputasi *Elman Recurrent Neural Network* pada *POS tagging* bahasa Indonesia.

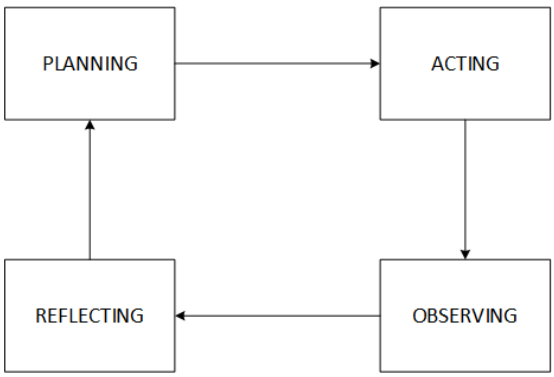
1.4 Batasan Masalah

Adapun batasan masalah yang diterapkan adalah sebagai berikut.

1. Data yang digunakan adalah dataset yang berasal dari Dinakaramani et. al yang sudah melalui proses splitting oleh Kurniawan dan Aji.
2. Format data adalah file *.tsv*
3. Data akan direpresentasikan menggunakan one-hot encoding.
4. Tidak ada penanganan khusus token *multi-word*.
5. Metode evaluasi yang dipakai adalah akurasi dan F_1 $score$ dengan K-cross validation.

1.5 Metode Penelitian

Metode penelitian yang digunakan adalah penelitian tindakan atau *action research*. Metode ini merupakan upaya untuk menguji dan mencoba ide-ide ke dalam praktek dengan tujuan memperbaiki atau mengubah sesuatu [13]. Hal tersebut sesuai dengan penelitian ini dimana kekurangan-kekurangan penelitian sebelumnya dijadikan dasar untuk membangun penelitian. Adapun tahapan-tahapan dalam penelitian ini bisa dilihat pada Gambar 1.1.



Gambar 1.1 Tahapan Penelitian

Berikut adalah penjelasan dari masing-masing tahapan yang akan dilakukan penelitian ini.

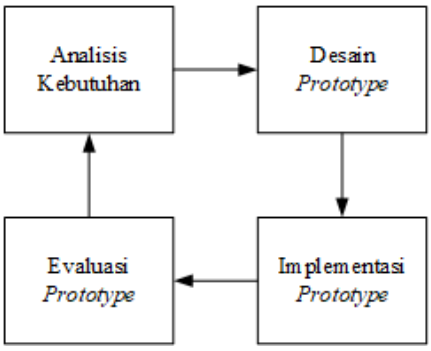
1. *Planning* (perencanaan)

   Pada tahap ini penelitian-penelitian sebelumnya ditinjau untuk melihat kekurangannya kemudian untuk mengatasi kekurangan-kekurangan tersebut

maka dilakukan identifikasi calon solusi dengan menelusuri berbagai literatur. Solusi yang akan dibangun kemudian dianalisis dan dirancang untuk tahap implementasi atau pelaksanaan.

2. *Acting* (pelaksanaan)

   Pada tahap ini calon solusi yang berupa algoritma akan diimplementasikan ke dalam bentuk program komputer baik sisi *preprocessing*, proses utama maupun sisi antarmuka menggunakan metode pembangunan perangkat lunak *Prototyping*. Alasan penggunaan metode pembangunan perangkat lunak tersebut didasari oleh kemungkinan program yang akan dibangun berubah-ubah selama proses penelitian, menyesuaikan kebutuhan dalam rangka meningkatkan performa algoritma. Adapun alur dari metode ini dapat dilihat pada Gambar 1.2.



Gambar 1.2 Metode *Prototyping* [14]

Berikut penjelasan mengenai setiap tahapan dari metode pembangunan perangkat lunak tersebut:

a. Analisis Kebutuhan

   Umumnya pada tahap ini seorang *system analyst* melakukan analisis kebutuhan yang diperlukan *client* dengan cara seperti wawancara. Namun hal tersebut tidak dilakukan dalam penelitian ini karena tidak perlu sebab segala kebutuhan sudah diketahui ketika tahap *Planning*, yaitu tahapan pertama dari metode penelitian. Pada tahap ini hanya dilakukan identifikasi kebutuhan non-fungsional.

b. Desain *Prototype*

Merancang sistem yang akan dibangun terutama kebutuhan fungsional, yaitu diagram konteks, DFD, spesifikasi proses, *mockup,* jaringan semantik, dan algoritma.

c. Implementasi *Prototype*

Proses implementasi menggunakan bahasa pemrograman Python dan alat-alat penunjangnya.

d. Evaluasi *Prototype*

Pada tahap ini sistem yang telah dibangun akan diuji untuk mencari kekurangan dan *bug* yang ada dengan metode *black box* dan *white box*. Apabila masih ada kekurangan maka kembali ke tahap analisis kebutuhan, desain *prototype,* implementasi *prototype* dan akhirnya kembali lagi ke evaluasi *prototype*.

3. *Observing* (pengamatan)

Pada tahap ini dilakukan pengujian terhadap Elman yang dimulai dari pengujian parameter untuk mendapatkan parameter terbaik yang kemudian akan digunakan pada pengujian pengujian performa dan terakhir pengujian waktu komputasi.

4. *Reflecting* (refleksi)

Pada tahap ini performa yang dihasilkan oleh algoritma akan dievaluasi lalu akan dilakukan perbandingan dengan penelitian sebelumnya. Hasil evaluasi akan mengungkap perbaikan yang diperlukan untuk meningkatkan performa. Informasi perbaikan yang masih diperlukan akan dibawa ke tahap awal penelitian, yaitu *Planning,* untuk nantinya diimplementasikan bila memungkinkan.

1.6 Sistematika Penulisan

Sistematika penulisan laporan akan dibagi menjadi beberapa bab dengan pokok pembahasan sebagai berikut.

BAB 1 PENDAHULUAN

Bab ini berisi latar belakang masalah, identifikasi masalah, maksud dan tujuan penelitian, batasan masalah, metode penelitian, serta sistematika penulisan.

BAB 2 LANDASAN TEORI

Bab ini membahas teori-teori yang berkaitan dengan sistem yang akan dibangun maupun teori dasar seperti penjelasan *POS tagging*, algoritma utama dan bagian-bagian dari algoritma, metode pengujian dan sebagainya. Selain itu hasil penelitian-penelitian sebelumnya dipaparkan pada bab ini.

BAB 3 ANALISIS DAN PERANCANGAN

Bab ini berisi analisis masalah, analisis solusi (yaitu rancangan algoritma yang akan dibangun), analisis kebutuhan non-fungsional, analisis kebutuhan fungsional, perancangan antarmuka, jaringan semantik dan perancangan prosedural.

BAB 4 IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi implementasi sistem yang terdiri dari implementasi perangkat keras, implementasi perangkat lunak, implementasi antarmuka, serta pengujian sistem yang terdiri dari pengujian fungsionalitas, pengujian parameter, pengujian performa, pengujian waktu komputasi dan analisis hasil pengujian performa.

BAB 5 KESIMPULAN DAN SARAN

Bab ini berisi pemaparan kesimpulan dari hasil penelitian yang sudah dilakukan dan saran untuk pengembangan penelitian selanjutnya.